
Linux Standard Base (LSB) —
Part 1-6:
Graphics and Gtk3 specification

Noyau de base normalisé Linux (LSB) —
Partie 1-6: Spécification graphique et Gtk3



STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2021

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by the Linux Foundation as Linux Standard Base (LSB): Graphics and gtk3 specification and drafted in accordance with its editorial rules. It was assigned to Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*, and adopted by National Bodies.

This first edition of ISO/IEC TS 23360-1-6 cancels and replaces ISO/IEC 23360-1:2006, which has been technically revised.

This document is based on "The GNU Free Documentation License, version 1.1". The license is available at <https://www.gnu.org/licenses/old-licenses/fdl-1.1.html>

A list of all parts in the ISO/IEC 23660 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Contents

Foreword	iii
Introduction	v
I Introductory Elements	1
1 Scope	2
2 Normative References.....	3
3 Requirements	4
3.1 TUM Libraries	4
3.2 Gtk3 Libraries	4
4 Terms and Definitions	5
5 Documentation Conventions.....	7
II PNG15 library	8
6 Libraries	9
6.1 Interfaces for libpng15	9
6.2 Interface Definitions for libpng15	1 3
III GTK+ Stack Libraries	7 8
7 Libraries	7 9
7.1 Introduction.....	7 9
7.2 Interfaces for libgdk-3.....	7 9
7.3 Data Definitions for libgdk-3.....	9 1
7.4 Interfaces for libgtk-3.....	1 7 1
7.5 Data Definitions for libgtk-3.....	2 4 5
Annex A Alphabetical Listing of Interfaces by Library	4 4 6
A.1 libpng15.....	4 4 6
A.2 libgdk-3.....	4 5 0
A.3 libgtk-3.....	4 5 9

Introduction

The Trial Use Specification describes components which may or may not be present on an otherwise conforming system. The purpose is to indicate that these components are on a Standards Track, that is, they are intended to become part of the LSB Specification in a future edition.

This document should be used in conjunction with the documents it references. Information referenced in this way is as much a part of this document as is the information explicitly included here.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

I Introductory Elements

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

1 Scope

The Trial Use Specification defines components which are not required parts of the LSB Specification.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

2 Normative References

The specifications listed below are referenced in whole or in part by the Trial Use Specification. Such references may be normative or informative; a reference to specification shall only be considered normative if it is explicitly cited as such. The Trial Use Specification may make normative references to a portion of these specifications (that is, to define a specific function or group of functions); in such cases, only the explicitly referenced portion of the specification is to be considered normative.

Table 2-1 Informative References

Name	Title	URL
Gdk 3.6.4 Reference Manual	Gdk 3.6.4 Reference Manual	http://developer.gnome.org/gdk3/3.6
Gtk 3.6.4 Reference Manual	Gtk 3.6.4 Reference Manual	http://developer.gnome.org/gtk3/3.6
ISO C (1999)	ISO/IEC 9899:1999 - Programming Languages -- C	

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

3 Requirements

3.1 TUM Libraries

The libraries listed in Table 3-1 shall be available on a Linux Standard Base system which implements the Trial Use TUM module, with the specified runtime names. This list may be supplemented or amended by the architecture-specific specification.

Table 3-1 Standard Library Names

Library	Runtime Name
libpng15	libpng15.so.15

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

3.2 Gtk3 Libraries

The libraries listed in Table 3-2 shall be available on a Linux Standard Base system which implements the Trial Use Toolkit_Gtk3 module, with the specified runtime names. This list may be supplemented or amended by the architecture-specific specification.

Table 3-2 Standard Library Names

Library	Runtime Name
libgdk-3	libgdk-3.so.0
libgtk-3	libgtk-3.so.0

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

4 Terms and Definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 2382, ISO 80000-2, and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <http://www.electropedia.org/>

4.1

archLSB

Some LSB specification documents have both a generic, architecture-neutral part and an architecture-specific part. The latter describes elements whose definitions may be unique to a particular processor architecture. The term archLSB may be used in the generic part to refer to the corresponding section of the architecture-specific part.

4.2

Binary Standard, ABI

The total set of interfaces that are available to be used in the compiled binary code of a conforming application, including the run-time details such as calling conventions, binary format, C++ name mangling, etc.

4.3

Implementation-defined

Describes a value or behavior that is not defined by this document but is selected by an implementor. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementor shall document such a value or behavior so that it can be used correctly by an application.

4.4

Shell Script

A file that is read by an interpreter (e.g., awk). The first line of the shell script includes a reference to its interpreter binary.

4.5

Source Standard, API

The total set of interfaces that are available to be used in the source code of a conforming application. Due to translations, the Binary Standard and the Source Standard may contain some different interfaces.

4.6

Undefined

Describes the nature of a value or behavior not defined by this document which results from use of an invalid program construct or invalid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

4.7

Unspecified

Describes the nature of a value or behavior not specified by this document which results from use of a valid program construct or valid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

In addition, for the portions of this specification which build on IEEE Std 1003.1-2001, the definitions given in *IEEE Std 1003.1-2001, Base Definitions, Chapter 3* apply.

5 Documentation Conventions

Throughout this document, the following typographic conventions are used:

`function()`

the name of a function

command

the name of a command or utility

CONSTANT

a constant value

parameter

a parameter

variable

a variable

Throughout this specification, several tables of interfaces are presented. Each entry in these tables has the following format:

name

the name of the interface

(symver)

An optional symbol version identifier, if required.

[*refno*]

A reference number indexing the table of referenced specifications that follows this table.

For example,

forkpty(GLIBC_2.0) [SUSv4]

refers to the interface named `forkpty()` with symbol version `GLIBC_2.0` that is defined in the reference indicated by the tag `SUSv4`.

Note: For symbols with versions which differ between architectures, the symbol versions are defined in the architecture specific parts of of this module specification only. In the generic part, they will appear without symbol versions.

II PNG15 library

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

6 Libraries

6.1 Interfaces for libpng15

Table 6-1 defines the library name and shared object name for the libpng15 library

Table 6-1 libpng15 Definition

Library:	libpng15
SONAME:	libpng15.so.15

The behavior of the interfaces in this library is specified by the following specifications:

[LSB] This Specification

6.1.1 libpng15 interfaces

6.1.1.1 Interfaces for libpng15 interfaces

An LSB conforming implementation shall provide the generic functions for libpng15 interfaces specified in Table 6-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-2 libpng15 - libpng15 interfaces Function Interfaces

png_access_version_number(PNG15_0) [LSB]	png_benign_error(PNG15_0) [LSB]	png_build_grayscale_palette(PNG15_0) [LSB]
png_calloc(PNG15_0) [LSB]	png_chunk_benign_error(PNG15_0) [LSB]	png_chunk_error(PNG15_0) [LSB]
png_chunk_warning(PNG15_0) [LSB]	png_convert_from_struct_tm(PNG15_0) [LSB]	png_convert_from_time_t(PNG15_0) [LSB]
png_convert_to_rfc1123(PNG15_0) [LSB]	png_create_info_struct(PNG15_0) [LSB]	png_create_read_struct(PNG15_0) [LSB]
png_create_read_struct_2(PNG15_0) [LSB]	png_create_write_struct(PNG15_0) [LSB]	png_create_write_struct_2(PNG15_0) [LSB]
png_data_freer(PNG15_0) [LSB]	png_destroy_info_struct(PNG15_0) [LSB]	png_destroy_read_struct(PNG15_0) [LSB]
png_destroy_write_struct(PNG15_0) [LSB]	png_error(PNG15_0) [LSB]	png_free(PNG15_0) [LSB]
png_free_data(PNG15_0) [LSB]	png_free_default(PNG15_0) [LSB]	png_get_IHDR(PNG15_0) [LSB]
png_get_PLTE(PNG15_0) [LSB]	png_get_bKGD(PNG15_0) [LSB]	png_get_bit_depth(PNG15_0) [LSB]
png_get_cHRM(PNG15_0) [LSB]	png_get_cHRM_XYZ(PNG15_0) [LSB]	png_get_cHRM_XYZ_fixed(PNG15_0) [LSB]
png_get_cHRM_fixed(PNG15_0) [LSB]	png_get_channels(PNG15_0) [LSB]	png_get_chunk_cache_max(PNG15_0) [LSB]

png_get_chunk_malloc_max(PNG15_0) [LSB]	png_get_color_type(PNG15_0) [LSB]	png_get_compression_buffer_size(PNG15_0) [LSB]
png_get_compression_type(PNG15_0) [LSB]	png_get_copyright(PNG15_0) [LSB]	png_get_current_pass_number(PNG15_0) [LSB]
png_get_current_row_number(PNG15_0) [LSB]	png_get_error_ptr(PNG15_0) [LSB]	png_get_filter_type(PNG15_0) [LSB]
png_get_gAMA(PNG15_0) [LSB]	png_get_gAMA_fixed(PNG15_0) [LSB]	png_get_hIST(PNG15_0) [LSB]
png_get_header_version(PNG15_0) [LSB]	png_get_header_version(PNG15_0) [LSB]	png_get_iCCP(PNG15_0) [LSB]
png_get_image_height(PNG15_0) [LSB]	png_get_image_width(PNG15_0) [LSB]	png_get_int_32(PNG15_0) [LSB]
png_get_interlace_type(PNG15_0) [LSB]	png_get_io_chunk_name(PNG15_0) [LSB]	png_get_io_chunk_type(PNG15_0) [LSB]
png_get_io_ptr(PNG15_0) [LSB]	png_get_io_state(PNG15_0) [LSB]	png_get_libpng_version(PNG15_0) [LSB]
png_get_mem_ptr(PNG15_0) [LSB]	png_get_offs(PNG15_0) [LSB]	png_get_pCAL(PNG15_0) [LSB]
png_get_pHYs(PNG15_0) [LSB]	png_get_pHYs_dpi(PNG15_0) [LSB]	png_get_pixel_aspect_ratio(PNG15_0) [LSB]
png_get_pixel_aspect_ratio_fixed(PNG15_0) [LSB]	png_get_pixels_per_inch(PNG15_0) [LSB]	png_get_pixels_per_meter(PNG15_0) [LSB]
png_get_progressive_ptr(PNG15_0) [LSB]	png_get_rgb_to_gray_status(PNG15_0) [LSB]	png_get_rowbytes(PNG15_0) [LSB]
png_get_rows(PNG15_0) [LSB]	png_get_sBIT(PNG15_0) [LSB]	png_get_sCAL(PNG15_0) [LSB]
png_get_sCAL_fixed(PNG15_0) [LSB]	png_get_sCAL_s(PNG15_0) [LSB]	png_get_sPLT(PNG15_0) [LSB]
png_get_sRGB(PNG15_0) [LSB]	png_get_signature(PNG15_0) [LSB]	png_get_tIME(PNG15_0) [LSB]
png_get_tRNS(PNG15_0) [LSB]	png_get_text(PNG15_0) [LSB]	png_get_uint_16(PNG15_0) [LSB]
png_get_uint_31(PNG15_0) [LSB]	png_get_uint_32(PNG15_0) [LSB]	png_get_unknown_chunks(PNG15_0) [LSB]
png_get_user_chunk_ptr(PNG15_0) [LSB]	png_get_user_height_max(PNG15_0) [LSB]	png_get_user_transform_ptr(PNG15_0) [LSB]
png_get_user_width_max(PNG15_0) [LSB]	png_get_valid(PNG15_0) [LSB]	png_get_x_offset_inches(PNG15_0) [LSB]
png_get_x_offset_inches_fixed(PNG15_0) [LSB]	png_get_x_offset_microns(PNG15_0) [LSB]	png_get_x_offset_pixels(PNG15_0) [LSB]

png_get_x_pixels_per_inch(PNG15_0) [LSB]	png_get_x_pixels_per_meter(PNG15_0) [LSB]	png_get_y_offset_inches(PNG15_0) [LSB]
png_get_y_offset_inches_fixed(PNG15_0) [LSB]	png_get_y_offset_microns(PNG15_0) [LSB]	png_get_y_offset_pixels(PNG15_0) [LSB]
png_get_y_pixels_per_inch(PNG15_0) [LSB]	png_get_y_pixels_per_meter(PNG15_0) [LSB]	png_handle_as_unknown(PNG15_0) [LSB]
png_info_init_3(PNG15_0) [LSB]	png_init_io(PNG15_0) [LSB]	png_longjmp(PNG15_0) [LSB]
png_malloc(PNG15_0) [LSB]	png_malloc_default(PNG15_0) [LSB]	png_malloc_warn(PNG15_0) [LSB]
png_permit_mng_features(PNG15_0) [LSB]	png_process_data(PNG15_0) [LSB]	png_process_data_pause(PNG15_0) [LSB]
png_process_data_skip(PNG15_0) [LSB]	png_progressive_combine_row(PNG15_0) [LSB]	png_read_end(PNG15_0) [LSB]
png_read_image(PNG15_0) [LSB]	png_read_info(PNG15_0) [LSB]	png_read_png(PNG15_0) [LSB]
png_read_row(PNG15_0) [LSB]	png_read_rows(PNG15_0) [LSB]	png_read_update_info(PNG15_0) [LSB]
png_reset_zstream(PNG15_0) [LSB]	png_save_int_32(PNG15_0) [LSB]	png_save_uint_16(PNG15_0) [LSB]
png_save_uint_32(PNG15_0) [LSB]	png_set_iHDR(PNG15_0) [LSB]	png_set_PLTE(PNG15_0) [LSB]
png_set_add_alpha(PNG15_0) [LSB]	png_set_alpha_mode(PNG15_0) [LSB]	png_set_alpha_mode_fixed(PNG15_0) [LSB]
png_set_bKGD(PNG15_0) [LSB]	png_set_background(PNG15_0) [LSB]	png_set_background_fixed(PNG15_0) [LSB]
png_set_benign_errors(PNG15_0) [LSB]	png_set_bgr(PNG15_0) [LSB]	png_set_cHRM(PNG15_0) [LSB]
png_set_cHRM_XYZ(PNG15_0) [LSB]	png_set_cHRM_XYZ_fixed(PNG15_0) [LSB]	png_set_cHRM_fixed(PNG15_0) [LSB]
png_set_check_for_invalid_index(PNG15_0) [LSB]	png_set_chunk_cache_max(PNG15_0) [LSB]	png_set_chunk_malloc_max(PNG15_0) [LSB]
png_set_compression_buffer_size(PNG15_0) [LSB]	png_set_compression_level(PNG15_0) [LSB]	png_set_compression_mem_level(PNG15_0) [LSB]
png_set_compression_method(PNG15_0) [LSB]	png_set_compression_strategy(PNG15_0) [LSB]	png_set_compression_window_bits(PNG15_0) [LSB]
png_set_crc_action(PNG15_0) [LSB]	png_set_error_fn(PNG15_0) [LSB]	png_set_expand(PNG15_0) [LSB]

png_set_expand_16(PNG15_0) [LSB]	png_set_expand_gray_1_2_4_to_8(PNG15_0) [LSB]	png_set_filler(PNG15_0) [LSB]
png_set_filter(PNG15_0) [LSB]	png_set_filter_heuristics(PNG15_0) [LSB]	png_set_filter_heuristics_fixed(PNG15_0) [LSB]
png_set_flush(PNG15_0) [LSB]	png_set_gAMA(PNG15_0) [LSB]	png_set_gAMA_fixed(PNG15_0) [LSB]
png_set_gamma(PNG15_0) [LSB]	png_set_gamma_fixed(PNG15_0) [LSB]	png_set_gray_to_rgb(PNG15_0) [LSB]
png_set_hIST(PNG15_0) [LSB]	png_set_iCCP(PNG15_0) [LSB]	png_set_interlace_handling(PNG15_0) [LSB]
png_set_invalid(PNG15_0) [LSB]	png_set_invert_alpha(PNG15_0) [LSB]	png_set_invert_mono(PNG15_0) [LSB]
png_set_keep_unknown_chunks(PNG15_0) [LSB]	png_set_longjmp_fn(PNG15_0) [LSB]	png_set_mem_fn(PNG15_0) [LSB]
png_set_oFFs(PNG15_0) [LSB]	png_set_pCAL(PNG15_0) [LSB]	png_set_pHYs(PNG15_0) [LSB]
png_set_packing(PNG15_0) [LSB]	png_set_packswap(PNG15_0) [LSB]	png_set_palette_to_rgb(PNG15_0) [LSB]
png_set_progressive_read_fn(PNG15_0) [LSB]	png_set_quantize(PNG15_0) [LSB]	png_set_read_fn(PNG15_0) [LSB]
png_set_read_status_fn(PNG15_0) [LSB]	png_set_read_user_chunk_fn(PNG15_0) [LSB]	png_set_read_user_transform_fn(PNG15_0) [LSB]
png_set_rgb_to_gray(PNG15_0) [LSB]	png_set_rgb_to_gray_fixed(PNG15_0) [LSB]	png_set_rows(PNG15_0) [LSB]
png_set_sBIT(PNG15_0) [LSB]	png_set_sCAL(PNG15_0) [LSB]	png_set_sCAL_fixed(PNG15_0) [LSB]
png_set_sCAL_s(PNG15_0) [LSB]	png_set_sPLT(PNG15_0) [LSB]	png_set_sRGB(PNG15_0) [LSB]
png_set_sRGB_gAMA_and_chRM(PNG15_0) [LSB]	png_set_scale_16(PNG15_0) [LSB]	png_set_shift(PNG15_0) [LSB]
png_set_sig_bytes(PNG15_0) [LSB]	png_set_strip_16(PNG15_0) [LSB]	png_set_strip_alpha(PNG15_0) [LSB]
png_set_swap(PNG15_0) [LSB]	png_set_swap_alpha(PNG15_0) [LSB]	png_set_tIME(PNG15_0) [LSB]
png_set_tRNS(PNG15_0) [LSB]	png_set_tRNS_to_alpha(PNG15_0) [LSB]	png_set_text(PNG15_0) [LSB]
png_set_text_compression_level(PNG15_0) [LSB]	png_set_text_compression_mem_level(PNG15_0) [LSB]	png_set_text_compression_method(PNG15_0) [LSB]

png_set_text_compression_strategy(PNG15_0) [LSB]	png_set_text_compression_window_bits(PNG15_0) [LSB]	png_set_unknown_chunk_location(PNG15_0) [LSB]
png_set_unknown_chunks(PNG15_0) [LSB]	png_set_user_limits(PNG15_0) [LSB]	png_set_user_transform_info(PNG15_0) [LSB]
png_set_write_fn(PNG15_0) [LSB]	png_set_write_status_fn(PNG15_0) [LSB]	png_set_write_user_transform_fn(PNG15_0) [LSB]
png_sig_cmp(PNG15_0) [LSB]	png_start_read_image(PNG15_0) [LSB]	png_warning(PNG15_0) [LSB]
png_write_chunk(PNG15_0) [LSB]	png_write_chunk_data(PNG15_0) [LSB]	png_write_chunk_end(PNG15_0) [LSB]
png_write_chunk_start(PNG15_0) [LSB]	png_write_end(PNG15_0) [LSB]	png_write_flush(PNG15_0) [LSB]
png_write_image(PNG15_0) [LSB]	png_write_info(PNG15_0) [LSB]	png_write_info_before_PLTE(PNG15_0) [LSB]
png_write_png(PNG15_0) [LSB]	png_write_row(PNG15_0) [LSB]	png_write_rows(PNG15_0) [LSB]
png_write_sig(PNG15_0) [LSB]		

An LSB conforming implementation shall provide the generic deprecated functions for libpng15 interfaces specified in Table 6-3, with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 6-3 libpng15 - libpng15 interfaces Deprecated Function Interfaces

png_get_io_chunk_name(PNG15_0) [LSB]		

6.2 Interface Definitions for libpng15

The interfaces defined on the following pages are included in libpng15 and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in Section 6.1 shall behave as described in the referenced base document.

png_access_version_number

Name

png_access_version_number — return version of the run-time library

Synopsis

```
#include <png.h>
png_uint_32 png_access_version_number(void);
```

Description

png_access_version_number() returns version of the libpng15 library available at run-time.

Return Value

png_access_version_number() returns integer constructed from the major version, minor version with leading zero and leading number with leading zero. For example, the version number for version 1.2.8 is 10208.

png_convert_from_struct_tm

Name

png_convert_from_struct_tm — convert struct tm to png_time

Synopsis

```
#include <png.h>
void png_convert_from_struct_tm(png_timep ptime, struct tm * ttime);
```

Description

This interface shall convert from the struct tm time format to the png_time format.

The parameter *ttime* shall specify the struct tm value to convert to png_time format.

The output parameter *ptime* shall contain the converted value.

png_convert_from_time_t

Name

png_convert_from_time_t — convert time_t to png_time

Synopsis

```
#include <png.h>
void png_convert_from_time_t(png_timep ptime, time_t ttime);
```

Description

This interface shall convert from the time_t time format to the png_time format.

The parameter *ttime* shall specify the time_t value to convert to png_time format.

The output parameter *ptime* shall contain the converted value.

png_create_info_struct

Name

png_create_info_struct — allocate and initialize a png_info structure

Synopsis

```
#include <png.h>
png_info* png_create_info_struct(png_structp png_ptr);
```

Description

png_create_info_struct() shall allocate and initialize a png_info structure.

Return Value

Returns the pointer to png_info structure. Returns NULL if it fails to create the structure.

Errors

png_create_info_struct() shall return NULL if the allocation fails. The application should check for the return value.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_create_read_struct

Name

`png_create_read_struct` — allocate and initialize a `png_struct` structure for reading PNG file

Synopsis

```
#include <png.h>
png_structp png_create_read_struct(png_const_charp user_png_ver,
png_voidp error_ptr, png_error_ptr error_fn, png_error_ptr warn_fn);
```

Description

`png_create_read_struct()` shall allocate and initialize a `png_struct` structure. The function shall return NULL if the allocation fails. The application should check for the return value. For handling errors and warnings, the application can pass desired error handling routines as arguments to `png_create_read_struct()`. Otherwise, the default error handling uses `stderr` and `longjmp`. The error handling routine must NOT return to the calling routine.

user_png_ver

version string of the library. Must be `PNG_LIBPNG_VER_STRING`

error_ptr

user defined struct for error functions.

error_fn

user defined function for printing errors and aborting.

warn_fn

user defined function for warnings.

Return Value

Returns the pointer to `png_struct` structure. Returns NULL if it fails to create the structure.

Errors

`png_create_read_struct()` shall return NULL if the allocation fails. The application should check for the return value.

png_create_read_struct_2

Name

png_create_read_struct_2 — register custom read function

Synopsis

```
#include <png.h>
png_structp png_create_read_struct_2(png_const_charp user_png_ver,
png_voidp error_ptr, png_error_ptr error_fn, png_error_ptr warn_fn,
png_voidp mem_ptr, png_malloc_ptr malloc_fn, png_free_ptr free_fn);
```

Description

This interface shall register a custom read function and allocate and initialize a `png_struct` structure. For handling errors and warnings, as well as allocating and deallocating memory, the application can pass functions as arguments. Otherwise, the default error handling functions `stderr` and `longjmp()` will be used. The error handling routine must not return to the calling routine.

The parameter `user_png_ver` shall specify the version string of the library, which must be `PNG_LIBPNG_VER_STRING`.

The parameter `error_ptr` shall specify a user-defined structure for error functions.

The parameter `error_fn` shall specify an optional user-defined function for printing errors and aborting.

The parameter `warn_fn` shall specify an optional user-defined function for warnings.

The parameter `mem_ptr` shall specify the memory to allocate.

The parameter `malloc_fn` shall specify an optional user-defined memory allocation function.

The parameter `free_fn` shall specify an optional user-defined memory deallocation function.

Application Usage (informative)

You should define `PNG_USER_MEM_SUPPORTED` before you call `png_create_read_struct2()`.

png_create_write_struct

Name

`png_create_write_struct` — allocate and initialize a `png_struct` structure for writing PNG file

Synopsis

```
#include <png.h>
png_structp png_create_write_struct(png_const_charp user_png_ver,
png_voidp error_ptr, png_error_ptr error_fn, png_error_ptr warn_fn);
```

Description

`png_create_write_struct()` shall allocate and initialize a `png_struct` structure. The function shall return NULL if the allocation fails. The application should check for the return value.

user_png_ver

version string of the library. Must be PNG_LIBPNG_VER_STRING

error_ptr

user defined struct for error functions.

error_fn

user defined function for printing errors and aborting.

warn_fn

user defined function for warnings.

Return Value

Returns the pointer to `png_struct` structure. Returns NULL if it fails to create the structure.

png_create_write_struct_2

Name

png_create_write_struct_2 — register custom write function

Synopsis

```
#include <png.h>
png_structp png_create_write_struct_2(png_const_charp user_png_ver,
png_voidp error_ptr, png_error_ptr error_fn, png_error_ptr warn_fn,
png_voidp mem_ptr, png_malloc_ptr malloc_fn, png_free_ptr free_fn);
```

Description

This interface shall register a custom write function and allocate and initialize a `png_struct` structure. For handling errors and warnings, as well as allocating and deallocating memory, the application can pass functions as arguments. Otherwise, the default error handling functions `stderr` and `longjmp()` will be used. The error handling routine must not return to the calling routine.

The parameter `user_png_ver` shall specify the version string of the library, which must be `PNG_LIBPNG_VER_STRING`.

The parameter `error_ptr` shall specify a user-defined structure for error functions.

The parameter `error_fn` shall specify an optional user-defined function for printing errors and aborting.

The parameter `warn_fn` shall specify an optional user-defined function for warnings.

The parameter `mem_ptr` shall specify the memory to allocate.

The parameter `malloc_fn` shall specify an optional user-defined memory allocation function.

The parameter `free_fn` shall specify an optional user-defined memory deallocation function.

Return Value

On success, returns a pointer to the `png_struct` that was created.

On failure, returns `NULL`.

png_data_freer

Name

`png_data_freer` — change the default behavior for freeing data

Synopsis

```
#include <png.h>
void png_data_freer(png_structp png_ptr, png_infop info_ptr, int freer,
png_uint_32 mask);
```

Description

This interface shall change the default behavior for freeing allocated data, from only freeing data allocated internally by libpng, to either freeing user-allocated data or to not freeing any data at all.

This interface can enable deallocation of user data allocated with `png_malloc()` or `png_zalloc()` and passed to libpng with one of the `png_set_*` interfaces. Users can call it both before and after passing the data. Calling it after reading PNG data but before passing the data controls whether the user or the interface is responsible for the existing data. Calling it after passing the data controls whether the user or the interface should deallocate the data.

If the user becomes responsible for data allocated by libpng, `png_free()` must be called to free it. If libpng becomes responsible for user-allocated data, the data must only have been allocated with `png_malloc()` or `png_zalloc()`.

The parameter `png_ptr` shall specify the PNG file.

The parameter `info_ptr` shall specify the PNG info structure.

The parameter `freer` shall specify one of these constants:
`PNG_DESTROY_WILL_FREE_DATA,` `PNG_SET_WILL_FREE_DATA,`
`PNG_USER_WILL_FREE_DATA.`

The parameter `mask` shall specify the data to free, as described under `png_free_data()`.

Application Usage (informative)

A `row_pointers` array allocated in a single block must not be freed with `png_set_rows()` or `png_read_destroy()`, because they would attempt to free the elements of the array as well.

Do not free `text_ptr` with libpng if some of its members were allocated separately, because this will actually only free `text_ptr.key`. If responsibility for freeing `text_ptr` moves from libpng to the user, the user must not free the members separately.

png_destroy_info_struct

Name

png_destroy_info_struct — free memory in PNG info structure

Synopsis

```
#include <png.h>
void png_destroy_info_struct(png_structp png_ptr, png_infopp
info_ptr_ptr);
```

Description

This interface shall free any memory in a single PNG info structure.

The parameter *png_ptr* shall specify the PNG file.

The parameter *info_ptr_ptr* shall point to a pointer to the png_info struct to destroy.

png_destroy_read_struct

Name

png_destroy_read_struct — free the memory associated with read png_struct

Synopsis

```
#include <png.h>
void png_destroy_read_struct(png_structpp png_ptr_ptr, png_infopp
info_ptr_ptr, png_infopp end_info_ptr_ptr);
```

Description

png_destroy_read_struct() frees the memory associated with the read png_struct struct that holds information from the given PNG file, the associated png_info struct for holding the image information and png_info struct for holding the information at end of the given PNG file.

png_destroy_write_struct

Name

png_destroy_write_struct — free the memory associated with write png_struct

Synopsis

```
#include <png.h>
void png_destroy_write_struct(png_structpp png_ptr_ptr, png_infopp
info_ptr_ptr);
```

Description

png_destroy_write_struct() frees the memory associated with the write png_struct struct that holds information for writing the PNG file and the associated png_info struct for holding the image information.

png_error**Name**

`png_error` — default function to handle fatal errors

Synopsis

```
#include <png.h>
void png_error(png_structp png_ptr, png_const_charp error_message);
```

Description

`png_error()` is the default error handling function for fatal errors. The default error handling functionality may be changed by using `png_set_error_fn()` to replace the error function at run-time.

Return Value**Errors****png_free****Name**

`png_free` — free a pointer allocated by `png_malloc()`

Synopsis

```
#include <png.h>
void png_free(png_structp png_ptr, png_voidp ptr);
```

Description

`png_free()` shall free memory pointed to by "ptr" previously allocated by `png_malloc()`.

png_free_data

Name

png_free_data — free internally allocated data

Synopsis

```
#include <png.h>
void png_free_data(png_structp png_ptr, png_infop info_ptr,
png_uint_32 free_me, int num);
```

Description

This interface shall free data that was internally allocated by libpng.

The parameter *png_ptr* shall specify the PNG file.

The parameter *info_ptr* shall specify the PNG info structure containing members to free that are pointing to memory allocated by libpng.

The parameter *mask* shall specify the data to free with a mask consisting of the logical OR of at least one of the following constants: PNG_FREE_HIST, PNG_FREE_ICCP, PNG_FREE_PCAL, PNG_FREE_PLTE, PNG_FREE_ROWS, PNG_FREE_SCAL, PNG_FREE_SPLT, PNG_FREE_TEXT, PNG_FREE_TRNS, PNG_FREE_UNKN. (To specify all constants, use PNG_FREE_ALL.)

The parameter *num* shall specify the sequence number of the item to free. The value -1 specifies all items.

png_get_IHDR

Name

png_get_IHDR — get PNG_IHDR chunk information from png_info structure

Synopsis

```
#include <png.h>
png_uint_32 png_get_IHDR(png_structp png_ptr, png_infop info_ptr,
png_uint_32 * width, png_uint_32 * height, int * bit_depth, int *
color_type, int * interlace_method, int * compression_method, int *
filter_method);
```

Description

png_get_IHDR() gets PNG_IHDR chunk type information from png_info structure.

width

holds the width of the image in pixels up to 2³¹.

height

holds the height of the image in pixels up to 2³¹.

bit_depth

holds the bit depth of one of the image channels. Valid values are 1, 2, 4, 8, 16 and also depend on the color_type.

color_type

describes which color/alpha channels are present. Supported color types shall include:

PNG_COLOR_TYPE_GRAY (bit depths 1, 2, 4, 8, 16)
 PNG_COLOR_TYPE_GRAY_ALPHA (bit depths 8, 16)
 PNG_COLOR_TYPE_PALETTE (bit depths 1, 2, 4, 8)
 PNG_COLOR_TYPE_RGB (bit depths 8, 16)
 PNG_COLOR_TYPE_RGB_ALPHA (bit depths 8, 16)
 PNG_COLOR_MASK_PALETTE
 PNG_COLOR_MASK_COLOR
 PNG_COLOR_MASK_ALPHA

filter_method

holds the filter method. If this argument is NULL, the filter method will not be retrieved. Valid values after retrieval are

PNG_FILTER_TYPE_BASE
 PNG_INTRAPIXEL_DIFFERENCING

compression_method

holds the compression method. If this argument is NULL, the compression method will not be retrieved. Valid values after retrieval are
 PNG_COMPRESSION_TYPE_BASE

interlace_method

holds the interlace method. If this argument is NULL, the interlacing method will not be retrieved. Valid values after retrieval are

PNG_INTERLACE_NONE
PNG_INTERLACE_ADAM7

Return Value

On success, `png_get_HDR()` shall return 1. Otherwise, `png_get_IHDR()` shall return 0.

png_get_PLTE

Name

`png_get_PLTE` — get image palette information from `png_info` structure

Synopsis

```
#include <png.h>
png_uint_32 png_get_PLTE(png_structp png_ptr, png_infop info_ptr,
png_colorp * palette, int * num_palette);
```

Description

`png_get_PLTE()` gets palette information from `png_info` structure. "palette" holds an array of color values with "num_palette" entries.

Return Value

On successful retrieval of palette information, `png_get_PLTE()` shall return `PNG_INFO_PLTE`. Otherwise, `png_get_PLTE()` shall return 0.

png_get_bKGD

Name

`png_get_bKGD` — get background color for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_bKGD(png_structp png_ptr, png_infop info_ptr,
png_color_16p * background);
```

Description

`png_get_bKGD()` shall return the background color to "background" if the validity flag for background is set.

Return Value

On success, `png_get_bKGD()` shall return `PNG_INFO_bKGD`. Otherwise, `png_get_bKGD()` shall return 0.

png_get_bit_depth

Name

png_get_bit_depth — return image bit_depth

Synopsis

```
#include <png.h>
png_byte png_get_bit_depth(png_structp png_ptr, png_infop info_ptr);
```

Description

Returns the image bit_depth.

Return Value

Returns 0 if png_ptr or info_ptr is NULL, bit_depth otherwise.

png_get_cHRM

Name

png_get_cHRM — get CIE chromacities and referenced white point for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_cHRM(png_structp png_ptr, png_infop info_ptr,
double * white_x, double * white_y, double * red_x, double * red_y,
double * green_x, double * green_y, double * blue_x, double * blue_y);
```

Description

png_get_cHRM() shall return the CIE x,y chromaticities of the red, green and blue display primaries used in the image and the referenced white point from the cHRM chunk in the image.

Return Value

On success, png_get_cHRM() shall return PNG_INFO_cHRM. Otherwise, the function shall return 0.

png_get_channels

Name

png_get_channels — get number of color channels in image

Synopsis

```
#include <png.h>
png_byte png_get_channels(png_structp png_ptr, png_infop info_ptr);
```

Description

png_get_channels() shall return the number of data channels per pixel for the color type of the image. The number of channels shall range from 1-4 depending on the color type as given below.

- 1 - PNG_COLOR_TYPE_GRAY or PNG_COLOR_TYPE_PALETTE
- 2 - PNG_COLOR_TYPE_GRAY_ALPHA
- 3 - PNG_COLOR_TYPE_RGB
- 4 - PNG_COLOR_TYPE_RGB_ALPHA or PNG_COLOR_TYPE_RGB+filler
byte

Return Value

On success, `png_get_channels()` shall return the number of channels ranging from 1-4. Otherwise, `png_get_channels` shall return 0.

png_get_color_type

Name

`png_get_color_type` — return image color type

Synopsis

```
#include <png.h>
png_byte png_get_color_type(png_structp png_ptr, png_info_ptr info_ptr);
```

Description

Returns the image color type.

Return Value

Returns 0 if `png_ptr` or `info_ptr` is NULL, `color_type` otherwise.

png_get_error_ptr

Name

`png_get_error_ptr` — return error_ptr for user-defined functions

Synopsis

```
#include <png.h>
png_voidp png_get_error_ptr(png_structp png_ptr);
```

Description

`png_get_error_ptr()` returns the `error_ptr` associated with user-defined functions.

Return Value

Returns `error_ptr`

png_get_gAMA

Name

png_get_gAMA — get the gamma value for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_gAMA(png_structp png_ptr, png_infop info_ptr,
double * file_gamma);
```

Description

Returns the gamma value of an image to "file_gamma" if the gAMA chunk information is valid for the image.

Return Value

PNG_INFO_gAMA - if png_ptr, info_ptr, file_gamma are not NULL and gAMA chunk information is valid 0 - otherwise.

png_get_hIST

Name

png_get_hIST — get the histogram for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_hIST(png_structp png_ptr, png_infop info_ptr,
png_uint_16p * hist);
```

Description

Returns the histogram of an image to *hist if the hIST chunk information is valid for the image.

Return Value

PNG_INFO_hIST - if png_ptr, info_ptr, hist are not NULL and hIST chunk information is valid 0 - otherwise.

png_get_header_ver

Name

png_get_header_ver — get version information for libpng header files

Synopsis

```
#include <png.h>
png_charp png_get_header_ver(png_structp png_ptr);
```

Description

This interface shall return the version of the header files used when building libpng as a short string in the format "1.0.0" through "99.99.99zz".

png_get_iCCP**Name**

png_get_iCCP — get the embedded ICC profile data for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_iCCP(png_structp png_ptr, png_infop info_ptr,
png_charpp name, int * compression_type, png_charpp profile,
png_uint_32 * proflen);
```

Description

png_get_iCCP() shall return the embedded ICC profile data in iCCP chunk. "name" shall contain the profile name, *compression_type shall contain the compression type, profile shall contain the International Color Consortium color profile data and *proflen shall contain the length of the profile data in bytes. *compression_type must always be set to PNG_COMPRESSION_TYPE_BASE.

Return Value

On success, png_get_iCCP() shall return PNG_INFO_iCCP. Otherwise, the function shall return 0.

png_get_image_height**Name**

png_get_image_height — return image height

Synopsis

```
#include <png.h>
png_uint_32 png_get_image_height(png_structp png_ptr, png_infop info_ptr);
```

Description

png_get_image_height() returns the image height in pixels.

Return Value

Returns 0 if png_ptr or info_ptr is NULL, image_height otherwise.

png_get_image_width

Name

png_get_image_width — return image width

Synopsis

```
#include <png.h>
png_uint_32 png_get_image_width(png_structp png_ptr, png_infop
info_ptr);
```

Description

png_get_image_width() returns the image width in pixels.

Return Value

Returns 0 if png_ptr or info_ptr is NULL, image_width otherwise.

png_get_interlace_type

Name

png_get_interlace_type — returns interlace method

Synopsis

```
#include <png.h>
png_byte png_get_interlace_type(png_structp png_ptr, png_infop
info_ptr);
```

Description

png_get_interlace_type() returns the interlace method used for the image.

Return Value

Valid values are PNG_INTERLACE_NONE, PNG_INTERLACE_ADAM7. Returns 0 if png_ptr or info_ptr is NULL.

png_get_io_ptr

Name

png_get_io_ptr — return pointer for user-defined I/O

Synopsis

```
#include <png.h>
png_voidp png_get_io_ptr(png_structp png_ptr);
```

Description

Returns the pointer associated with user-defined input-output functions.

png_get_libpng_ver**Name**

png_get_libpng_ver — get the library version string

Synopsis

```
#include <png.h>
png_charp png_get_libpng_ver(png_structp png_ptr);
```

Description

png_get_libpng_ver() shall return the library version as a short string in the format "1.0.0" through "99.99.99zz".

png_get_oFFs**Name**

png_get_oFFs — get screen offsets for the given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_oFFs(png_structp png_ptr, png_infop info_ptr,
png_int_32 * x_offset, png_int_32 * y_offset, int * unit_type);
```

Description

png_get_oFFs() shall read the positive offset from the left edge of the screen from info_ptr to x_offset and the positive offset from the top edge of the screen to y_offset. The unit type shall be returned in unit_type, which can take the following values

```
PNG_OFFSET_PIXEL
PNG_OFFSET_MICROMETER
```

Return Value

On success, png_get_oFFs() shall return PNG_INFO_oFFs. Otherwise, png_get_oFFs() shall return 0.

png_get_pHYs

Name

png_get_pHYs — get the physical resolution for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_pHYs(png_structp png_ptr, png_infop info_ptr,
png_uint_32 * res_x, png_uint_32 * res_y, int * unit_type);
```

Description

png_get_pHYs() shall return the physical pixel resolution of the image and the unit of resolution. Upon success, *res_x shall contain the horizontal resolution and *res_y shall contain the vertical resolution in pixels per unit. *unit_type will be set to PNG_RESOLUTION_METER if the resolution is expressed in pixels per meter. Otherwise *unit_type will be PNG_RESOLUTION_UNKNOWN.

Return Value

PNG_INFO_pHYs - on success 0 - otherwise.

png_get_progressive_ptr

Name

png_get_progressive_ptr — return pointer to user-defined push read functions

Synopsis

```
#include <png.h>
png_voidp png_get_progressive_ptr(png_structp png_ptr);
```

Description

Returns the pointer to user-defined structure containing information about the callback functions.

png_get_rowbytes

Name

png_get_rowbytes — Return number of bytes for a row

Synopsis

```
#include <png.h>
png_uint_32 png_get_rowbytes(png_structp png_ptr, png_infop info_ptr);
```

Description

png_get_rowbytes() returns the number of bytes needed to hold a transformed row of an image.

Return Value

Returns 0 if png_ptr or info_ptr is NULL, number of bytes otherwise.

png_get_rows**Name**

`png_get_rows` — retrieve image data from `png_info` structure

Synopsis

```
#include <png.h>
png_bytepp png_get_rows(png_structp png_ptr, png_infop info_ptr);
```

Description

`png_get_rows()` shall retrieve rows of image data from the `info_ptr` structure in an array of pointers to the pixel data for each row.

Return Value

On success, `png_get_rows()` shall return an array of pointers to the pixel data for each row of the image.

png_get_sBIT**Name**

`png_get_sBIT` — get number of significant bits for each color channel

Synopsis

```
#include <png.h>
png_uint_32 png_get_sBIT(png_structp png_ptr, png_infop info_ptr,
png_color_8p * sig_bit);
```

Description

`png_get_sBIT()` shall return the number of significant bit for each of the gray, red, blue and green color channels.

Return Value

On success, `png_get_sBIT()` shall return `PNG_INFO_sBIT`. Otherwise, `png_get_sBIT()` shall return 0.

png_get_sRGB

Name

png_get_sRGB — get the rendering intent for given image

Synopsis

```
#include <png.h>
png_uint_32 png_get_sRGB(png_structp png_ptr, png_infop info_ptr, int
* srgb_intent);
```

Description

png_get_sRGB() shall return the rendering intent of an image to *srgb_intent if the sRGB chunk information is valid for the image.

Return Value

PNG_INFO_sRGB - if png_ptr, info_ptr, srgb_intent are not NULL and sRGB chunk information is valid 0 - otherwise.

png_get_tIME

Name

png_get_tIME — get last modification time for the image

Synopsis

```
#include <png.h>
png_uint_32 png_get_tIME(png_structp png_ptr, png_infop info_ptr,
png_timep * mod_time);
```

Description

png_get_sBIT() shall return the time of last modification of the image if the tIME information is valid for the image.

Return Value

PNG_INFO_tIME - if png_ptr, info_ptr, mod_time are not NULL and tIME information is valid 0 - otherwise.

png_get_tRNS

Name

png_get_tRNS — get transparency data for images

Synopsis

```
#include <png.h>
png_uint_32 png_get_tRNS(png_structp png_ptr, png_info_ptr info_ptr,
    png_bytep * trans, int * num_trans, png_color_16p * trans_values);
```

Description

png_get_tRNS() shall obtain the transparency data for paletted images and image types that don't need a full alpha channel from info_ptr. For a paletted image, the function retrieves the transparency values stored in the same order as the palette colors, starting from index 0. For non-paletted images, the function retrieves the single color value which is treated as fully transparent. If the transparency information is valid, i.e. PNG_INFO_tRNS bit is set for info_ptr->valid: *trans shall be set to the transparency values for a paletted image. Values for the data shall be in range [0,255], ranging from fully transparent to fully opaque, respectively. *num_trans shall be set to the number of transparency values. *trans_values shall be set to the single color value specified for non-paletted images.

Return Value

Returns PNG_INFO_tRNS on successful return, 0 otherwise.

png_get_text

Name

png_get_text — get comments information from png_info structure

Synopsis

```
#include <png.h>
png_uint_32 png_get_text(png_structp png_ptr, png_info_ptr info_ptr,
    png_textp * text_ptr, int * num_text);
```

Description

png_get_text() returns the text chunk information from the PNG stream in the array pointed to by text_ptr. It also returns the number of text chunks in num_text. text_ptr is an array of structure png_text whose members include:

compression

type of compression used on text. Valid values are:

PNG_TEXT_COMPRESSION_NONE
PNG_TEXT_COMPRESSION_zTXt
PNG_ITXT_COMPRESSION_NONE
PNG_ITXT_COMPRESSION_zTXt

key

keyword for comment. Must contain 1-79 characters.

text

text comment for current keyword. May be empty.

text_length

length of text string after decompression. 0 for iTXt.

Return Value

Returns 0 if *png_ptr* or *info_ptr* is NULL, returns the number of text chunks otherwise.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_get_unknown_chunks

Name

`png_get_unknown_chunks` — retrieve the unknown chunks from a PNG file

Synopsis

```
#include <png.h>
png_uint_32 png_get_unknown_chunks(png_structp png_ptr, png_infop
info_ptr, png_unknown_chunkpp unknowns);
```

Description

This interface shall retrieve the unknown chunks from a PNG file.

The parameter `png_ptr` shall specify the PNG file.

The parameter `info_ptr` shall specify the PNG info structure.

The parameter `unknowns` shall specify an array of `png_unknown_chunk` structures containing the unknown chunks. The position of a structure in the array shall correspond to the order in which `png_set_unknown_chunks()` inserted its chunk into the PNG file, or in which it was read.

The `png_unknown_chunkpp` structure shall contain the following members.

name

the name of the chunk

data

the data of the chunk

size

the size of the chunk's data

location

the position of the chunk in the PNG file

Return Value

On success, this interface shall return the number of unknown chunks obtained.

On failure, this interface shall return 0.

png_get_user_chunk_ptr

Name

png_get_user_chunk_ptr — get pointer to user chunk data

Synopsis

```
#include <png.h>
png_voidp png_get_user_chunk_ptr(png_structp png_ptr);
```

Description

This interface shall return the pointer to the user chunk data associated with the specified PNG file.

The parameter *png_ptr* shall specify the PNG file.

png_get_valid

Name

png_get_valid — determine if given chunk data is valid

Synopsis

```
#include <png.h>
png_uint_32 png_get_valid(png_structp png_ptr, png_infop info_ptr,
png_uint_32 flag);
```

Description

png_get_valid() shall obtain the validity of chunk data specified by the bits set in "flag". The following bits may be set in flag: PNG_INFO_gAMA PNG_INFO_sBIT PNG_INFO_cHRM PNG_INFO_PLTE PNG_INFO_tRNS PNG_INFO_bKGD PNG_INFO_hIST PNG_INFO_pHYs PNG_INFO_oFFs PNG_INFO_tIME PNG_INFO_pCAL PNG_INFO_sRGB PNG_INFO_iCCP PNG_INFO_sPLT PNG_INFO_sCAL PNG_INFO_IDAT

Return Value

On success, png_get_valid() shall return "flag" with the chunk bits set. Otherwise, png_get_valid() shall return 0.

png_get_x_offset_pixels**Name**

`png_get_x_offset_pixels` — return x offset in pixels from oFFs chunk

Synopsis

```
#include <png.h>
png_int_32 png_get_x_offset_pixels(png_structp png_ptr, png_infop
info_ptr);
```

Description

`png_get_x_offset_pixels()` shall obtain x offset in pixels for the image from its PNG_oFFs chunk data stored in `info_ptr`, if the unit for offset is pixels.

Return Value

On success, `png_get_x_offset_pixels()` shall return x offset in pixels. Otherwise `png_get_x_offset_pixels()` shall return 0.

png_get_x_pixels_per_meter**Name**

`png_get_x_pixels_per_meter` — return horizontal pixel density per meter

Synopsis

```
#include <png.h>
png_uint_32 png_get_x_pixels_per_meter(png_structp png_ptr, png_infop
info_ptr);
```

Description

`png_get_x_pixels_per_meter()` shall obtain the horizontal pixel density in pixels per meter from its PNG_pHYs chunk data stored in `info_ptr`, if the unit for resolution is pixels per meter.

Return Value

On success, `png_get_x_pixels_per_meter()` shall return horizontal pixel density in pixels per meter. Otherwise `png_get_x_pixels_per_meter()` shall return 0.

png_get_y_offset_pixels**Name**

`png_get_y_offset_pixels` — return y offset in pixels from oFFs chunk

Synopsis

```
#include <png.h>
png_int_32 png_get_y_offset_pixels(png_structp png_ptr, png_infop
info_ptr);
```

Description

`png_get_y_offset_pixels()` shall obtain y offset in pixels for the image from its PNG_oFFs chunk data stored in `info_ptr`, if the unit for offset is pixels.

Return Value

On success, `png_get_y_offset_pixels()` shall return y offset in pixels. Otherwise `png_get_y_offset_pixels()` shall return 0.

png_get_y_pixels_per_meter**Name**

`png_get_y_pixels_per_meter` — return vertical pixel density per meter

Synopsis

```
#include <png.h>
png_uint_32 png_get_y_pixels_per_meter(png_structp png_ptr, png_infop
info_ptr);
```

Description

`png_get_y_pixels_per_meter()` shall obtain the vertical pixel density in pixels per meter from its PNG_pHYs chunk data stored in `info_ptr`, if the unit for resolution is pixels per meter.

Return Value

On success, `png_get_y_pixels_per_meter()` shall return vertical pixel density in pixels per meter. Otherwise `png_get_y_pixels_per_meter()` shall return 0.

png_info_init_3

Name

png_info_init_3 — initialize an info structure (DEPRECATED)

Synopsis

```
#include <png.h>
void      png_info_init_3(png_infopp      info_ptr,      png_size_t
png_info_struct_size);
```

Description

This interface shall initialize a PNG info structure. This interface is deprecated.

The parameter *info_ptr* shall specify the PNG info structure to initialize.

The parameter *png_info_struct_size* shall specify the size of the new structure.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_init_io

Name

png_init_io — initialize input/output for the PNG file

Synopsis

```
#include <png.h>
void png_init_io(png_structp png_ptr, png_FILE_p fp);
```

Description

Initialize the default input/output functions for the PNG file to standard C streams. To replace the default read and write functions, use png_set_read_fn() and png_set_write_fn() respectively.

Return Value

Errors

png_malloc

Name

png_malloc — allocate memory

Synopsis

```
#include <png.h>
png_voidp png_malloc(png_structp png_ptr, png_uint_32 size);
```

Description

png_malloc() shall return a pointer to allocated memory of the specified size.

Return Value

Pointer to the block of memory allocated.

Errors

Invokes error handling function if the system is out of memory and sets PNG_FLAG_MALLOC_NULL_MEM_OK in member flags of png_struct.

png_permit_mng_features

Name

`png_permit_mng_features` — enable MNG extensions for PNG image wrapped in MNG datastream

Synopsis

```
#include <png.h>
png_uint_32 png_permit_mng_features(png_structp png_ptr, png_uint_32
mng_features_permitted);
```

Description

This interface shall enable some MNG extensions for a PNG image wrapped in a MNG datastream.

The parameter `png_ptr` shall specify the PNG image.

The parameter `mng_features_permitted` shall specify the logical OR of the features to be enabled, possibly including `PNG_ALL_MNG_FEATURES`, `PNG_FLAG_MNG_EMPTY_PLTE`, and `PNG_FLAG_MNG_FILTER_64`.

Return Value

This interface shall return the logical AND of the parameter `mng_features_permitted` with the set of MNG features supported by the local version of libpng.

Application Usage (informational)

This interface may not read or write a standalone PNG file; the PNG datastream must be embedded in a MNG datastream with an 8-byte MNG signature and MEND and MHDR chunks.

png_process_data

Name

`png_process_data` — read PNG file progressively

Synopsis

```
#include <png.h>
void png_process_data(png_structp png_ptr, png_infop info_ptr,
png_bytep buffer, png_size_t length);
```

Description

`png_process_data()` shall process data progressively from the PNG stream using callback functions set within `png_set_progressive_read_fn()`. The data is passed in "buffer" and length of data to be processed is specified by "length".

png_progressive_combine_row

Name

png_progressive_combine_row — combines current row data with processed row

Synopsis

```
#include <png.h>
void png_progressive_combine_row(png_structp png_ptr, png_bytep old_row,
png_bytep new_row);
```

Description

For non-NULL rows of interlaced images during progressive read, png_progressive_combine_row() shall combine the data for the current row with the previously processed row data. png_progressive_combine_row() shall return for NULL rows of interlaced images and memcpy rows for non-interlaced images.

png_read_end

Name

png_read_end — read the end of PNG file

Synopsis

```
#include <png.h>
void png_read_end(png_structp png_ptr, png_info_ptr info_ptr);
```

Description

png_read_end() reads the end of a PNG file after reading the image data, including any comments or time information at the end of the file. The function shall not read past the end of the file.

png_read_image

Name

png_read_image — read the entire image into memory

Synopsis

```
#include <png.h>
void png_read_image(png_structp png_ptr, png_bytepp image);
```

Description

png_read_image() reads the entire image into memory at once. For each pass of an interlaced image, use png_read_rows() instead.

png_read_info

Name

png_read_info — read the PNG image information

Synopsis

```
#include <png.h>
void png_read_info(png_structp png_ptr, png_infop info_ptr);
```

Description

Reads the information before the actual image data from the PNG file. The function allows reading a file that already has the PNG signature bytes read from the stream.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_read_png

Name

png_read_png — read the entire PNG file

Synopsis

```
#include <png.h>
void png_read_png(png_structp png_ptr, png_infop info_ptr, int
transforms, png_voidp params);
```

Description

png_read_png() shall provide the high-level read operation. The function shall read the entire image into memory. The integer "transforms" shall contain the logical OR of a set of the following transformation flags:

PNG_TRANSFORM_IDENTITY

No transformation

PNG_TRANSFORM_STRIP_16

Strip 16-bit samples to 8 bits

PNG_TRANSFORM_STRIP_ALPHA

Discard the alpha channel

PNG_TRANSFORM_PACKING

Expand 1, 2 and 4-bit samples to bytes

PNG_TRANSFORM_PACKSWAP

Change order of packed pixels to LSB first

PNG_TRANSFORM_EXPAND

Expand paletted images to RGB, grayscale to 8-bit images and tRNS chunks to alpha channels

PNG_TRANSFORM_INVERT_MONO

Invert monochrome images

PNG_TRANSFORM_SHIFT

Normalize pixels to the sBIT depth

PNG_TRANSFORM_BGR

Flip RGB to BGR, RGBA to BGRA

PNG_TRANSFORM_SWAP_ALPHA

Flip RGBA to ARGB or GA to AG

PNG_TRANSFORM_INVERT_ALPHA

Change alpha from opacity to transparency

PNG_TRANSFORM_SWAP_ENDIAN

Byte-swap 16-bit samples

"params" is unused and must be set to NULL.

png_read_row

Name

png_read_row — read a row of image data

Synopsis

```
#include <png.h>
void png_read_row(png_structp png_ptr, png_bytep row, png_bytep
display_row);
```

Description

png_read_row() reads a row of actual image data. "row" holds the image pixels as they are processed. If the image is displayed after each pass, "display_row" is used to display a blurred progressive image. "display_row" can be NULL if the progressive image is not required.

png_read_rows

Name

png_read_rows — read multiple rows of image data

Synopsis

```
#include <png.h>
void png_read_rows(png_structp png_ptr, png_bytepp row, png_bytepp
display_row, png_uint_32 num_rows);
```

Description

Read "num_rows" rows of image data starting from "row". If the image is interlaced, the rows must contain the contents of the rows from the previous pass. If the image is displayed after each pass, "display_row" is used to display a blurred progressive image. "display_row" can be NULL if the progressive image is not required.

png_read_update_info

Name

png_read_update_info — update png_info structure

Synopsis

```
#include <png.h>
void png_read_update_info(png_structp png_ptr, png_infop info_ptr);
```

Description

png_read_update_info() updates the structure pointed to by info_ptr to reflect any transformations that have been requested. For example, rowbytes will be updated to handle expansion of an interlaced image with png_read_update_info().

png_set_IHDR

Name

png_set_IHDR — set the PNG_IHDR chunk information

Synopsis

```
#include <png.h>
void png_set_IHDR(png_structp png_ptr, png_info_ptr info_ptr, png_uint_32
width, png_uint_32 height, int bit_depth, int color_type, int
interlace_type, int compression_type, int filter_type);
```

Description

png_set_IHDR() shall set image header information in info_ptr. width is the image width in pixels. height is the image height in pixels. bit_depth is the bit depth of the image. Valid values shall include 1, 2, 4, 8, 16 and shall also depend on the color type. color_type is the type of image. Supported color types shall include: PNG_COLOR_TYPE_GRAY (bit depths 1, 2, 4, 8, 16) PNG_COLOR_TYPE_GRAY_ALPHA (bit depths 8, 16) PNG_COLOR_TYPE_PALETTE (bit depths 1, 2, 4, 8) PNG_COLOR_TYPE_RGB (bit depths 8, 16) PNG_COLOR_TYPE_RGB_ALPHA (bit depths 8, 16) PNG_COLOR_MASK_PALETTE PNG_COLOR_MASK_COLOR PNG_COLOR_MASK_ALPHA interlace_type is the image interlace method. Supported values shall include: PNG_INTERLACE_NONE or PNG_INTERLACE_ADAM7 compression_type is the method used for image compression. The value must be PNG_COMPRESSION_TYPE_DEFAULT. filter_type is the method used for image filtering. The value must be PNG_FILTER_TYPE_DEFAULT.

Errors

png_set_IHDR() shall invoke error function if any of the arguments has an invalid value.

png_set_PLTE

Name

png_set_PLTE — set color values for the palette

Synopsis

```
#include <png.h>
void png_set_PLTE(png_structp png_ptr, png_info_ptr info_ptr, png_colorp
palette, int num_palette);
```

Description

png_set_PLTE() shall set the array of color values used as palette for image to "palette". The palette shall include "num_palette" entries.

png_set_bKGD**Name**

png_set_bKGD — set the background color for given image

Synopsis

```
#include <png.h>
void png_set_bKGD(png_structp png_ptr, png_infop info_ptr,
png_color_16p background);
```

Description

png_set_bKGD() shall set the background color of an image to "background" and sets bKGD chunk information to valid for the image.

png_set_background**Name**

png_set_background — set the background for given image

Synopsis

```
#include <png.h>
void png_set_background(png_structp png_ptr, png_color_16p
background_color, int background_gamma_code, int need_expand, double
background_gamma);
```

Description

png_set_background() shall set the background of an image with alpha channel or simple transparency with the specified background color. If background_gamma_code is set to PNG_BACKGROUND_GAMMA_SCREEN, it indicates that the supplied background color is in the gamma space of the display, else if it is set to PNG_BACKGROUND_GAMMA_FILE, the color is in the gamma space of the file. If the background color is supplied at the original bit-depth for a grayscale image that is expanded to truecolor or to a higher bit-depth, need_expand must be set to 1, but if the background color is supplied at the expanded bit-depth, need_expand must be set to 0. Similarly for paletted images, if background color is supplied as a palette index, need_expand must be set to 1, else if background color is supplied as an RGB triplet, need_expand must be set to 0.

png_set_bgr**Name**

png_set_bgr — set pixel order to blue, green, red

Synopsis

```
#include <png.h>
void png_set_bgr(png_structp png_ptr);
```

Description

png_set_bgr() shall set the pixel order to blue, green, red.

png_set_cHRM

Name

`png_set_cHRM` — set CIE chromacities and referenced white point for given image

Synopsis

```
#include <png.h>
void png_set_cHRM(png_structp png_ptr, png_infop info_ptr, double
white_x, double white_y, double red_x, double red_y, double green_x,
double green_y, double blue_x, double blue_y);
```

Description

`png_set_cHRM()` shall set the CIE x,y chromaticities of the red, green and blue display primaries for the image and the referenced white point. The values must range from 0 to 21474.83 both inclusive.

Errors

`png_set_cHRM()` shall report a non-fatal error and exit if any of the chromacity values lies outside the range 0 to 21474.83.

png_set_compression_buffer_size

Name

`png_set_compression_buffer_size` — set the size of the compression buffer

Synopsis

```
#include <png.h>
void png_set_compression_buffer_size(png_structp png_ptr, png_uint_32
size);
```

Description

This interface shall set the size of the libz compression buffer `zbuf` for the specified PNG file.

The parameter `png_ptr` shall specify the PNG file for which to set the size of the compression buffer.

The parameter `size` shall specify the size to which to set the compression buffer, in bytes.

png_set_compression_level

Name

png_set_compression_level — set image compression level

Synopsis

```
#include <png.h>
void png_set_compression_level(png_structp png_ptr, int level);
```

Description

png_set_compression_level() shall set the compression level to "level". The valid values for "level" range from [0,9], corresponding directly to compression levels for zlib. The value 0 implies no compression and 9 implies maximal compression. Note: Tests have shown that zlib compression levels 3-6 usually perform as well as level 9 for PNG images, and do considerably fewer calculations.

png_set_compression_mem_level

Name

png_set_compression_mem_level — set how much memory to use for the internal state during PNG compression

Synopsis

```
#include <png.h>
void png_set_compression_mem_level(png_structp png_ptr, int mem_level);
```

Description

This interface shall set how much memory to use for the internal state during PNG compression.

The parameter *png_ptr* shall specify the PNG file to compress.

The parameter *mem_level* corresponds directly to the *memLevel* parameter of the `libz deflateInit2()` interface. This parameter shall specify how much memory to use for the internal state. The value of *mem_level* must be between 1 and `MAX_MEM_LEVEL`. Smaller values use less memory but are slower, while higher values use more memory to gain compression speed.

png_set_compression_method

Name

png_set_compression_method — set PNG compression algorithm

Synopsis

```
#include <png.h>
void png_set_compression_method(png_structp png_ptr, int method);
```

Description

This interface shall set the PNG compression algorithm to use.

The parameter *png_ptr* shall specify the PNG file to compress.

The parameter *method* corresponds directly to the *method* parameter of the libz `deflateInit2_()` interface. An LSB-conforming implementation shall support the `Z_DEFLATED` method, and may support other implementation defined methods.

png_set_compression_strategy

Name

png_set_compression_strategy — set PNG compression strategy

Synopsis

```
#include <png.h>
void png_set_compression_strategy(png_structp png_ptr, int strategy);
```

Description

This interface shall set the PNG compression strategy.

The parameter *png_ptr* shall specify the PNG file to compress.

The parameter *strategy* corresponds directly to the *strategy* parameter of the libz `deflateInit2_()` interface. This parameter shall specify the PNG compression strategy to use: one of `Z_DEFAULT_STRATEGY`, `Z_FILTERED`, and `Z_HUFFMAN_ONLY`.

png_set_compression_window_bits

Name

png_set_compression_window_bits — set PNG compression window size

Synopsis

```
#include <png.h>
void png_set_compression_window_bits(png_structp png_ptr, int window_bits);
```

Description

This interface shall set the PNG compression window size.

The parameter *window_bits* corresponds directly to the *windowBits* parameter of the libz `deflateInit2_()` interface. The value of this parameter equals the base 2 logarithm of the window size to use, and must be a value between 8 and 15.

png_set_error_fn

Name

png_set_error_fn — set user defined functions for error handling

Synopsis

```
#include <png.h>
void png_set_error_fn(png_structp png_ptr, png_voidp error_ptr,
png_error_ptr error_fn, png_error_ptr warning_fn);
```

Description

`png_set_error_fn()` shall replace the default error handling and warning functions with user defined function *error_fn* for handling fatal errors and function *warning_fn* for handling non-fatal errors. The replacement functions must do a `longjmp` to the last `setjmp` location if `setjmp/longjmp` method of error handling is used. If *error_fn* or *warning_fn* is NULL, the default functions for error handling shall be used.

png_set_expand

Name

png_set_expand — set expansion transformation

Synopsis

```
#include <png.h>
void png_set_expand(png_structp png_ptr);
```

Description

`png_set_expand()` shall set transformation in *png_ptr* such that paletted images are expanded to RGB, grayscale images of bit-depth less than 8 are expanded to 8-bit images and tRNS chunks are expanded to alpha channels.

png_set_filler

Name

png_set_filler — add a filler byte to given image

Synopsis

```
#include <png.h>
void png_set_filler(png_structp png_ptr, png_uint_32 filler, int flags);
```

Description

png_set_filler() shall set transformations in png_ptr such that a filler byte is added when an 8-bit grayscale image or 24-bit RGB image is read and a filler byte is deleted when an 8-bit grayscale image or 24-bit RGB image is written.

png_set_filter

Name

png_set_filter — set filtering method

Synopsis

```
#include <png.h>
void png_set_filter(png_structp png_ptr, int method, int filters);
```

Description

png_set_filter() shall set the filtering method used for scan-line filtering. The only valid value for "method" is 0. "filters" is a bitmap for which the following bits may be set. PNG_NO_FILTERS PNG_FILTER_NONE PNG_FILTER_SUB PNG_FILTER_UP PNG_FILTER_AVG PNG_FILTER_PAETH PNG_ALL_FILTERS

png_set_gAMA

Name

png_set_gAMA — set the gamma value for given image

Synopsis

```
#include <png.h>
void png_set_gAMA(png_structp png_ptr, png_info_ptr info_ptr, double file_gamma);
```

Description

Sets the gamma value of an image to "file_gamma" and sets gAMA chunk information to valid for the image.

Errors

png_set_gAMA() shall generate warning if file_gamma > 21474.83 or file_gamma = 0

png_set_gamma**Name**

png_set_gamma — transform the image from file gamma to screen gamma

Synopsis

```
#include <png.h>
void png_set_gamma(png_structp png_ptr, double screen_gamma, double
file_gamma);
```

Description

png_set_gamma() shall set the transformation for gamma correction of the PNG file based on the screen gamma i.e. the display exponent. The gamma transformation may be turned off later if no semitransparent entries are present in the tRNS array for palette images.

png_set_gray_to_rgb**Name**

png_set_gray_to_rgb — expand the grayscale image to 24-bit RGB

Synopsis

```
#include <png.h>
void png_set_gray_to_rgb(png_structp png_ptr);
```

Description

png_set_gray_to_rgb() shall set transformations such that the grayscale image is converted to 24-bit RGB.

png_set_hIST**Name**

png_set_hIST — set the histogram of color palette

Synopsis

```
#include <png.h>
void png_set_hIST(png_structp png_ptr, png_infop info_ptr,
png_uint_16p hist);
```

Description

png_set_hIST() shall set the histogram of palette to "hist".

png_set_iCCP

Name

png_set_iCCP — set ICC component

Synopsis

```
#include <png.h>
void png_set_iCCP(png_structp png_ptr, png_infop info_ptr, png_charp
name, int compression_type, png_charp profile, png_uint_32 proflen);
```

Description

png_set_iCCP() shall set the ICC component information to info_ptr. The arguments used to describe the ICC profile information have been described below:

name

ICC profile name

compression_type

compression type used must be 0

profile

profile data

proflen

length of profile data

png_set_interlace_handling

Name

png_set_interlace_handling — get the number of passes for image interlacing

Synopsis

```
#include <png.h>
int png_set_interlace_handling(png_structp png_ptr);
```

Description

png_set_interlace_handling() shall set the scheme to interlacing for writing an image and return the number of sub-images required to write the image.

Return Value

png_set_interlace_handling() shall return 7 if the image is interlaced, otherwise png_set_interlace_handling() shall return 1.

png_set_invert_alpha**Name**

`png_set_invert_alpha` — invert the level of opacity of a PNG file

Synopsis

```
#include <png.h>
void png_set_invert_alpha(png_structp png_ptr);
```

Description

This interface shall invert the level of opacity (alpha) of a PNG file.

The parameter *png_ptr* shall specify the PNG file for which to invert the opacity.

png_set_invert_mono**Name**

`png_set_invert_mono` — reverse values for monochromaticity

Synopsis

```
#include <png.h>
void png_set_invert_mono(png_structp png_ptr);
```

Description

`png_set_invert_mono()` shall set monochromaticity value 0 to white and value 1 to black.

png_set_keep_unknown_chunks

Name

`png_set_keep_unknown_chunks` — specify list of chunks and how to handle them

Synopsis

```
#include <png.h>
void png_set_keep_unknown_chunks(png_structp png_ptr, int keep,
png_bytep chunk_list, int num_chunks);
```

Description

This interface shall specify a list of chunks in the input PNG stream and how to handle them. Any unspecified chunks shall be handled in the default way. The IEND and IHDR chunks must not be specified.

The parameter `png_ptr` shall specify the PNG file.

The parameter `keep` shall specify how the unknown chunks are to be handled (see below).

The parameter `chunk_list` shall specify the list of chunks that shall be affected. The value passed must be a string of bytes with five bytes per chunk, or NULL or \0 if the value of `num_chunks` is 0.

The parameter `num_chunks` shall specify the number of chunks to be affected. If the value is 0, all unknown chunks shall be affected.

The possible values of `keep` are as follows.

- 0
handle unknown chunks in the default way
- 1
do not keep unknown chunks
- 2
keep unknown chunks only if they are safe to copy
- 3
keep unknown chunks even if they are unsafe to copy

Application Usage (informative)

The normal behavior of libpng is that known chunks are processed and unknown chunks are discarded. This interface reads both known and unknown chunks, handling them as specified by the user.

Unknown chunks specified to this interface are saved unchanged in a list of `png_unknown_chunk` structures. If a known chunk is specified in the list of unknown chunks, it will be handled per the `keep` parameter. If a chunk is specified in successive calls to this interface, the final call takes precedence.

png_set_mem_fn

Name

png_set_mem_fn — install custom memory allocation functions

Synopsis

```
#include <png.h>
void png_set_mem_fn(png_structp png_ptr, png_voidp mem_ptr,
png_malloc_ptr malloc_fn, png_free_ptr free_fn);
```

Description

This interface shall install custom memory allocation functions.

The parameter *png_ptr* shall specify the PNG file.

The parameter *mem_ptr* shall specify a struct provided by the user for memory functions.

The parameter *malloc_fn* shall specify the function used to allocate memory. If the value of this parameter is `NULL`, then the capability to allocate memory with the libpng ABI shall be disabled.

The parameter *free_fn* shall specify the function used to free memory. If the value of this parameter is `NULL`, then the capability to free memory with the libpng ABI shall be disabled.

png_set_oFFs

Name

png_set_oFFs — set screen offsets for given image

Synopsis

```
#include <png.h>
void png_set_oFFs(png_structp png_ptr, png_infop info_ptr, png_int_32
offset_x, png_int_32 offset_y, int unit_type);
```

Description

png_set_oFFs() shall set the positive offset from the left edge of the screen to offset_x and the positive offset from the left edge of the screen to offset_y. The unit_type must be `PNG_OFFSET_PIXEL` if the offset is defined in pixels or `PNG_OFFSET_MICROMETER` if the offset is defined in microns.

png_set_pHYs

Name

png_set_pHYs — set physical resolution

Synopsis

```
#include <png.h>
void png_set_pHYs(png_structp png_ptr, png_infop info_ptr, png_uint_32
res_x, png_uint_32 res_y, int unit_type);
```

Description

png_set_pHYs() sets the physical resolution for the image in pixels per unit. The physical resolution in x direction is set to res_x and that in y direction is set to res_y. unit_type must be set to PNG_RESOLUTION_METER is the unit for resolution is pixels per unit, otherwise unit_type must be set to PNG_RESOLUTION_UNKNOWN.

png_set_packing

Name

png_set_packing — expand image to 1 pixel per byte for bit-depths 1,2 and 4

Synopsis

```
#include <png.h>
void png_set_packing(png_structp png_ptr);
```

Description

png_set_packing() shall expand image to 1 pixel per byte for bit-depths 1, 2 and 4 without changing the order of the pixels. If png_set_packing() is not called, PNG files pack pixels of bit depths 1, 2 and 4 into bytes as small as possible, for example, 8 pixels per byte for 1-bit files.

png_set_packswap

Name

png_set_packswap — swap the order of pixels for packed-pixel image

Synopsis

```
#include <png.h>
void png_set_packswap(png_structp png_ptr);
```

Description

png_set_swap() shall change the pixel packing order for each byte for packed-pixel images with bit-depths 1, 2 or 4.

png_set_palette_to_rgb

Name

png_set_palette_to_rgb — set expansion transformation

Synopsis

```
#include <png.h>
void png_set_palette_to_rgb(png_structp png_ptr);
```

Description

png_set_palette_to_rgb() shall set transformation in *png_ptr* such that paletted images are expanded to RGB. png_set_palette_to_rgb() is actually an alias for png_set_expand().

png_set_progressive_read_fn

Name

png_set_progressive_read_fn — set progressive read callback functions

Synopsis

```
#include <png.h>
void png_set_progressive_read_fn(png_structp png_ptr, png_voidp
user_ptr, png_progressive_info_ptr info_callback,
png_progressive_row_ptr row_callback, png_progressive_end_ptr
end_callback);
```

Description

png_set_progressive_read_fn() shall provide function callbacks for which shall be called for processing image data by png_process_data(). "info_callback" shall be called to process header information, "row_callback" shall be called when each row is completed and "end_callback" shall be called to process end of image information. png_set_progressive_read_fn() must be called even if all callback functions are NULL. The user-defined structure pointed to by "user_ptr" may be retrieved from inside the callbacks using function get_progressive_ptr().

png_set_read_fn

Name

png_set_read_fn — set user-defined function for reading a PNG stream

Synopsis

```
#include <png.h>
void png_set_read_fn(png_structp png_ptr, png_voidp io_ptr, png_rw_ptr
read_data_fn);
```

Description

png_set_read_fn() sets the read_data_fn as the input function for reading PNG files instead of using standard C I/O stream functions. png_ptr - pointer to input data structure png_struct io_ptr - pointer to user-defined structure containing information about the input functions. This value may be NULL. read_data_fn - pointer to new input function that shall take the following arguments: - a pointer to a png_struct - a pointer to a structure where input data can be stored - 32-bit unsigned int to indicate number of bytes to read The input function should invoke png_error() to handle any fatal errors and png_warning() to handle non-fatal errors.

png_set_read_user_chunk_fn

Name

png_set_read_user_chunk_fn — install custom callback function to handle unknown chunks in the input stream

Synopsis

```
#include <png.h>
void png_set_read_user_chunk_fn(png_structp png_ptr, png_voidp
user_chunk_ptr, png_user_chunk_ptr read_user_chunk_fn);
```

Description

This interface shall install a custom callback function to handle unknown chunks in the input stream.

The parameter *png_ptr* shall specify the PNG file.

The parameter *user_chunk_ptr* shall specify a user pointer obtainable with *png_get_user_chunk_ptr()*.

The parameter *read_user_chunk_fn* shall specify the custom callback function.

png_set_read_user_transform_fn**Name**

`png_set_read_user_transform_fn` — install a custom input transformation callback function

Synopsis

```
#include <png.h>
void png_set_read_user_transform_fn(png_structp png_ptr,
png_user_transform_ptr read_user_transform_fn);
```

Description

This interface shall install a custom input transformation callback function.

The parameter `png_ptr` shall specify the PNG file to be transformed.

The parameter `read_user_transform_fn` shall specify the custom callback function.

png_set_rgb_to_gray**Name**

`png_set_rgb_to_gray` — reduce 24-bit RGB to grayscale image

Synopsis

```
#include <png.h>
void png_set_rgb_to_gray(png_structp png_ptr);
```

Description

`png_set_rgb_to_gray()` shall set transformations such that the 24-bit RGB image is converted to grayscale.

png_set_rows**Name**

`png_set_rows` — put image data in `png_info` structure

Synopsis

```
#include <png.h>
void png_set_rows(png_structp png_ptr, png_infop info_ptr, png_bytepp
row_pointers);
```

Description

`png_set_rows()` shall put rows of image data into the `info_ptr` structure, where `row_pointers` is an array of pointers to the pixel data for each row.

png_set_sBIT

Name

png_set_sBIT — set number of significant bits for each channel

Synopsis

```
#include <png.h>
void png_set_sBIT(png_structp png_ptr, png_infop info_ptr,
png_color_8p sig_bit);
```

Description

png_set_sBIT shall set the number of significant bits for each of gray, red, green and blue channels, whichever are appropriate for the given color type.

png_set_sRGB

Name

png_set_sRGB — set the rendering intent for given image

Synopsis

```
#include <png.h>
void png_set_sRGB(png_structp png_ptr, png_infop info_ptr, int
srgb_intent);
```

Description

png_set_sRGB() shall set the rendering intent of an image as specified by *srgb_intent* and shall set the sRGB chunk information to valid for the image. The presence of sRGB chunk implies that the pixel data is in the sRGB color space. *srgb_intent* can take one of the following values

```
PNG_sRGB_INTENT_SATURATION
PNG_sRGB_INTENT_PERCEPTUAL
PNG_sRGB_INTENT_ABSOLUTE
PNG_sRGB_INTENT_RELATIVE
```

png_set_sRGB_gAMA_and_cHRM**Name**

`png_set_sRGB_gAMA_and_cHRM` — set rendering intent, gamma values, and CIE chromaticities of a PNG file

Synopsis

```
#include <png.h>
void png_set_sRGB_gAMA_and_cHRM(png_structp png_ptr, png_infop info_ptr,
int srgb_intent);
```

Description

This interface shall set the rendering intent, gamma values, and CIE chromaticities of a PNG file.

The parameter *png_ptr* shall specify the PNG file.

The parameter *info_ptr* shall specify the PNG info structure.

The parameter *srgb_intent* shall specify the rendering intent. Because the sRGB chunk is present, the pixel data uses the sRGB color space. This interface shall also write gAMA and cHRM chunks with values consistent with sRGB.

png_set_shift**Name**

`png_set_shift` — shift pixel values to valid bit-depth

Synopsis

```
#include <png.h>
void png_set_shift(png_structp png_ptr, png_color_8p true_bits);
```

Description

If image data in a row buffer is stored in a bit depth other than those supported by PNG, `png_set_shift()` shall scale the values to a valid bit-depth defined by PNG format. For example, 3-bit data in range 0-7 is scaled to 4-bit PNG.

png_set_sig_bytes**Name**

`png_set_sig_bytes` — number of bytes read from PNG file

Synopsis

```
#include <png.h>
void png_set_sig_bytes(png_structp png_ptr, int num_bytes);
```

Description

`png_set_sig_bytes()` shall store the number of bytes of the PNG file signature that have been read from the PNG stream.

Errors

`png_set_sig_bytes()` shall invoke error function if `num_bytes > 8`.

png_set_strip_16**Name**

png_set_strip_16 — strip 16 bit PNG file to 8 bit depth

Synopsis

```
#include <png.h>
void png_set_strip_16(png_structp png_ptr);
```

Description

png_set_strip_16() shall strip the pixels of a PNG stream with 16 bits per channel to 8 bits per channel.

png_set_strip_alpha**Name**

png_set_strip_alpha — remove alpha channel on the given image

Synopsis

```
#include <png.h>
void png_set_strip_alpha(png_structp png_ptr);
```

Description

png_set_strip_alpha() shall set transformation on the image to remove the alpha channel.

png_set_swap**Name**

png_set_swap — swap byte-order for 16 bit depth files

Synopsis

```
#include <png.h>
void png_set_swap(png_structp png_ptr);
```

Description

PNG files store 16-bit pixels in network byte order (big-endian, ie most significant bytes first). png_set_swap() shall switch the byte-order to little-endian (ie, least significant bits first).

png_set_swap_alpha**Name**

png_set_swap_alpha — swap image data from RGBA to ARGB format

Synopsis

```
#include <png.h>
void png_set_swap_alpha(png_structp png_ptr);
```

Description

png_set_swap_alpha() shall swap data for an image with an alpha channel from RGBA format to ARGB format.

png_set_tIME**Name**

png_set_tIME — set last modification time for the image

Synopsis

```
#include <png.h>
void png_set_tIME(png_structp png_ptr, png_infop info_ptr, png_timep
mod_time);
```

Description

png_set_sBIT shall set the time of last modification of the image in info_ptr as specified by mod_time.

png_set_tRNS**Name**

png_set_tRNS — set transparency values for images

Synopsis

```
#include <png.h>
void png_set_tRNS(png_structp png_ptr, png_infop info_ptr, png_bytep
trans, int num_trans, png_color_16p trans_values);
```

Description

png_set_tRNS() shall set the transparency data for paletted images and image types that don't need a full alpha channel. For a paletted image, png_set_tRNS() shall set the array of transparency values for the palette colors to "trans". The number of transparency entries is given by "num_trans". For non-paletted images, png_set_tRNS() shall set the single color value or graylevel to "trans_values"

png_set_tRNS_to_alpha

Name

png_set_tRNS_to_alpha — set expansion transformation

Synopsis

```
#include <png.h>
void png_set_tRNS_to_alpha(png_structp png_ptr);
```

Description

png_set_tRNS_to_alpha() shall set transformation in *png_ptr* such that tRNS chunks are expanded to alpha channels. png_set_tRNS_to_alpha() is actually an alias for png_set_expand().

png_set_text

Name

png_set_text — stores information for image comments

Synopsis

```
#include <png.h>
void png_set_text(png_structp png_ptr, png_info_ptr info_ptr, png_textp
text_ptr, int num_text);
```

Description

png_set_text() shall store information for image comments given in *text_ptr* to *info_ptr*. *text_ptr* is an array of size *num_text* of png_text structures whose member fields include:

compression

type of compression used on text. Valid values are:

```
PNG_TEXT_COMPRESSION_NONE
PNG_TEXT_COMPRESSION_zTXt
PNG_ITXT_COMPRESSION_NONE
PNG_ITXT_COMPRESSION_zTXt
```

key

keyword for comment. Must contain 1-79 characters.

text

text comment for current keyword. May be empty.

text_length

length of text string after decompression. 0 for iTXt.

png_set_unknown_chunk_location

Name

`png_set_unknown_chunk_location` — set the location of an unknown chunk in a PNG file

Synopsis

```
#include <png.h>
void png_set_unknown_chunk_location(png_structp png_ptr, png_info_ptr info_ptr, int chunk, int location);
```

Description

This interface shall set the location of an unknown chunk in a PNG file.

The parameter *png_ptr* shall specify the PNG file.

The parameter *info_ptr* shall specify the PNG info structure.

The parameter *chunk* shall specify the number of the chunk in the array of unknown chunks.

The parameter *location* shall specify the new location of the chunk within the PNG file.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_set_unknown_chunks

Name

`png_set_unknown_chunks` — insert unknown chunks into a PNG file

Synopsis

```
#include <png.h>
void png_set_unknown_chunks(png_structp png_ptr, png_infop info_ptr,
    png_unknown_chunkp unknowns, int num_unknowns);
```

Description

This interface shall insert unknown chunks into a PNG file.

The parameter `png_ptr` shall specify the PNG file.

The parameter `info_ptr` shall specify the PNG info structure.

The parameter `unknowns` shall specify an array of `png_unknown_chunk` structures containing the unknown chunks, as described under `png_get_unknown_chunks()`. The `location` member of a `png_unknown_chunk` structure can take several special values (see below).

The parameter `num_unknowns` shall specify the number of unknown chunks.

The special values for the `location` members of the `png_unknown_chunk` structures are as follows.

0

do not write the chunk

`PNG_HAVE_IHDR`

insert chunk before PLTE

`PNG_HAVE_PLTE`

insert chunk before IDAT

`PNG_AFTER_IDAT`

insert chunk after IDAT

Notes

The `location` member of the `png_unknown_chunk` structure is set automatically depending on how much of the PNG file has been written. Its value can be changed after calling this interface. The chunk is placed within a location according to its position in the array of structures, as described under `png_get_unknown_chunks()`.

png_set_write_fn

Name

png_set_write_fn — set user-defined function for writing a PNG stream

Synopsis

```
#include <png.h>
void png_set_write_fn(png_structp png_ptr, png_voidp io_ptr,
png_rw_ptr write_data_fn, png_flush_ptr output_flush_fn);
```

Description

png_set_write_fn() sets the write_data_fn as the output function for writing PNG files instead of using standard C I/O stream functions. png_ptr - pointer to output data structure png_struct io_ptr - pointer to user-defined structure containing information about the output functions. This value may be NULL. write_data_fn - pointer to new output function that shall take the following arguments: - a pointer to a png_struct - a pointer to a structure where output data can be stored - 32-bit unsigned int to indicate number of bytes to write The output function should invoke png_error() to handle any fatal errors and png_warning() to handle non-fatal errors. flush_data_fn - pointer to a new flush function that shall take a pointer to a png_struct as argument. This function shall flush any remaining data in buffers used by the output function. If the output function does not buffer output, a function prototype must still be supplied.

png_set_write_status_fn

Name

png_set_write_status_fn — install custom callback function to be called after row is written

Synopsis

```
#include <png.h>
void png_set_write_status_fn(png_structp png_ptr,
png_write_status_ptr write_row_fn);
```

Description

This interface shall install a custom callback function to be called after a row has been written.

The parameter *png_ptr* shall specify the PNG file to be transformed.

The parameter *write_row_fn* shall specify the custom callback function.

png_set_write_user_transform_fn

Name

png_set_write_user_transform_fn — install a custom output transformation callback function

Synopsis

```
#include <png.h>
void      png_set_write_user_transform_fn(png_structp      png_ptr,
png_user_transform_ptr write_user_transform_fn);
```

Description

This interface shall install a custom output transformation callback function.

The parameter *png_ptr* shall specify the PNG file to be transformed.

The parameter *write_user_transform_fn* shall specify the custom callback function.

png_sig_cmp

Name

png_sig_cmp — match the PNG signature

Synopsis

```
#include <png.h>
int      png_sig_cmp(png_bytep      sig,      png_size_t      start,      png_size_t
num_to_check);
```

Description

png_sig_cmp() checks whether the given number of bytes match the PNG signature starting from the start position. The function shall return non-zero if num_to_check == 0 or start > 7.

Return Value

Zero - the given number of bytes starting from start position match the respective bytes of the PNG signature. Non-zero - the given number of bytes starting from start position do not match the respective bytes of the PNG signature or num_to_check == 0 or start > 7.

png_start_read_image**Name**

png_start_read_image — start reading a PNG file

Synopsis

```
#include <png.h>
void png_start_read_image(png_structp png_ptr);
```

Description

This interface shall update the palette with the previously specified transformations, and then start reading the specified PNG file.

The parameter *png_ptr* shall specify the PNG file to read.

png_warning**Name**

png_warning — default function to handle non-fatal errors

Synopsis

```
#include <png.h>
void png_warning(png_structp png_ptr, png_const_charp warning_message);
```

Description

png_warning() is the default function for handling non-fatal errors. The default function to handle warnings may be changed by using png_set_error_fn() to replace the warning function at run-time.

png_write_chunk**Name**

png_write_chunk — write a PNG chunk

Synopsis

```
#include <png.h>
void png_write_chunk(png_structp png_ptr, png_bytep chunk_name,
png_bytep data, png_size_t length);
```

Description

png_write_chunk() writes the start of a PNG chunk, the chunk data and the end of the chunk all at once.

png_write_end**Name**

png_write_end — write the end of a PNG file

Synopsis

```
#include <png.h>
void png_write_end(png_structp png_ptr, png_infop info_ptr);
```

Description

png_write_end() writes the end of a PNG file to which the image data has already been written. The user may write time information or comments at the end of the PNG file.

png_write_flush**Name**

png_write_flush — flush the current output buffers

Synopsis

```
#include <png.h>
void png_write_flush(png_structp png_ptr);
```

Description

png_write_flush() shall the current output buffers for any pending data.

png_write_image**Name**

png_write_image — write the given image data

Synopsis

```
#include <png.h>
void png_write_image(png_structp png_ptr, png_bytepp image);
```

Description

Write the rows of given image data. If the image is not interlaced, the image shall be written in a single pass.

png_write_info**Name**

png_write_info — write PNG information to file

Synopsis

```
#include <png.h>
void png_write_info(png_structp png_ptr, png_infop info_ptr);
```

Description

png_write_info() writes the PNG information in info_ptr to file.

png_write_info_before_PLTE**Name**

png_write_info_before_PLTE — INSERT PURPOSE HERE

Synopsis

```
#include <png.h>
void png_write_info_before_PLTE(png_structp arg0, png_infop arg1);
```

Description

INSERT TEXT HERE

Return Value

INSERT TEXT HERE

Errors

INSERT TEXT HERE

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

png_write_png

Name

png_write_png — write the entire PNG file

Synopsis

```
#include <png.h>
void png_write_png(png_structp png_ptr, png_infop info_ptr, int
transforms, png_voidp params);
```

Description

png_write_png() shall provide the high-level write operation. The function shall write the PNG stream if the entire image information is available in png_ptr. The integer "transforms" shall contain the logical OR of a set of the following transformation flags:

PNG_TRANSFORM_IDENTITY

No transformation

PNG_TRANSFORM_PACKING

Expand 1, 2 and 4-bit samples to bytes

PNG_TRANSFORM_PACKSWAP

Change order of packed pixels to LSB first

PNG_TRANSFORM_INVERT_MONO

Invert monochrome images

PNG_TRANSFORM_SHIFT

Normalize pixels to the sBIT depth

PNG_TRANSFORM_BGR

Flip RGB to BGR, RGBA to BGRA

PNG_TRANSFORM_SWAP_ALPHA

Flip RGBA to ARGB or GA to AG

PNG_TRANSFORM_INVERT_ALPHA

Change alpha from opacity to transparency

PNG_TRANSFORM_SWAP_ENDIAN

Byte-swap 16-bit samples

PNG_TRANSFORM_STRIP_FILLER

Strip off filler bytes

"params" is unused and must be set to NULL.

png_write_row**Name**

png_write_row — write a row of image data

Synopsis

```
#include <png.h>
void png_write_row(png_structp png_ptr, png_bytep row);
```

Description

Process and write a row of image data. The header information must have been written before the image data can be written.

png_write_rows**Name**

png_write_rows — write multiple rows of image data

Synopsis

```
#include <png.h>
void png_write_rows(png_structp png_ptr, png_bytepp row, png_uint_32
num_rows);
```

Description

Process and write "num_rows" rows of image data starting from "row".

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

III GTK+ Stack Libraries

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC TS 23360-1-6:2021

7 Libraries

7.1 Introduction

A conforming implementation shall support the following libraries from the GTK+ 3 stack which provide interfaces for creating rich graphical user interface applications.

GDK

GDK is the abstraction layer that allows GTK+ to support multiple windowing systems. GDK provides drawing and window system facilities on X11, Windows, and the Linux framebuffer device.

libgdk-3

GTK+

The GTK+ library contains widgets, that is, GUI components such as GtkButton or GtkTextView.

libgtk-3

There are three main parts to the definition of each of these libraries.

The "Interfaces" section defines the required library name and version, and the required public symbols (interfaces and global data), as well as symbol versions, if any.

The "Interface Definitions" section provides complete or partial definitions of certain interfaces where either this specification is the source specification, or where there are variations from the source specification. If an interface definition requires one or more header files, one of those headers shall include the function prototype for the interface.

For source definitions of interfaces which include a reference to a header file, the contents of such header files form a part of the specification. The "Data Definitions" section provides the binary-level details for the header files from the source specifications, such as values for macros and enumerated types, as well as structure layouts, sizes and padding, etc. These data definitions, although presented in the form of header files for convenience, should not be taken as representing complete header files, as they are a supplement to the source specifications. Application developers should follow the guidelines of the source specifications when determining which header files need to be included to completely resolve all references.

Note: While the Data Definitions supplement the source specifications, this specification itself does not require conforming implementations to supply any header files.

7.2 Interfaces for libgdk-3

Table 7-1 defines the library name and shared object name for the libgdk-3 library

Table 7-1 libgdk-3 Definition

Library:	libgdk-3
SONAME:	libgdk-3.so.0

The behavior of the interfaces in this library is specified by the following specifications:

[Gdk3 3.6] Gdk 3.6.4 Reference Manual

7.2.1 libgdk-3 interfaces

7.2.1.1 Interfaces for libgdk-3 interfaces

An LSB conforming implementation shall provide the generic functions for libgdk-3 interfaces specified in Table 7-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 7-2 libgdk-3 - libgdk-3 interfaces Function Interfaces

gdk_app_launch_context_new [Gdk3 3.6]	gdk_app_launch_context_set_desktop [Gdk3 3.6]
gdk_app_launch_context_set_display [Gdk3 3.6]	gdk_app_launch_context_set_icon [Gdk3 3.6]
gdk_app_launch_context_set_icon_name [Gdk3 3.6]	gdk_app_launch_context_set_screen [Gdk3 3.6]
gdk_app_launch_context_set_timestamp [Gdk3 3.6]	gdk_atom_intern [Gdk3 3.6]
gdk_atom_intern_static_string [Gdk3 3.6]	gdk_atom_name [Gdk3 3.6]
gdk_beep [Gdk3 3.6]	gdk_cairo_create [Gdk3 3.6]
gdk_cairo_get_clip_rectangle [Gdk3 3.6]	gdk_cairo_rectangle [Gdk3 3.6]
gdk_cairo_region [Gdk3 3.6]	gdk_cairo_region_create_from_surface [Gdk3 3.6]
gdk_cairo_set_source_color [Gdk3 3.6]	gdk_cairo_set_source_pixbuf [Gdk3 3.6]
gdk_cairo_set_source_rgba [Gdk3 3.6]	gdk_cairo_set_source_window [Gdk3 3.6]
gdk_color_copy [Gdk3 3.6]	gdk_color_equal [Gdk3 3.6]
gdk_color_free [Gdk3 3.6]	gdk_color_hash [Gdk3 3.6]
gdk_color_parse [Gdk3 3.6]	gdk_color_to_string [Gdk3 3.6]
gdk_cursor_get_cursor_type [Gdk3 3.6]	gdk_cursor_get_display [Gdk3 3.6]
gdk_cursor_get_image [Gdk3 3.6]	gdk_cursor_new [Gdk3 3.6]
gdk_cursor_new_for_display [Gdk3 3.6]	gdk_cursor_new_from_name [Gdk3 3.6]
gdk_cursor_new_from_pixbuf [Gdk3 3.6]	gdk_cursor_ref [Gdk3 3.6]
gdk_cursor_unref [Gdk3 3.6]	gdk_device_free_history [Gdk3 3.6]

gdk_device_get_associated_device [Gdk3 3.6]	gdk_device_get_axis [Gdk3 3.6]
gdk_device_get_axis_use [Gdk3 3.6]	gdk_device_get_axis_value [Gdk3 3.6]
gdk_device_get_device_type [Gdk3 3.6]	gdk_device_get_display [Gdk3 3.6]
gdk_device_get_has_cursor [Gdk3 3.6]	gdk_device_get_history [Gdk3 3.6]
gdk_device_get_key [Gdk3 3.6]	gdk_device_get_mode [Gdk3 3.6]
gdk_device_get_n_axes [Gdk3 3.6]	gdk_device_get_n_keys [Gdk3 3.6]
gdk_device_get_name [Gdk3 3.6]	gdk_device_get_position [Gdk3 3.6]
gdk_device_get_source [Gdk3 3.6]	gdk_device_get_state [Gdk3 3.6]
gdk_device_get_window_at_position [Gdk3 3.6]	gdk_device_grab [Gdk3 3.6]
gdk_device_list_axes [Gdk3 3.6]	gdk_device_list_slave_devices [Gdk3 3.6]
gdk_device_manager_get_client_pointer [Gdk3 3.6]	gdk_device_manager_get_display [Gdk3 3.6]
gdk_device_manager_list_devices [Gdk3 3.6]	gdk_device_set_axis_use [Gdk3 3.6]
gdk_device_set_key [Gdk3 3.6]	gdk_device_set_mode [Gdk3 3.6]
gdk_device_ungrab [Gdk3 3.6]	gdk_device_warp [Gdk3 3.6]
gdk_disable_multidevice [Gdk3 3.6]	gdk_display_beep [Gdk3 3.6]
gdk_display_close [Gdk3 3.6]	gdk_display_device_is_grabbed [Gdk3 3.6]
gdk_display_flush [Gdk3 3.6]	gdk_display_get_app_launch_context [Gdk3 3.6]
gdk_display_get_default [Gdk3 3.6]	gdk_display_get_default_cursor_size [Gdk3 3.6]
gdk_display_get_default_group [Gdk3 3.6]	gdk_display_get_default_screen [Gdk3 3.6]
gdk_display_get_device_manager [Gdk3 3.6]	gdk_display_get_event [Gdk3 3.6]
gdk_display_get_maximal_cursor_size [Gdk3 3.6]	gdk_display_get_n_screens [Gdk3 3.6]
gdk_display_get_name [Gdk3 3.6]	gdk_display_get_pointer [Gdk3 3.6]
gdk_display_get_screen [Gdk3 3.6]	gdk_display_get_window_at_pointer [Gdk3 3.6]
gdk_display_has_pending [Gdk3 3.6]	gdk_display_is_closed [Gdk3 3.6]

gdk_display_keyboard_ungrab [Gdk3 3.6]	gdk_display_list_devices [Gdk3 3.6]
gdk_display_manager_get [Gdk3 3.6]	gdk_display_manager_get_default_display [Gdk3 3.6]
gdk_display_manager_list_displays [Gdk3 3.6]	gdk_display_manager_open_display [Gdk3 3.6]
gdk_display_manager_set_default_display [Gdk3 3.6]	gdk_display_notify_startup_complete [Gdk3 3.6]
gdk_display_open [Gdk3 3.6]	gdk_display_peek_event [Gdk3 3.6]
gdk_display_pointer_is_grabbed [Gdk3 3.6]	gdk_display_pointer_ungrab [Gdk3 3.6]
gdk_display_put_event [Gdk3 3.6]	gdk_display_request_selection_notification [Gdk3 3.6]
gdk_display_set_double_click_distance [Gdk3 3.6]	gdk_display_set_double_click_time [Gdk3 3.6]
gdk_display_store_clipboard [Gdk3 3.6]	gdk_display_supports_clipboard_persistence [Gdk3 3.6]
gdk_display_supports_composite [Gdk3 3.6]	gdk_display_supports_cursor_alpha [Gdk3 3.6]
gdk_display_supports_cursor_color [Gdk3 3.6]	gdk_display_supports_input_shapes [Gdk3 3.6]
gdk_display_supports_selection_notification [Gdk3 3.6]	gdk_display_supports_shapes [Gdk3 3.6]
gdk_display_sync [Gdk3 3.6]	gdk_display_warp_pointer [Gdk3 3.6]
gdk_drag_abort [Gdk3 3.6]	gdk_drag_begin [Gdk3 3.6]
gdk_drag_begin_for_device [Gdk3 3.6]	gdk_drag_context_get_actions [Gdk3 3.6]
gdk_drag_context_get_dest_window [Gdk3 3.6]	gdk_drag_context_get_device [Gdk3 3.6]
gdk_drag_context_get_protocol [Gdk3 3.6]	gdk_drag_context_get_selected_action [Gdk3 3.6]
gdk_drag_context_get_source_window [Gdk3 3.6]	gdk_drag_context_get_suggested_action [Gdk3 3.6]
gdk_drag_context_list_targets [Gdk3 3.6]	gdk_drag_context_set_device [Gdk3 3.6]
gdk_drag_drop [Gdk3 3.6]	gdk_drag_drop_succeeded [Gdk3 3.6]
gdk_drag_find_window_for_screen [Gdk3 3.6]	gdk_drag_get_selection [Gdk3 3.6]
gdk_drag_motion [Gdk3 3.6]	gdk_drag_status [Gdk3 3.6]

gdk_drop_finish [Gdk3 3.6]	gdk_drop_reply [Gdk3 3.6]
gdk_error_trap_pop [Gdk3 3.6]	gdk_error_trap_pop_ignored [Gdk3 3.6]
gdk_error_trap_push [Gdk3 3.6]	gdk_event_copy [Gdk3 3.6]
gdk_event_free [Gdk3 3.6]	gdk_event_get [Gdk3 3.6]
gdk_event_get_axis [Gdk3 3.6]	gdk_event_get_button [Gdk3 3.6]
gdk_event_get_click_count [Gdk3 3.6]	gdk_event_get_coords [Gdk3 3.6]
gdk_event_get_device [Gdk3 3.6]	gdk_event_get_event_sequence [Gdk3 3.6]
gdk_event_get_keycode [Gdk3 3.6]	gdk_event_get_keyval [Gdk3 3.6]
gdk_event_get_root_coords [Gdk3 3.6]	gdk_event_get_screen [Gdk3 3.6]
gdk_event_get_scroll_deltas [Gdk3 3.6]	gdk_event_get_scroll_direction [Gdk3 3.6]
gdk_event_get_source_device [Gdk3 3.6]	gdk_event_get_state [Gdk3 3.6]
gdk_event_get_time [Gdk3 3.6]	gdk_event_handler_set [Gdk3 3.6]
gdk_event_new [Gdk3 3.6]	gdk_event_peek [Gdk3 3.6]
gdk_event_put [Gdk3 3.6]	gdk_event_request_motions [Gdk3 3.6]
gdk_event_set_device [Gdk3 3.6]	gdk_event_set_screen [Gdk3 3.6]
gdk_event_set_source_device [Gdk3 3.6]	gdk_event_triggers_context_menu [Gdk3 3.6]
gdk_events_get_angle [Gdk3 3.6]	gdk_events_get_center [Gdk3 3.6]
gdk_events_get_distance [Gdk3 3.6]	gdk_events_pending [Gdk3 3.6]
gdk_flush [Gdk3 3.6]	gdk_get_default_root_window [Gdk3 3.6]
gdk_get_display [Gdk3 3.6]	gdk_get_display_arg_name [Gdk3 3.6]
gdk_get_program_class [Gdk3 3.6]	gdk_get_show_events [Gdk3 3.6]
gdk_init [Gdk3 3.6]	gdk_init_check [Gdk3 3.6]
gdk_keyboard_grab [Gdk3 3.6]	gdk_keyboard_ungrab [Gdk3 3.6]
gdk_keymap_add_virtual_modifiers [Gdk3 3.6]	gdk_keymap_get_caps_lock_state [Gdk3 3.6]
gdk_keymap_get_default [Gdk3 3.6]	gdk_keymap_get_direction [Gdk3 3.6]
gdk_keymap_get_entries_for_keycode [Gdk3 3.6]	gdk_keymap_get_entries_for_keyval [Gdk3 3.6]

gdk_keymap_get_for_display [Gdk3 3.6]	gdk_keymap_get_modifier_mask [Gdk3 3.6]
gdk_keymap_get_modifier_state [Gdk3 3.6]	gdk_keymap_get_num_lock_state [Gdk3 3.6]
gdk_keymap_have_bidi_layouts [Gdk3 3.6]	gdk_keymap_lookup_key [Gdk3 3.6]
gdk_keymap_map_virtual_modifiers [Gdk3 3.6]	gdk_keymap_translate_keyboard_state [Gdk3 3.6]
gdk_keyval_convert_case [Gdk3 3.6]	gdk_keyval_from_name [Gdk3 3.6]
gdk_keyval_is_lower [Gdk3 3.6]	gdk_keyval_is_upper [Gdk3 3.6]
gdk_keyval_name [Gdk3 3.6]	gdk_keyval_to_lower [Gdk3 3.6]
gdk_keyval_to_unicode [Gdk3 3.6]	gdk_keyval_to_upper [Gdk3 3.6]
gdk_list_visuals [Gdk3 3.6]	gdk_notify_startup_complete [Gdk3 3.6]
gdk_notify_startup_complete_with_id [Gdk3 3.6]	gdk_offscreen_window_get_embedder [Gdk3 3.6]
gdk_offscreen_window_get_surface [Gdk3 3.6]	gdk_offscreen_window_set_embedder [Gdk3 3.6]
gdk_pango_context_get [Gdk3 3.6]	gdk_pango_context_get_for_screen [Gdk3 3.6]
gdk_pango_layout_get_clip_region [Gdk3 3.6]	gdk_pango_layout_line_get_clip_region [Gdk3 3.6]
gdk_parse_args [Gdk3 3.6]	gdk_pixbuf_get_from_surface [Gdk3 3.6]
gdk_pixbuf_get_from_window [Gdk3 3.6]	gdk_pointer_grab [Gdk3 3.6]
gdk_pointer_is_grabbed [Gdk3 3.6]	gdk_pointer_ungrab [Gdk3 3.6]
gdk_property_change [Gdk3 3.6]	gdk_property_delete [Gdk3 3.6]
gdk_property_get [Gdk3 3.6]	gdk_query_depths [Gdk3 3.6]
gdk_query_visual_types [Gdk3 3.6]	gdk_rectangle_intersect [Gdk3 3.6]
gdk_rectangle_union [Gdk3 3.6]	gdk_rgba_copy [Gdk3 3.6]
gdk_rgba_equal [Gdk3 3.6]	gdk_rgba_free [Gdk3 3.6]
gdk_rgba_hash [Gdk3 3.6]	gdk_rgba_parse [Gdk3 3.6]
gdk_rgba_to_string [Gdk3 3.6]	gdk_screen_get_active_window [Gdk3 3.6]
gdk_screen_get_default [Gdk3 3.6]	gdk_screen_get_display [Gdk3 3.6]
gdk_screen_get_font_options [Gdk3 3.6]	gdk_screen_get_height [Gdk3 3.6]

gdk_screen_get_height_mm [Gdk3 3.6]	gdk_screen_get_monitor_at_point [Gdk3 3.6]
gdk_screen_get_monitor_at_window [Gdk3 3.6]	gdk_screen_get_monitor_geometry [Gdk3 3.6]
gdk_screen_get_monitor_height_mm [Gdk3 3.6]	gdk_screen_get_monitor_plug_name [Gdk3 3.6]
gdk_screen_get_monitor_width_mm [Gdk3 3.6]	gdk_screen_get_monitor_workarea [Gdk3 3.6]
gdk_screen_get_n_monitors [Gdk3 3.6]	gdk_screen_get_number [Gdk3 3.6]
gdk_screen_get_primary_monitor [Gdk3 3.6]	gdk_screen_get_resolution [Gdk3 3.6]
gdk_screen_get_rgba_visual [Gdk3 3.6]	gdk_screen_get_root_window [Gdk3 3.6]
gdk_screen_get_setting [Gdk3 3.6]	gdk_screen_get_system_visual [Gdk3 3.6]
gdk_screen_get_toplevel_windows [Gdk3 3.6]	gdk_screen_get_width [Gdk3 3.6]
gdk_screen_get_width_mm [Gdk3 3.6]	gdk_screen_get_window_stack [Gdk3 3.6]
gdk_screen_height [Gdk3 3.6]	gdk_screen_height_mm [Gdk3 3.6]
gdk_screen_is_composited [Gdk3 3.6]	gdk_screen_list_visuals [Gdk3 3.6]
gdk_screen_make_display_name [Gdk3 3.6]	gdk_screen_set_font_options [Gdk3 3.6]
gdk_screen_set_resolution [Gdk3 3.6]	gdk_screen_width [Gdk3 3.6]
gdk_screen_width_mm [Gdk3 3.6]	gdk_selection_convert [Gdk3 3.6]
gdk_selection_owner_get [Gdk3 3.6]	gdk_selection_owner_get_for_display [Gdk3 3.6]
gdk_selection_owner_set [Gdk3 3.6]	gdk_selection_owner_set_for_display [Gdk3 3.6]
gdk_selection_property_get [Gdk3 3.6]	gdk_selection_send_notify [Gdk3 3.6]
gdk_selection_send_notify_for_display [Gdk3 3.6]	gdk_set_double_click_time [Gdk3 3.6]
gdk_set_program_class [Gdk3 3.6]	gdk_set_show_events [Gdk3 3.6]
gdk_setting_get [Gdk3 3.6]	gdk_test_render_sync [Gdk3 3.6]
gdk_test_simulate_button [Gdk3 3.6]	gdk_test_simulate_key [Gdk3 3.6]
gdk_text_property_to_utf8_list_for_display [Gdk3 3.6]	gdk_threads_add_idle [Gdk3 3.6]

gdk_threads_add_idle_full [Gdk3 3.6]	gdk_threads_add_timeout [Gdk3 3.6]
gdk_threads_add_timeout_full [Gdk3 3.6]	gdk_threads_add_timeout_seconds [Gdk3 3.6]
gdk_threads_add_timeout_seconds_full [Gdk3 3.6]	gdk_threads_enter [Gdk3 3.6]
gdk_threads_init [Gdk3 3.6]	gdk_threads_leave [Gdk3 3.6]
gdk_threads_set_lock_functions [Gdk3 3.6]	gdk_unicode_to_keyval [Gdk3 3.6]
gdk_utf8_to_string_target [Gdk3 3.6]	gdk_visual_get_best [Gdk3 3.6]
gdk_visual_get_best_depth [Gdk3 3.6]	gdk_visual_get_best_type [Gdk3 3.6]
gdk_visual_get_best_with_both [Gdk3 3.6]	gdk_visual_get_best_with_depth [Gdk3 3.6]
gdk_visual_get_best_with_type [Gdk3 3.6]	gdk_visual_get_bits_per_rgb [Gdk3 3.6]
gdk_visual_get_blue_pixel_details [Gdk3 3.6]	gdk_visual_get_byte_order [Gdk3 3.6]
gdk_visual_get_colormap_size [Gdk3 3.6]	gdk_visual_get_depth [Gdk3 3.6]
gdk_visual_get_green_pixel_details [Gdk3 3.6]	gdk_visual_get_red_pixel_details [Gdk3 3.6]
gdk_visual_get_screen [Gdk3 3.6]	gdk_visual_get_system [Gdk3 3.6]
gdk_visual_get_visual_type [Gdk3 3.6]	gdk_window_add_filter [Gdk3 3.6]
gdk_window_at_pointer [Gdk3 3.6]	gdk_window_beep [Gdk3 3.6]
gdk_window_begin_move_drag [Gdk3 3.6]	gdk_window_begin_move_drag_for_device [Gdk3 3.6]
gdk_window_begin_paint_rect [Gdk3 3.6]	gdk_window_begin_paint_region [Gdk3 3.6]
gdk_window_begin_resize_drag [Gdk3 3.6]	gdk_window_begin_resize_drag_for_device [Gdk3 3.6]
gdk_window_configure_finished [Gdk3 3.6]	gdk_window_constrain_size [Gdk3 3.6]
gdk_window_coords_from_parent [Gdk3 3.6]	gdk_window_coords_to_parent [Gdk3 3.6]
gdk_window_create_similar_surface [Gdk3 3.6]	gdk_window_deiconify [Gdk3 3.6]
gdk_window_destroy [Gdk3 3.6]	gdk_window_enable_synchronized_configure [Gdk3 3.6]

gdk_window_end_paint [Gdk3 3.6]	gdk_window_ensure_native [Gdk3 3.6]
gdk_window_flush [Gdk3 3.6]	gdk_window_focus [Gdk3 3.6]
gdk_window_freeze_updates [Gdk3 3.6]	gdk_window_fullscreen [Gdk3 3.6]
gdk_window_geometry_changed [Gdk3 3.6]	gdk_window_get_accept_focus [Gdk3 3.6]
gdk_window_get_background_pattern [Gdk3 3.6]	gdk_window_get_children [Gdk3 3.6]
gdk_window_get_clip_region [Gdk3 3.6]	gdk_window_get_composited [Gdk3 3.6]
gdk_window_get_cursor [Gdk3 3.6]	gdk_window_get_decorations [Gdk3 3.6]
gdk_window_get_device_cursor [Gdk3 3.6]	gdk_window_get_device_events [Gdk3 3.6]
gdk_window_get_device_position [Gdk3 3.6]	gdk_window_get_display [Gdk3 3.6]
gdk_window_get_drag_protocol [Gdk3 3.6]	gdk_window_get_effective_parent [Gdk3 3.6]
gdk_window_get_effective_toplevel [Gdk3 3.6]	gdk_window_get_events [Gdk3 3.6]
gdk_window_get_focus_on_map [Gdk3 3.6]	gdk_window_get_frame_extents [Gdk3 3.6]
gdk_window_get_geometry [Gdk3 3.6]	gdk_window_get_group [Gdk3 3.6]
gdk_window_get_height [Gdk3 3.6]	gdk_window_get_modal_hint [Gdk3 3.6]
gdk_window_get_origin [Gdk3 3.6]	gdk_window_get_parent [Gdk3 3.6]
gdk_window_get_pointer [Gdk3 3.6]	gdk_window_get_position [Gdk3 3.6]
gdk_window_get_root_coords [Gdk3 3.6]	gdk_window_get_root_origin [Gdk3 3.6]
gdk_window_get_screen [Gdk3 3.6]	gdk_window_get_source_events [Gdk3 3.6]
gdk_window_get_state [Gdk3 3.6]	gdk_window_get_support_multidevice [Gdk3 3.6]
gdk_window_get_toplevel [Gdk3 3.6]	gdk_window_get_type_hint [Gdk3 3.6]
gdk_window_get_update_area [Gdk3 3.6]	gdk_window_get_user_data [Gdk3 3.6]
gdk_window_get_visible_region [Gdk3 3.6]	gdk_window_get_visual [Gdk3 3.6]

gdk_window_get_width [Gdk3 3.6]	gdk_window_get_window_type [Gdk3 3.6]
gdk_window_has_native [Gdk3 3.6]	gdk_window_hide [Gdk3 3.6]
gdk_window_iconify [Gdk3 3.6]	gdk_window_input_shape_combine_region [Gdk3 3.6]
gdk_window_invalidate_maybe_recurse [Gdk3 3.6]	gdk_window_invalidate_rect [Gdk3 3.6]
gdk_window_invalidate_region [Gdk3 3.6]	gdk_window_is_destroyed [Gdk3 3.6]
gdk_window_is_input_only [Gdk3 3.6]	gdk_window_is_shaped [Gdk3 3.6]
gdk_window_is_viewable [Gdk3 3.6]	gdk_window_is_visible [Gdk3 3.6]
gdk_window_lower [Gdk3 3.6]	gdk_window_maximize [Gdk3 3.6]
gdk_window_merge_child_input_shapes [Gdk3 3.6]	gdk_window_merge_child_shapes [Gdk3 3.6]
gdk_window_move [Gdk3 3.6]	gdk_window_move_region [Gdk3 3.6]
gdk_window_move_resize [Gdk3 3.6]	gdk_window_new [Gdk3 3.6]
gdk_window_peek_children [Gdk3 3.6]	gdk_window_process_all_updates [Gdk3 3.6]
gdk_window_process_updates [Gdk3 3.6]	gdk_window_raise [Gdk3 3.6]
gdk_window_register_dnd [Gdk3 3.6]	gdk_window_remove_filter [Gdk3 3.6]
gdk_window_reparent [Gdk3 3.6]	gdk_window_resize [Gdk3 3.6]
gdk_window_restack [Gdk3 3.6]	gdk_window_scroll [Gdk3 3.6]
gdk_window_set_accept_focus [Gdk3 3.6]	gdk_window_set_background [Gdk3 3.6]
gdk_window_set_background_pattern [Gdk3 3.6]	gdk_window_set_background_rgba [Gdk3 3.6]
gdk_window_set_child_input_shapes [Gdk3 3.6]	gdk_window_set_child_shapes [Gdk3 3.6]
gdk_window_set_composited [Gdk3 3.6]	gdk_window_set_cursor [Gdk3 3.6]
gdk_window_set_debug_updates [Gdk3 3.6]	gdk_window_set_decorations [Gdk3 3.6]
gdk_window_set_device_cursor [Gdk3 3.6]	gdk_window_set_device_events [Gdk3 3.6]
gdk_window_set_events [Gdk3 3.6]	gdk_window_set_focus_on_map [Gdk3 3.6]

gdk_window_set_functions [Gdk3 3.6]	gdk_window_set_geometry_hints [Gdk3 3.6]
gdk_window_set_group [Gdk3 3.6]	gdk_window_set_icon_list [Gdk3 3.6]
gdk_window_set_icon_name [Gdk3 3.6]	gdk_window_set_keep_above [Gdk3 3.6]
gdk_window_set_keep_below [Gdk3 3.6]	gdk_window_set_modal_hint [Gdk3 3.6]
gdk_window_set_opacity [Gdk3 3.6]	gdk_window_set_override_redirect [Gdk3 3.6]
gdk_window_set_role [Gdk3 3.6]	gdk_window_set_skip_pager_hint [Gdk3 3.6]
gdk_window_set_skip_taskbar_hint [Gdk3 3.6]	gdk_window_set_source_events [Gdk3 3.6]
gdk_window_set_startup_id [Gdk3 3.6]	gdk_window_set_static_gravities [Gdk3 3.6]
gdk_window_set_support_multidevice [Gdk3 3.6]	gdk_window_set_title [Gdk3 3.6]
gdk_window_set_transient_for [Gdk3 3.6]	gdk_window_set_type_hint [Gdk3 3.6]
gdk_window_set_urgency_hint [Gdk3 3.6]	gdk_window_set_user_data [Gdk3 3.6]
gdk_window_shape_combine_region [Gdk3 3.6]	gdk_window_show [Gdk3 3.6]
gdk_window_show_unraised [Gdk3 3.6]	gdk_window_stick [Gdk3 3.6]
gdk_window_thaw_updates [Gdk3 3.6]	gdk_window_unfullscreen [Gdk3 3.6]
gdk_window_unmaximize [Gdk3 3.6]	gdk_window_unstick [Gdk3 3.6]
gdk_window_withdraw [Gdk3 3.6]	gdk_x11_atom_to_xatom [Gdk3 3.6]
gdk_x11_atom_to_xatom_for_display [Gdk3 3.6]	gdk_x11_cursor_get_xcursor [Gdk3 3.6]
gdk_x11_cursor_get_xdisplay [Gdk3 3.6]	gdk_x11_device_get_id [Gdk3 3.6]
gdk_x11_device_manager_lookup [Gdk3 3.6]	gdk_x11_display_broadcast_startup_message [Gdk3 3.6]
gdk_x11_display_error_trap_pop [Gdk3 3.6]	gdk_x11_display_error_trap_pop_ignored [Gdk3 3.6]
gdk_x11_display_error_trap_push [Gdk3 3.6]	gdk_x11_display_get_startup_notification_id [Gdk3 3.6]

gdk_x11_display_get_user_time [Gdk3 3.6]	gdk_x11_display_get_xdisplay [Gdk3 3.6]
gdk_x11_display_grab [Gdk3 3.6]	gdk_x11_display_set_cursor_theme [Gdk3 3.6]
gdk_x11_display_set_startup_notification_id [Gdk3 3.6]	gdk_x11_display_string_to_compound_text [Gdk3 3.6]
gdk_x11_display_text_property_to_text_list [Gdk3 3.6]	gdk_x11_display_ungrab [Gdk3 3.6]
gdk_x11_display_utf8_to_compound_text [Gdk3 3.6]	gdk_x11_free_compound_text [Gdk3 3.6]
gdk_x11_free_text_list [Gdk3 3.6]	gdk_x11_get_default_root_xwindow [Gdk3 3.6]
gdk_x11_get_default_screen [Gdk3 3.6]	gdk_x11_get_default_xdisplay [Gdk3 3.6]
gdk_x11_get_server_time [Gdk3 3.6]	gdk_x11_get_xatom_by_name [Gdk3 3.6]
gdk_x11_get_xatom_by_name_for_display [Gdk3 3.6]	gdk_x11_get_xatom_name [Gdk3 3.6]
gdk_x11_get_xatom_name_for_display [Gdk3 3.6]	gdk_x11_grab_server [Gdk3 3.6]
gdk_x11_keymap_get_group_for_state [Gdk3 3.6]	gdk_x11_keymap_key_is_modifier [Gdk3 3.6]
gdk_x11_lookup_xdisplay [Gdk3 3.6]	gdk_x11_register_standard_event_type [Gdk3 3.6]
gdk_x11_screen_get_monitor_output [Gdk3 3.6]	gdk_x11_screen_get_screen_number [Gdk3 3.6]
gdk_x11_screen_get_window_manager_name [Gdk3 3.6]	gdk_x11_screen_get_xscreen [Gdk3 3.6]
gdk_x11_screen_lookup_visual [Gdk3 3.6]	gdk_x11_screen_supports_net_wm_hint [Gdk3 3.6]
gdk_x11_set_sm_client_id [Gdk3 3.6]	gdk_x11_ungrab_server [Gdk3 3.6]
gdk_x11_visual_get_xvisual [Gdk3 3.6]	gdk_x11_window_foreign_new_for_display [Gdk3 3.6]
gdk_x11_window_get_xid [Gdk3 3.6]	gdk_x11_window_lookup_for_display [Gdk3 3.6]
gdk_x11_window_move_to_current_desktop [Gdk3 3.6]	gdk_x11_window_set_hide_titlebar_when_maximized [Gdk3 3.6]
gdk_x11_window_set_theme_variant [Gdk3 3.6]	gdk_x11_window_set_user_time [Gdk3 3.6]
gdk_x11_xatom_to_atom [Gdk3 3.6]	gdk_x11_xatom_to_atom_for_display [Gdk3 3.6]

7.3 Data Definitions for libgdk-3

This section defines global identifiers and their values that are associated with interfaces contained in libgdk-3. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

7.3.1 gtk-3.0/gdk/gdk.h

```
#define GDK_AVAILABLE_IN_3_0
#define GDK_AVAILABLE_IN_3_2
#define GDK_AVAILABLE_IN_3_4
#define GDK_AVAILABLE_IN_3_6
#define GDK_THREADS_DEPRECATED
#define GDK_WINDOWING_BROADWAY
#define GDK_BUTTON_PRIMARY (1)
#define GDK_BUTTON_MIDDLE (2)
#define GDK_BUTTON_SECONDARY (3)
#define GDK_MAJOR_VERSION (3)
#define GDK_MICRO_VERSION (4)
#define GDK_MINOR_VERSION (6)
#define GDK_EVENT_PROPAGATE (FALSE)
#define GDK_TYPE_APP_LAUNCH_CONTEXT
(gdk_app_launch_context_get_type ())
#define GDK_TYPE_DEVICE_MANAGER (gdk_device_manager_get_type ())
#define GDK_TYPE_DEVICE_TYPE (gdk_device_type_get_type ())
#define GDK_TYPE_GRAB_OWNERSHIP (gdk_grab_ownership_get_type ())
#define GDK_TYPE_MODIFIER_INTENT
(gdk_modifier_intent_get_type ())
#define GDK_TYPE_RGBA (gdk_rgba_get_type ())
#define GDK_VERSION_MIN_REQUIRED (GDK_VERSION_CUR_STABLE)
#define GDK_TYPE_WINDOW_WINDOW_CLASS
(gdk_window_window_class_get_type ())
#define GDK_VERSION_3_0 (G_ENCODE_VERSION (3, 0))
#define GDK_VERSION_3_2 (G_ENCODE_VERSION (3, 2))
#define GDK_VERSION_3_4 (G_ENCODE_VERSION (3, 4))
#define GDK_VERSION_3_6 (G_ENCODE_VERSION (3, 6))
#define GDK_VERSION_PREV_STABLE (G_ENCODE_VERSION
(GDK_MAJOR_VERSION, GDK_MINOR_VERSION - 2))
#define GDK_VERSION_CUR_STABLE (G_ENCODE_VERSION
(GDK_MAJOR_VERSION, GDK_MINOR_VERSION))
#define GDK_APP_LAUNCH_CONTEXT(o) (G_TYPE_CHECK_INSTANCE_CAST
((o), GDK_TYPE_APP_LAUNCH_CONTEXT, GdkAppLaunchContext))
#define GDK_DEVICE(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GDK_TYPE_DEVICE, GdkDevice))
#define GDK_DEVICE_MANAGER(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GDK_TYPE_DEVICE_MANAGER, GdkDeviceManager))
#define GDK_CURSOR(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_CURSOR, GdkCursor))
#define GDK_DISPLAY(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_DISPLAY, GdkDisplay))
```

```

#define GDK_IS_APP_LAUNCH_CONTEXT(o)
(G_TYPE_CHECK_INSTANCE_TYPE ((o), GDK_TYPE_APP_LAUNCH_CONTEXT))
#define GDK_IS_DEVICE(o) (G_TYPE_CHECK_INSTANCE_TYPE ((o),
GDK_TYPE_DEVICE))
#define GDK_IS_DEVICE_MANAGER(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GDK_TYPE_DEVICE_MANAGER))
#define GDK_IS_CURSOR(object) (G_TYPE_CHECK_INSTANCE_TYPE
((object), GDK_TYPE_CURSOR))
#define GDK_EVENT_STOP (TRUE)
#define __INT16_C(c) c
#define __INT32_C(c) c
#define __INT8_C(c) c
#define __UINT16_C(c) c
#define __UINT8_C(c) c
#define __INTMAX_C(c) c ## L
#define __UINT32_C(c) c ## U
#define __UINTMAX_C(c) c ## UL
#define GDK_DEPRECATED_IN_3_0 GDK_DEPRECATED
#define GDK_DEPRECATED_IN_3_2 GDK_DEPRECATED
#define GDK_DEPRECATED_IN_3_4 GDK_DEPRECATED
#define GDK_DEPRECATED_IN_3_6 GDK_DEPRECATED
#define GDK_DEPRECATED_IN_3_0_FOR(f) GDK_DEPRECATED_FOR(f)
#define GDK_DEPRECATED_IN_3_2_FOR(f) GDK_DEPRECATED_FOR(f)
#define GDK_DEPRECATED_IN_3_4_FOR(f) GDK_DEPRECATED_FOR(f)
#define GDK_DEPRECATED_IN_3_6_FOR(f) GDK_DEPRECATED_FOR(f)
#define GDK_VERSION_MAX_ALLOWED GDK_VERSION_MIN_REQUIRED
#define GDK_DEPRECATED_G_DEPRECATED
#define GDK_DEPRECATED_FOR(f) G_DEPRECATED_FOR(f)
#define GDK_UNAVAILABLE(maj,min) G_UNAVAILABLE(maj,min)

typedef struct _GdkKeymapKey {
    guint keycode;
    gint group;
    gint level;
} GdkKeymapKey;
typedef struct _GdkEventAny {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
} GdkEventAny;
typedef struct _GdkEventExpose {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkRectangle area;
    cairo_region_t *region;
    gint count;
} GdkEventExpose;
typedef struct _GdkEventVisibility {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkVisibilityState state;
} GdkEventVisibility;
typedef struct _GdkEventMotion {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    gdouble x;
    gdouble y;
    gdouble *axes;
    guint state;
    gint16 is_hint;
    GdkDevice *device;
    gdouble x_root;

```

```

    gdouble y_root;
} GdkEventMotion;
typedef struct _GdkEventButton {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    gdouble x;
    gdouble y;
    gdouble *axes;
    guint state;
    guint button;
    GdkDevice *device;
    gdouble x_root;
    gdouble y_root;
} GdkEventButton;
typedef struct _GdkEventTouch {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    gdouble x;
    gdouble y;
    gdouble *axes;
    guint state;
    GdkEventSequence *sequence;
    gboolean emulating_pointer;
    GdkDevice *device;
    gdouble x_root;
    gdouble y_root;
} GdkEventTouch;
typedef struct _GdkEventScroll {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    gdouble x;
    gdouble y;
    guint state;
    GdkScrollDirection direction;
    GdkDevice *device;
    gdouble x_root;
    gdouble y_root;
    gdouble delta_x;
    gdouble delta_y;
} GdkEventScroll;
typedef struct _GdkEventKey {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    guint state;
    guint keyval;
    gint length;
    gchar *string;
    guint16 hardware_keycode;
    guint8 group;
    guint is_modifier;
} GdkEventKey;
typedef struct _GdkEventFocus {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    gint16 in;
} GdkEventFocus;
typedef struct _GdkEventCrossing {

```

```

    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkWindow *subwindow;
    guint32 time;
    gdouble x;
    gdouble y;
    gdouble x_root;
    gdouble y_root;
    GdkCrossingMode mode;
    GdkNotifyType detail;
    gboolean focus;
    guint state;
} GdkEventCrossing;
typedef struct _GdkEventConfigure {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    gint x;
    gint y;
    gint width;
    gint height;
} GdkEventConfigure;
typedef struct _GdkEventProperty {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkAtom atom;
    guint32 time;
    guint state;
} GdkEventProperty;
typedef struct _GdkEventSelection {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkAtom selection;
    GdkAtom target;
    GdkAtom property;
    guint32 time;
    GdkWindow *requestor;
} GdkEventSelection;
typedef struct _GdkEventOwnerChange {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkWindow *owner;
    GdkOwnerChange reason;
    GdkAtom selection;
    guint32 time;
    guint32 selection_time;
} GdkEventOwnerChange;
typedef struct _GdkEventProximity {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    guint32 time;
    GdkDevice *device;
} GdkEventProximity;
typedef struct _GdkEventDND {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkDragContext *context;
    guint32 time;
    gshort x_root;
    gshort y_root;

```

```

} GdkEventDND;
typedef struct _GdkEventWindowState {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkWindowState changed_mask;
    GdkWindowState new_window_state;
} GdkEventWindowState;
typedef struct _GdkEventSetting {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    GdkSettingAction action;
    char *name;
} GdkEventSetting;
typedef struct _GdkEventGrabBroken {
    GdkEventType type;
    GdkWindow *window;
    gint8 send_event;
    gboolean keyboard;
    gboolean implicit;
    GdkWindow *grab_window;
} GdkEventGrabBroken;
typedef struct _GdkEventSequence GdkEventSequence;
typedef union _GdkEvent {
    GdkEventType type;
    GdkEventAny any;
    GdkEventExpose expose;
    GdkEventVisibility visibility;
    GdkEventMotion motion;
    GdkEventButton button;
    GdkEventTouch touch;
    GdkEventScroll scroll;
    GdkEventKey key;
    GdkEventCrossing crossing;
    GdkEventFocus focus_change;
    GdkEventConfigure configure;
    GdkEventProperty property;
    GdkEventSelection selection;
    GdkEventOwnerChange owner_change;
    GdkEventProximity proximity;
    GdkEventDND dnd;
    GdkEventWindowState window_state;
    GdkEventSetting setting;
    GdkEventGrabBroken grab_broken;
} GdkEvent;
typedef void (*GdkEventFunc) (GdkEvent * event, gpointer data);
typedef void GdkXEvent;
typedef enum {
    GDK_FILTER_CONTINUE,
    GDK_FILTER_TRANSLATE,
    GDK_FILTER_REMOVE
} GdkFilterReturn;
typedef GdkFilterReturn(*GdkFilterFunc) (GdkXEvent * xevent,
                                         GdkEvent * event, gpointer data);
typedef enum {
    GDK_NOTHING = -1,
    GDK_DELETE = 0,
    GDK_DESTROY = 1,
    GDK_EXPOSE = 2,
    GDK_MOTION_NOTIFY = 3,
    GDK_BUTTON_PRESS = 4,
    GDK_2BUTTON_PRESS = 5,
    GDK_DOUBLE_BUTTON_PRESS = GDK_2BUTTON_PRESS,
    GDK_3BUTTON_PRESS = 6,
    GDK_TRIPLE_BUTTON_PRESS = GDK_3BUTTON_PRESS,

```

```

    GDK_BUTTON_RELEASE = 7,
    GDK_KEY_PRESS = 8,
    GDK_KEY_RELEASE = 9,
    GDK_ENTER_NOTIFY = 10,
    GDK_LEAVE_NOTIFY = 11,
    GDK_FOCUS_CHANGE = 12,
    GDK_CONFIGURE = 13,
    GDK_MAP = 14,
    GDK_UNMAP = 15,
    GDK_PROPERTY_NOTIFY = 16,
    GDK_SELECTION_CLEAR = 17,
    GDK_SELECTION_REQUEST = 18,
    GDK_SELECTION_NOTIFY = 19,
    GDK_PROXIMITY_IN = 20,
    GDK_PROXIMITY_OUT = 21,
    GDK_DRAG_ENTER = 22,
    GDK_DRAG_LEAVE = 23,
    GDK_DRAG_MOTION = 24,
    GDK_DRAG_STATUS = 25,
    GDK_DROP_START = 26,
    GDK_DROP_FINISHED = 27,
    GDK_CLIENT_EVENT = 28,
    GDK_VISIBILITY_NOTIFY = 29,
    GDK_SCROLL = 31,
    GDK_WINDOW_STATE = 32,
    GDK_SETTING = 33,
    GDK_OWNER_CHANGE = 34,
    GDK_GRAB_BROKEN = 35,
    GDK_DAMAGE = 36,
    GDK_TOUCH_BEGIN = 37,
    GDK_TOUCH_UPDATE = 38,
    GDK_TOUCH_END = 39,
    GDK_TOUCH_CANCEL = 40,
    GDK_EVENT_LAST
} GdkEventType;
typedef enum {
    GDK_VISIBILITY_UNOBSCURED,
    GDK_VISIBILITY_PARTIAL,
    GDK_VISIBILITY_FULLY_OBSCURED
} GdkVisibilityState;
typedef enum {
    GDK_SCROLL_UP,
    GDK_SCROLL_DOWN,
    GDK_SCROLL_LEFT,
    GDK_SCROLL_RIGHT,
    GDK_SCROLL_SMOOTH
} GdkScrollDirection;
typedef enum {
    GDK_NOTIFY_ANCESTOR = 0,
    GDK_NOTIFY_VIRTUAL = 1,
    GDK_NOTIFY_INFERIOR = 2,
    GDK_NOTIFY_NONLINEAR = 3,
    GDK_NOTIFY_NONLINEAR_VIRTUAL = 4,
    GDK_NOTIFY_UNKNOWN = 5
} GdkNotifyType;
typedef enum {
    GDK_CROSSING_NORMAL,
    GDK_CROSSING_GRAB,
    GDK_CROSSING_UNGRAB,
    GDK_CROSSING_GTK_GRAB,
    GDK_CROSSING_GTK_UNGRAB,
    GDK_CROSSING_STATE_CHANGED,
    GDK_CROSSING_TOUCH_BEGIN,
    GDK_CROSSING_TOUCH_END,
    GDK_CROSSING_DEVICE_SWITCH
} GdkCrossingMode;

```

```

typedef enum {
    GDK_PROPERTY_NEW_VALUE,
    GDK_PROPERTY_DELETE
} GdkPropertyState;
typedef enum {
    GDK_WINDOW_STATE_WITHDRAWN = 1 << 0,
    GDK_WINDOW_STATE_ICONIFIED = 1 << 1,
    GDK_WINDOW_STATE_MAXIMIZED = 1 << 2,
    GDK_WINDOW_STATE_STICKY = 1 << 3,
    GDK_WINDOW_STATE_FULLSCREEN = 1 << 4,
    GDK_WINDOW_STATE_ABOVE = 1 << 5,
    GDK_WINDOW_STATE_BELOW = 1 << 6,
    GDK_WINDOW_STATE_FOCUSED = 1 << 7
} GdkWindowState;
typedef enum {
    GDK_SETTING_ACTION_NEW,
    GDK_SETTING_ACTION_CHANGED,
    GDK_SETTING_ACTION_DELETED
} GdkSettingAction;
typedef enum {
    GDK_OWNER_CHANGE_NEW_OWNER,
    GDK_OWNER_CHANGE_DESTROY,
    GDK_OWNER_CHANGE_CLOSE
} GdkOwnerChange;
union _GdkEvent {
    GdkEventType type;
    GdkEventAny any;
    GdkEventExpose expose;
    GdkEventVisibility visibility;
    GdkEventMotion motion;
    GdkEventButton button;
    GdkEventTouch touch;
    GdkEventScroll scroll;
    GdkEventKey key;
    GdkEventCrossing crossing;
    GdkEventFocus focus_change;
    GdkEventConfigure configure;
    GdkEventProperty property;
    GdkEventSelection selection;
    GdkEventOwnerChange owner_change;
    GdkEventProximity proximity;
    GdkEventDND dnd;
    GdkEventWindowState window_state;
    GdkEventSetting setting;
    GdkEventGrabBroken grab_broken;
};
typedef enum {
    GDK_PROP_MODE_REPLACE,
    GDK_PROP_MODE_PREPEND,
    GDK_PROP_MODE_APPEND
} GdkPropMode;
typedef enum {
    GDK_VISUAL_STATIC_GRAY,
    GDK_VISUAL_GRAYSCALE,
    GDK_VISUAL_STATIC_COLOR,
    GDK_VISUAL_PSEUDO_COLOR,
    GDK_VISUAL_TRUE_COLOR,
    GDK_VISUAL_DIRECT_COLOR
} GdkVisualType;
typedef enum {
    GDK_ACTION_DEFAULT = 1 << 0,
    GDK_ACTION_COPY = 1 << 1,
    GDK_ACTION_MOVE = 1 << 2,
    GDK_ACTION_LINK = 1 << 3,
    GDK_ACTION_PRIVATE = 1 << 4,
    GDK_ACTION_ASK = 1 << 5
}

```

```

} GdkDragAction;
typedef enum {
    GDK_DRAG_PROTO_NONE,
    GDK_DRAG_PROTO_MOTIF,
    GDK_DRAG_PROTO_XDND,
    GDK_DRAG_PROTO_ROOTWIN,
    GDK_DRAG_PROTO_WIN32_DROPPFILES,
    GDK_DRAG_PROTO_OLE2,
    GDK_DRAG_PROTO_LOCAL
} GdkDragProtocol;
typedef struct _GdkGeometry {
    gint min_width;
    gint min_height;
    gint max_width;
    gint max_height;
    gint base_width;
    gint base_height;
    gint width_inc;
    gint height_inc;
    gdouble min_aspect;
    gdouble max_aspect;
    GdkGravity win_gravity;
} GdkGeometry;
typedef struct _GdkWindowAttr {
    gchar *title;
    gint event_mask;
    gint x;
    gint y;
    gint width;
    gint height;
    GdkWindowWindowClass wclass;
    GdkVisual *visual;
    GdkWindowType window_type;
    GdkCursor *cursor;
    gchar *wmclass_name;
    gchar *wmclass_class;
    gboolean override_redirect;
    GdkWindowTypeHint type_hint;
} GdkWindowAttr;
typedef struct _GdkWindowRedirect GdkWindowRedirect;
typedef enum {
    GDK_INPUT_OUTPUT,
    GDK_INPUT_ONLY
} GdkWindowWindowClass;
typedef enum {
    GDK_WINDOW_ROOT,
    GDK_WINDOW_TOPLEVEL,
    GDK_WINDOW_CHILD,
    GDK_WINDOW_TEMP,
    GDK_WINDOW_FOREIGN,
    GDK_WINDOW_OFFSCREEN
} GdkWindowType;
typedef enum {
    GDK_WA_TITLE = 1 << 1,
    GDK_WA_X = 1 << 2,
    GDK_WA_Y = 1 << 3,
    GDK_WA_CURSOR = 1 << 4,
    GDK_WA_VISUAL = 1 << 5,
    GDK_WA_WMCLASS = 1 << 6,
    GDK_WA_NOREDIR = 1 << 7,
    GDK_WA_TYPE_HINT = 1 << 8
} GdkWindowAttributesType;
typedef enum {
    GDK_HINT_POS = 1 << 0,
    GDK_HINT_MIN_SIZE = 1 << 1,
    GDK_HINT_MAX_SIZE = 1 << 2,

```



```

    GDK_HINT_BASE_SIZE = 1 << 3,
    GDK_HINT_ASPECT = 1 << 4,
    GDK_HINT_RESIZE_INC = 1 << 5,
    GDK_HINT_WIN_GRAVITY = 1 << 6,
    GDK_HINT_USER_POS = 1 << 7,
    GDK_HINT_USER_SIZE = 1 << 8
} GdkWindowHints;
typedef enum {
    GDK_WINDOW_TYPE_HINT_NORMAL,
    GDK_WINDOW_TYPE_HINT_DIALOG,
    GDK_WINDOW_TYPE_HINT_MENU,
    GDK_WINDOW_TYPE_HINT_TOOLBAR,
    GDK_WINDOW_TYPE_HINT_SPLASHSCREEN,
    GDK_WINDOW_TYPE_HINT_UTILITY,
    GDK_WINDOW_TYPE_HINT_DOCK,
    GDK_WINDOW_TYPE_HINT_DESKTOP,
    GDK_WINDOW_TYPE_HINT_DROPDOWN_MENU,
    GDK_WINDOW_TYPE_HINT_POPUP_MENU,
    GDK_WINDOW_TYPE_HINT_TOOLTIP,
    GDK_WINDOW_TYPE_HINT_NOTIFICATION,
    GDK_WINDOW_TYPE_HINT_COMBO,
    GDK_WINDOW_TYPE_HINT_DND
} GdkWindowTypeHint;
typedef enum {
    GDK_DECOR_ALL = 1 << 0,
    GDK_DECOR_BORDER = 1 << 1,
    GDK_DECOR_RESIZEH = 1 << 2,
    GDK_DECOR_TITLE = 1 << 3,
    GDK_DECOR_MENU = 1 << 4,
    GDK_DECOR_MINIMIZE = 1 << 5,
    GDK_DECOR_MAXIMIZE = 1 << 6
} GdkWMDecoration;
typedef enum {
    GDK_FUNC_ALL = 1 << 0,
    GDK_FUNC_RESIZE = 1 << 1,
    GDK_FUNC_MOVE = 1 << 2,
    GDK_FUNC_MINIMIZE = 1 << 3,
    GDK_FUNC_MAXIMIZE = 1 << 4,
    GDK_FUNC_CLOSE = 1 << 5
} GdkWMFunction;
typedef enum {
    GDK_GRAVITY_NORTH_WEST = 1,
    GDK_GRAVITY_NORTH = 2,
    GDK_GRAVITY_NORTH_EAST = 3,
    GDK_GRAVITY_WEST = 4,
    GDK_GRAVITY_CENTER = 5,
    GDK_GRAVITY_EAST = 6,
    GDK_GRAVITY_SOUTH_WEST = 7,
    GDK_GRAVITY_SOUTH = 8,
    GDK_GRAVITY_SOUTH_EAST = 9,
    GDK_GRAVITY_STATIC = 10
} GdkGravity;
typedef enum {
    GDK_WINDOW_EDGE_NORTH_WEST,
    GDK_WINDOW_EDGE_NORTH,
    GDK_WINDOW_EDGE_NORTH_EAST,
    GDK_WINDOW_EDGE_WEST,
    GDK_WINDOW_EDGE_EAST,
    GDK_WINDOW_EDGE_SOUTH_WEST,
    GDK_WINDOW_EDGE_SOUTH,
    GDK_WINDOW_EDGE_SOUTH_EAST
} GdkWindowEdge;
typedef struct _GdkWindowClass {
    GObjectClass parent_class;
    GdkWindow *(*pick_embedded_child) (GdkWindow * window, gdouble
x,

```

```

        gdouble y);
void (*to_embedder) (GdkWindow * window, gdouble offscreen_x,
                    gdouble offscreen_y, gdouble * embedder_x,
                    gdouble * embedder_y);
void (*from_embedder) (GdkWindow * window, gdouble offscreen_x,
                    gdouble offscreen_y, gdouble * embedder_x,
                    gdouble * embedder_y);
cairo_surface_t *(*create_surface) (GdkWindow * window, gint
width,
                                gint height);

void (*_gdk_reserved1) (void);
void (*_gdk_reserved2) (void);
void (*_gdk_reserved3) (void);
void (*_gdk_reserved4) (void);
void (*_gdk_reserved5) (void);
void (*_gdk_reserved6) (void);
void (*_gdk_reserved7) (void);
void (*_gdk_reserved8) (void);
} GdkWindowClass;
typedef gboolean(*GdkWindowChildFunc) (GdkWindow * window,
                                gpointer user_data);

typedef struct _GdkTimeCoord {
    guint32 time;
    gdouble axes[128];
} GdkTimeCoord;
typedef enum {
    GDK_SOURCE_MOUSE,
    GDK_SOURCE_PEN,
    GDK_SOURCE_ERASER,
    GDK_SOURCE_CURSOR,
    GDK_SOURCE_KEYBOARD,
    GDK_SOURCE_TOUCHSCREEN,
    GDK_SOURCE_TOUCHPAD
} GdkInputSource;
typedef enum {
    GDK_MODE_DISABLED,
    GDK_MODE_SCREEN,
    GDK_MODE_WINDOW
} GdkInputMode;
typedef enum {
    GDK_AXIS_IGNORE,
    GDK_AXIS_X,
    GDK_AXIS_Y,
    GDK_AXIS_PRESSURE,
    GDK_AXIS_XTILT,
    GDK_AXIS_YTILT,
    GDK_AXIS_WHEEL,
    GDK_AXIS_LAST
} GdkAxisUse;
typedef enum {
    GDK_DEVICE_TYPE_MASTER,
    GDK_DEVICE_TYPE_SLAVE,
    GDK_DEVICE_TYPE_FLOATING
} GdkDeviceType;
typedef enum {
    GDK_X_CURSOR = 0,
    GDK_ARROW = 2,
    GDK_BASED_ARROW_DOWN = 4,
    GDK_BASED_ARROW_UP = 6,
    GDK_BOAT = 8,
    GDK_BOGOSITY = 10,
    GDK_BOTTOM_LEFT_CORNER = 12,
    GDK_BOTTOM_RIGHT_CORNER = 14,
    GDK_BOTTOM_SIDE = 16,
    GDK_BOTTOM_TEE = 18,
    GDK_BOX_SPIRAL = 20,

```

GDK_CENTER_PTR = 22,
 GDK_CIRCLE = 24,
 GDK_CLOCK = 26,
 GDK_COFFEE_MUG = 28,
 GDK_CROSS = 30,
 GDK_CROSS_REVERSE = 32,
 GDK_CROSSHAIR = 34,
 GDK_DIAMOND_CROSS = 36,
 GDK_DOT = 38,
 GDK_DOTBOX = 40,
 GDK_DOUBLE_ARROW = 42,
 GDK_DRAFT_LARGE = 44,
 GDK_DRAFT_SMALL = 46,
 GDK_DRAPED_BOX = 48,
 GDK_EXCHANGE = 50,
 GDK_FLEUR = 52,
 GDK_GOBBLER = 54,
 GDK_GUMBY = 56,
 GDK_HAND1 = 58,
 GDK_HAND2 = 60,
 GDK_HEART = 62,
 GDK_ICON = 64,
 GDK_IRON_CROSS = 66,
 GDK_LEFT_PTR = 68,
 GDK_LEFT_SIDE = 70,
 GDK_LEFT_TEE = 72,
 GDK_LEFTBUTTON = 74,
 GDK_LL_ANGLE = 76,
 GDK_LR_ANGLE = 78,
 GDK_MAN = 80,
 GDK_MIDDLEBUTTON = 82,
 GDK_MOUSE = 84,
 GDK_PENCIL = 86,
 GDK_PIRATE = 88,
 GDK_PLUS = 90,
 GDK_QUESTION_ARROW = 92,
 GDK_RIGHT_PTR = 94,
 GDK_RIGHT_SIDE = 96,
 GDK_RIGHT_TEE = 98,
 GDK_RIGHTBUTTON = 100,
 GDK_RTL_LOGO = 102,
 GDK_SAILBOAT = 104,
 GDK_SB_DOWN_ARROW = 106,
 GDK_SB_H_DOUBLE_ARROW = 108,
 GDK_SB_LEFT_ARROW = 110,
 GDK_SB_RIGHT_ARROW = 112,
 GDK_SB_UP_ARROW = 114,
 GDK_SB_V_DOUBLE_ARROW = 116,
 GDK_SHUTTLE = 118,
 GDK_SIZING = 120,
 GDK_SPIDER = 122,
 GDK_SPRAYCAN = 124,
 GDK_STAR = 126,
 GDK_TARGET = 128,
 GDK_TCROSS = 130,
 GDK_TOP_LEFT_ARROW = 132,
 GDK_TOP_LEFT_CORNER = 134,
 GDK_TOP_RIGHT_CORNER = 136,
 GDK_TOP_SIDE = 138,
 GDK_TOP_TEE = 140,
 GDK_TREK = 142,
 GDK_UL_ANGLE = 144,
 GDK_UMBRELLA = 146,
 GDK_UR_ANGLE = 148,
 GDK_WATCH = 150,
 GDK_XTERM = 152,

```

        GDK_LAST_CURSOR,
        GDK_BLANK_CURSOR = -2,
        GDK_CURSOR_IS_PIXMAP = -1
    } GdkCursorType;
typedef struct _GdkPoint {
    gint x;
    gint y;
} GdkPoint;
typedef const cairo_rectangle_int_t GdkRectangle;
typedef struct _GdkAtom *GdkAtom;
typedef struct _GdkColor {
    guint32 pixel;
    guint16 red;
    guint16 green;
    guint16 blue;
} GdkColor;
typedef struct _GdkRGBA {
    gdouble red;
    gdouble green;
    gdouble blue;
    gdouble alpha;
} GdkRGBA;
typedef struct _GdkCursor GdkCursor;
typedef struct _GdkVisual GdkVisual;
typedef struct _GdkDevice GdkDevice;
typedef struct _GdkDragContext GdkDragContext;
typedef struct _GdkDisplayManager GdkDisplayManager;
typedef struct _GdkDeviceManager GdkDeviceManager;
typedef struct _GdkDisplay GdkDisplay;
typedef struct _GdkScreen GdkScreen;
typedef struct _GdkWindow GdkWindow;
typedef struct _GdkKeymap GdkKeymap;
typedef struct _GdkAppLaunchContext GdkAppLaunchContext;
typedef enum {
    GDK_LSB_FIRST,
    GDK_MSB_FIRST
} GdkByteOrder;
typedef enum {
    GDK_SHIFT_MASK = 1 << 0,
    GDK_LOCK_MASK = 1 << 1,
    GDK_CONTROL_MASK = 1 << 2,
    GDK_MOD1_MASK = 1 << 3,
    GDK_MOD2_MASK = 1 << 4,
    GDK_MOD3_MASK = 1 << 5,
    GDK_MOD4_MASK = 1 << 6,
    GDK_MOD5_MASK = 1 << 7,
    GDK_BUTTON1_MASK = 1 << 8,
    GDK_BUTTON2_MASK = 1 << 9,
    GDK_BUTTON3_MASK = 1 << 10,
    GDK_BUTTON4_MASK = 1 << 11,
    GDK_BUTTON5_MASK = 1 << 12,
    GDK_MODIFIER_RESERVED_13_MASK = 1 << 13,
    GDK_MODIFIER_RESERVED_14_MASK = 1 << 14,
    GDK_MODIFIER_RESERVED_15_MASK = 1 << 15,
    GDK_MODIFIER_RESERVED_16_MASK = 1 << 16,
    GDK_MODIFIER_RESERVED_17_MASK = 1 << 17,
    GDK_MODIFIER_RESERVED_18_MASK = 1 << 18,
    GDK_MODIFIER_RESERVED_19_MASK = 1 << 19,
    GDK_MODIFIER_RESERVED_20_MASK = 1 << 20,
    GDK_MODIFIER_RESERVED_21_MASK = 1 << 21,
    GDK_MODIFIER_RESERVED_22_MASK = 1 << 22,
    GDK_MODIFIER_RESERVED_23_MASK = 1 << 23,
    GDK_MODIFIER_RESERVED_24_MASK = 1 << 24,
    GDK_MODIFIER_RESERVED_25_MASK = 1 << 25,
    GDK_SUPER_MASK = 1 << 26,
    GDK_HYPER_MASK = 1 << 27,

```

```

    GDK_META_MASK = 1 << 29,
    GDK_MODIFIER_RESERVED_29_MASK = 1 << 29,
    GDK_RELEASE_MASK = 1 << 30,
    GDK_MODIFIER_MASK = 0x5c001fff
} GdkModifierType;
typedef enum {
    GDK_MODIFIER_INTENT_PRIMARY_ACCELERATOR,
    GDK_MODIFIER_INTENT_CONTEXT_MENU,
    GDK_MODIFIER_INTENT_EXTEND_SELECTION,
    GDK_MODIFIER_INTENT_MODIFY_SELECTION,
    GDK_MODIFIER_INTENT_NO_TEXT_INPUT,
    GDK_MODIFIER_INTENT_SHIFT_GROUP
} GdkModifierIntent;
typedef enum {
    GDK_OK = 0,
    GDK_ERROR = -1,
    GDK_ERROR_PARAM = -2,
    GDK_ERROR_FILE = -3,
    GDK_ERROR_MEM = -4
} GdkStatus;
typedef enum {
    GDK_GRAB_SUCCESS,
    GDK_GRAB_ALREADY_GRABBED,
    GDK_GRAB_INVALID_TIME,
    GDK_GRAB_NOT_VIEWABLE,
    GDK_GRAB_FROZEN
} GdkGrabStatus;
typedef enum {
    GDK_OWNERSHIP_NONE,
    GDK_OWNERSHIP_WINDOW,
    GDK_OWNERSHIP_APPLICATION
} GdkGrabOwnership;
typedef enum {
    GDK_EXPOSURE_MASK = 1 << 1,
    GDK_POINTER_MOTION_MASK = 1 << 2,
    GDK_POINTER_MOTION_HINT_MASK = 1 << 3,
    GDK_BUTTON_MOTION_MASK = 1 << 4,
    GDK_BUTTON1_MOTION_MASK = 1 << 5,
    GDK_BUTTON2_MOTION_MASK = 1 << 6,
    GDK_BUTTON3_MOTION_MASK = 1 << 7,
    GDK_BUTTON_PRESS_MASK = 1 << 8,
    GDK_BUTTON_RELEASE_MASK = 1 << 9,
    GDK_KEY_PRESS_MASK = 1 << 10,
    GDK_KEY_RELEASE_MASK = 1 << 11,
    GDK_ENTER_NOTIFY_MASK = 1 << 12,
    GDK_LEAVE_NOTIFY_MASK = 1 << 13,
    GDK_FOCUS_CHANGE_MASK = 1 << 14,
    GDK_STRUCTURE_MASK = 1 << 15,
    GDK_PROPERTY_CHANGE_MASK = 1 << 16,
    GDK_VISIBILITY_NOTIFY_MASK = 1 << 17,
    GDK_PROXIMITY_IN_MASK = 1 << 18,
    GDK_PROXIMITY_OUT_MASK = 1 << 19,
    GDK_SUBSTRUCTURE_MASK = 1 << 20,
    GDK_SCROLL_MASK = 1 << 21,
    GDK_TOUCH_MASK = 1 << 22,
    GDK_SMOOTH_SCROLL_MASK = 1 << 23,
    GDK_ALL_EVENTS_MASK = 0xFFFFFE
} GdkEventMask;
extern void gdk_add_option_entries_libgtk_only(GOptionGroup *
group);
extern GType gdk_app_launch_context_get_type(void);
extern GdkAppLaunchContext *gdk_app_launch_context_new(void);
extern void gdk_app_launch_context_set_desktop(GdkAppLaunchContext
*
context, gint);

```

```

extern void gdk_app_launch_context_set_display(GdkAppLaunchContext
*
context, GdkDisplay *);
extern void gdk_app_launch_context_set_icon(GdkAppLaunchContext *
context,
GIcon *);
extern void gdk_app_launch_context_set_icon_name(GdkAppLaunchContext *
context, const char *);
extern void gdk_app_launch_context_set_screen(GdkAppLaunchContext
*
context, GdkScreen *);
extern void gdk_app_launch_context_set_timestamp(GdkAppLaunchContext *
context, guint32);
extern GdkAtom gdk_atom_intern(const char *atom_name, gboolean);
extern GdkAtom gdk_atom_intern_static_string(const char
*atom_name);
extern gchar *gdk_atom_name(GdkAtom atom);
extern GType gdk_axis_use_get_type(void);
extern void gdk_beep(void);
extern GType gdk_byte_order_get_type(void);
extern cairo_t *gdk_cairo_create(GdkWindow * window);
extern gboolean gdk_cairo_get_clip_rectangle(cairo_t * cr,
GdkRectangle *);
extern void gdk_cairo_rectangle(cairo_t * cr, const GdkRectangle
*);
extern void gdk_cairo_region(cairo_t * cr, const cairo_region_t);
extern cairo_region_t *gdk_cairo_region_create_from_surface(cairo_surface_t
* surface);
extern void gdk_cairo_set_source_color(cairo_t * cr, const GdkColor
*);
extern void gdk_cairo_set_source_pixbuf(cairo_t * cr, const
GdkPixbuf *,
gdouble, gdouble);
extern void gdk_cairo_set_source_rgba(cairo_t * cr, const GdkRGBA
*);
extern void gdk_cairo_set_source_window(cairo_t * cr, GdkWindow *,
gdouble,
gdouble);
extern GdkColor *gdk_color_copy(const GdkColor * color);
extern gboolean gdk_color_equal(const GdkColor * colora, const
GdkColor *);
extern void gdk_color_free(GdkColor * color);
extern GType gdk_color_get_type(void);
extern guint gdk_color_hash(const GdkColor * color);
extern gboolean gdk_color_parse(const char *spec, GdkColor *);
extern gchar *gdk_color_to_string(const GdkColor * color);
extern GType gdk_crossing_mode_get_type(void);
extern GdkCursorType gdk_cursor_get_cursor_type(GdkCursor *
cursor);
extern GdkDisplay *gdk_cursor_get_display(GdkCursor * cursor);
extern GdkPixbuf *gdk_cursor_get_image(GdkCursor * cursor);
extern GType gdk_cursor_get_type(void);
extern GdkCursor *gdk_cursor_new(GdkCursorType cursor_type);
extern GdkCursor *gdk_cursor_new_for_display(GdkDisplay * display,
GdkCursorType);
extern GdkCursor *gdk_cursor_new_from_name(GdkDisplay * display,
const char *);
extern GdkCursor *gdk_cursor_new_from_pixbuf(GdkDisplay * display,
GdkPixbuf *, gint, gint);
extern GdkCursor *gdk_cursor_ref(GdkCursor * cursor);
extern GType gdk_cursor_type_get_type(void);
extern void gdk_cursor_unref(GdkCursor * cursor);
extern void gdk_device_free_history(GdkTimeCoord * *events, gint);

```

```

extern GdkDevice *gdk_device_get_associated_device(GdkDevice *
device);
extern gboolean gdk_device_get_axis(GdkDevice * device, gdouble *,
GdkAxisUse, gdouble *);
extern GdkAxisUse gdk_device_get_axis_use(GdkDevice * device,
guint);
extern gboolean gdk_device_get_axis_value(GdkDevice * device,
gdouble *,
GdkAtom, gdouble *);
extern GdkDeviceType gdk_device_get_device_type(GdkDevice *
device);
extern GdkDisplay *gdk_device_get_display(GdkDevice * device);
extern gboolean gdk_device_get_has_cursor(GdkDevice * device);
extern gboolean gdk_device_get_history(GdkDevice * device,
GdkWindow *,
guint32, guint32, GdkTimeCoord *
**),
gint *);
extern gboolean gdk_device_get_key(GdkDevice * device, guint, guint
*,
GdkModifierType *);
extern GdkInputMode gdk_device_get_mode(GdkDevice * device);
extern gint gdk_device_get_n_axes(GdkDevice * device);
extern gint gdk_device_get_n_keys(GdkDevice * device);
extern const char *gdk_device_get_name(GdkDevice * device);
extern void gdk_device_get_position(GdkDevice * device, GdkScreen
* *,
gint *, gint *);
extern GdkInputSource gdk_device_get_source(GdkDevice * device);
extern void gdk_device_get_state(GdkDevice * device, GdkWindow *,
gdouble *, GdkModifierType *);
extern GType gdk_device_get_type(void);
extern GdkWindow *gdk_device_get_window_at_position(GdkDevice *
device,
gint *, gint *);
extern GdkGrabStatus gdk_device_grab(GdkDevice * device, GdkWindow
*,
GdkGrabOwnership, gboolean,
GdkEventMask, GdkCursor *, guint32);
extern gboolean gdk_device_grab_info_libgtk_only(GdkDisplay *
display,
GdkDevice *,
GdkWindow * *,
gboolean *);
extern GList *gdk_device_list_axes(GdkDevice * device);
extern GList *gdk_device_list_slave_devices(GdkDevice * device);
extern GdkDevice
*gdk_device_manager_get_client_pointer(GdkDeviceManager *
device_manager);
extern GdkDisplay *gdk_device_manager_get_display(GdkDeviceManager
*,
device_manager);
extern GType gdk_device_manager_get_type(void);
extern GList *gdk_device_manager_list_devices(GdkDeviceManager *
device_manager,
GdkDeviceType);
extern void gdk_device_set_axis_use(GdkDevice * device, guint,
GdkAxisUse);
extern void gdk_device_set_key(GdkDevice * device, guint, guint,
GdkModifierType);
extern gboolean gdk_device_set_mode(GdkDevice * device,
GdkInputMode);
extern GType gdk_device_type_get_type(void);
extern void gdk_device_ungrab(GdkDevice * device, guint32);
extern void gdk_device_warp(GdkDevice * device, GdkScreen *, gint,
gint);

```

```

extern void gdk_disable_multidevice(void);
extern void gdk_display_beep(GdkDisplay * display);
extern void gdk_display_close(GdkDisplay * display);
extern gboolean gdk_display_device_is_grabbed(GdkDisplay * display,
                                              GdkDevice *);
extern void gdk_display_flush(GdkDisplay * display);
extern                                     GdkAppLaunchContext
*gdk_display_get_app_launch_context(GdkDisplay *
                                   display);
extern GdkDisplay *gdk_display_get_default(void);
extern guint   gdk_display_get_default_cursor_size(GdkDisplay *
display);
extern   GdkWindow *gdk_display_get_default_group(GdkDisplay *
display);
extern   GdkScreen *gdk_display_get_default_screen(GdkDisplay *
display);
extern   GdkDeviceManager *gdk_display_get_device_manager(GdkDisplay
*
                                   display);
extern   GdkEvent *gdk_display_get_event(GdkDisplay * display);
extern   void   gdk_display_get_maximal_cursor_size(GdkDisplay *
display,
                                   guint *, guint *);
extern   gint   gdk_display_get_n_screens(GdkDisplay * display);
extern   const char *gdk_display_get_name(GdkDisplay * display);
extern   void   gdk_display_get_pointer(GdkDisplay * display, GdkScreen
* *,
                                   gint *, gint *, GdkModifierType *);
extern   GdkScreen *gdk_display_get_screen(GdkDisplay * display,
gint);
extern   GType   gdk_display_get_type(void);
extern   GdkWindow *gdk_display_get_window_at_pointer(GdkDisplay *
display,
                                   gint *, gint *);
extern   gboolean gdk_display_has_pending(GdkDisplay * display);
extern   gboolean gdk_display_is_closed(GdkDisplay * display);
extern   void   gdk_display_keyboard_ungrab(GdkDisplay * display,
guint32);
extern   GList *gdk_display_list_devices(GdkDisplay * display);
extern   GdkDisplayManager *gdk_display_manager_get(void);
extern   GdkDisplay
*gdk_display_manager_get_default_display(GdkDisplayManager *
manager);
extern   GType   gdk_display_manager_get_type(void);
extern   GList *gdk_display_manager_list_displays(GdkDisplayManager
*
                                   manager);
extern                                       GdkDisplay
*gdk_display_manager_open_display(GdkDisplayManager *
                                   manager, const char *);
extern   void   gdk_display_manager_set_default_display(GdkDisplayManager *
                                   manager, GdkDisplay *);
extern   void   gdk_display_notify_startup_complete(GdkDisplay *
display,
                                   const char *);
extern   GdkDisplay *gdk_display_open(const char *display_name);
extern   GdkDisplay *gdk_display_open_default_libgtk_only(void);
extern   GdkEvent *gdk_display_peek_event(GdkDisplay * display);
extern   gboolean   gdk_display_pointer_is_grabbed(GdkDisplay *
display);
extern   void   gdk_display_pointer_ungrab(GdkDisplay * display,
guint32);
extern   void   gdk_display_put_event(GdkDisplay * display,
                                   const GdkEvent * event);

```



```

extern                                     gboolean
gdk_display_request_selection_notification(GdkDisplay *
                                         display,
                                         GdkAtom);

extern void gdk_display_set_double_click_distance(GdkDisplay *
display,
                                         guint);

extern void gdk_display_set_double_click_time(GdkDisplay * display,
guint);
extern void gdk_display_store_clipboard(GdkDisplay * display,
GdkWindow *,
                                         guint32, const GdkAtom *, gint);

extern                                     gboolean
gdk_display_supports_clipboard_persistence(GdkDisplay *
                                         display);

extern gboolean gdk_display_supports_composite(GdkDisplay *
display);
extern gboolean gdk_display_supports_cursor_alpha(GdkDisplay *
display);
extern gboolean gdk_display_supports_cursor_color(GdkDisplay *
display);
extern gboolean gdk_display_supports_input_shapes(GdkDisplay *
display);
extern                                     gboolean
gdk_display_supports_selection_notification(GdkDisplay *
                                         display);

extern gboolean gdk_display_supports_shapes(GdkDisplay * display);
extern void gdk_display_sync(GdkDisplay * display);
extern void gdk_display_warp_pointer(GdkDisplay * display,
GdkScreen *,
                                         gint, gint);

extern void gdk_drag_abort(GdkDragContext * context, guint32);
extern GType gdk_drag_action_get_type(void);
extern GdkDragContext *gdk_drag_begin(GdkWindow * window, GList *);
extern GdkDragContext *gdk_drag_begin_for_device(GdkWindow *
window,
                                         GdkDevice *, GList *);

extern GdkDragAction gdk_drag_context_get_actions(GdkDragContext *
context);
extern GdkWindow *gdk_drag_context_get_dest_window(GdkDragContext
*
context);
extern GdkDevice *gdk_drag_context_get_device(GdkDragContext *
context);
extern                                     GdkDragProtocol
gdk_drag_context_get_protocol(GdkDragContext *
context);
extern                                     GdkDragAction
gdk_drag_context_get_selected_action(GdkDragContext *
context);
extern                                     GdkWindow
*gdk_drag_context_get_source_window(GdkDragContext *
context);
extern                                     GdkDragAction
gdk_drag_context_get_suggested_action(GdkDragContext *
context);
extern GType gdk_drag_context_get_type(void);
extern GList *gdk_drag_context_list_targets(GdkDragContext *
context);
extern void gdk_drag_context_set_device(GdkDragContext * context,
GdkDevice *);
extern void gdk_drag_drop(GdkDragContext * context, guint32);
extern gboolean gdk_drag_drop_succeeded(GdkDragContext * context);
extern void gdk_drag_find_window_for_screen(GdkDragContext *
context,
                                         GdkWindow *, GdkScreen *, gint,

```

```

                                gint, GdkWindow * *,
                                GdkDragProtocol *);
extern GdkAtom gdk_drag_get_selection(GdkDragContext * context);
extern gboolean gdk_drag_motion(GdkDragContext * context, GdkWindow
*,
                                GdkDragProtocol,      gint,      gint,
GdkDragAction,
                                GdkDragAction, guint32);
extern GType gdk_drag_protocol_get_type(void);
extern void gdk_drag_status(GdkDragContext * context, GdkDragAction,
                                guint32);
extern void gdk_drop_finish(GdkDragContext * context, gboolean,
guint32);
extern void gdk_drop_reply(GdkDragContext * context, gboolean,
guint32);
extern gint gdk_error_trap_pop(void);
extern void gdk_error_trap_pop_ignored(void);
extern void gdk_error_trap_push(void);
extern GdkEvent *gdk_event_copy(const GdkEvent * event);
extern void gdk_event_free(GdkEvent * event);
extern GdkEvent *gdk_event_get(void);
extern gboolean gdk_event_get_axis(const GdkEvent * event,
GdkAxisUse,
                                gdouble *);
extern gboolean gdk_event_get_button(const GdkEvent * event, guint
*);
extern gboolean gdk_event_get_click_count(const GdkEvent * event,
guint *);
extern gboolean gdk_event_get_coords(const GdkEvent * event,
gdouble *,
                                gdouble *);
extern GdkDevice *gdk_event_get_device(const GdkEvent * event);
extern GdkEventSequence *gdk_event_get_event_sequence(const
GdkEvent *
                                event);
extern gboolean gdk_event_get_keycode(const GdkEvent * event,
guint16 *);
extern gboolean gdk_event_get_keyval(const GdkEvent * event, guint
*);
extern gboolean gdk_event_get_root_coords(const GdkEvent * event,
gdouble *, gdouble *);
extern GdkScreen *gdk_event_get_screen(const GdkEvent * event);
extern gboolean gdk_event_get_scroll_deltas(const GdkEvent * event,
gdouble *, gdouble *);
extern gboolean gdk_event_get_scroll_direction(const GdkEvent *
event,
                                GdkScrollDirection *);
extern GdkDevice *gdk_event_get_source_device(const GdkEvent *
event);
extern gboolean gdk_event_get_state(const GdkEvent * event,
GdkModifierType *);
extern guint32 gdk_event_get_time(const GdkEvent * event);
extern GType gdk_event_get_type(void);
extern void gdk_event_handler_set(GdkEventFunc func, gpointer,
GDestroyNotify);
extern GType gdk_event_mask_get_type(void);
extern GdkEvent *gdk_event_new(GdkEventType type);
extern GdkEvent *gdk_event_peek(void);
extern void gdk_event_put(const GdkEvent * event);
extern void gdk_event_request_motions(const GdkEventMotion * event);
extern void gdk_event_set_device(GdkEvent * event, GdkDevice *);
extern void gdk_event_set_screen(GdkEvent * event, GdkScreen *);
extern void gdk_event_set_source_device(GdkEvent * event, GdkDevice
*);
extern gboolean gdk_event_triggers_context_menu(const GdkEvent *
event);

```

```

extern GType gdk_event_type_get_type(void);
extern gboolean gdk_events_get_angle(GdkEvent * event1, GdkEvent *,
                                     gdouble *);
extern gboolean gdk_events_get_center(GdkEvent * event1, GdkEvent
*,
                                     gdouble *, gdouble *);
extern gboolean gdk_events_get_distance(GdkEvent * event1, GdkEvent
*,
                                     gdouble *);
extern gboolean gdk_events_pending(void);
extern GType gdk_filter_return_get_type(void);
extern void gdk_flush(void);
extern GdkWindow *gdk_get_default_root_window(void);
extern gchar *gdk_get_display(void);
extern const char *gdk_get_display_arg_name(void);
extern const char *gdk_get_program_class(void);
extern gboolean gdk_get_show_events(void);
extern GType gdk_grab_ownership_get_type(void);
extern GType gdk_grab_status_get_type(void);
extern GType gdk_gravity_get_type(void);
extern void gdk_init(gint * argc, gchar * **);
extern gboolean gdk_init_check(gint * argc, gchar * **);
extern GType gdk_input_mode_get_type(void);
extern GType gdk_input_source_get_type(void);
extern GdkGrabStatus gdk_keyboard_grab(GdkWindow * window, gboolean,
                                       guint32);
extern void gdk_keyboard_ungrab(guint32 time);
extern void gdk_keymap_add_virtual_modifiers(GdkKeymap * keymap,
                                             GdkModifierType *);
extern gboolean gdk_keymap_get_caps_lock_state(GdkKeymap * keymap);
extern GdkKeymap *gdk_keymap_get_default(void);
extern PangoDirection gdk_keymap_get_direction(GdkKeymap * keymap);
extern gboolean gdk_keymap_get_entries_for_keycode(GdkKeymap *
keymap,
                                             guint, GdkKeymapKey * *,
                                             guint * *, gint *);
extern gboolean gdk_keymap_get_entries_for_keyval(GdkKeymap *
keymap,
                                             guint, GdkKeymapKey * *,
                                             gint *);
extern GdkKeymap *gdk_keymap_get_for_display(GdkDisplay * display);
extern GdkModifierType gdk_keymap_get_modifier_mask(GdkKeymap *
keymap,
                                             GdkModifierIntent);
extern guint *gdk_keymap_get_modifier_state(GdkKeymap * keymap);
extern gboolean gdk_keymap_get_num_lock_state(GdkKeymap * keymap);
extern GType gdk_keymap_get_type(void);
extern gboolean gdk_keymap_have_bidi_layouts(GdkKeymap * keymap);
extern guint gdk_keymap_lookup_key(GdkKeymap * keymap,
                                   const GdkKeymapKey *);
extern gboolean gdk_keymap_map_virtual_modifiers(GdkKeymap *
keymap,
                                             GdkModifierType *);
extern gboolean gdk_keymap_translate_keyboard_state(GdkKeymap *
keymap,
                                             guint, GdkModifierType,
                                             gint, guint *, gint *,
                                             gint *,
                                             GdkModifierType *);
extern void gdk_keyval_convert_case(guint symbol, guint *, guint
*);
extern guint gdk_keyval_from_name(const char *keyval_name);
extern gboolean gdk_keyval_is_lower(guint keyval);
extern gboolean gdk_keyval_is_upper(guint keyval);
extern gchar *gdk_keyval_name(guint keyval);
extern guint gdk_keyval_to_lower(guint keyval);

```

```

extern guint32 gdk_keyval_to_unicode(guint keyval);
extern guint gdk_keyval_to_upper(guint keyval);
extern GList *gdk_list_visuals(void);
extern GType gdk_modifier_intent_get_type(void);
extern GType gdk_modifier_type_get_type(void);
extern void gdk_notify_startup_complete(void);
extern void gdk_notify_startup_complete_with_id(const char
*startup_id);
extern GType gdk_notify_type_get_type(void);
extern GdkWindow *gdk_offscreen_window_get_embedder(GdkWindow *
window);
extern cairo_surface_t *gdk_offscreen_window_get_surface(GdkWindow
*
window);
extern void gdk_offscreen_window_set_embedder(GdkWindow * window,
GdkWindow *);
extern GType gdk_owner_change_get_type(void);
extern PangoContext *gdk_pango_context_get(void);
extern PangoContext *gdk_pango_context_get_for_screen(GdkScreen *
screen);
extern cairo_region_t
*gdk_pango_layout_get_clip_region(PangoLayout *
layout, gint, gint,
const int *, gint);
extern cairo_region_t
*gdk_pango_layout_line_get_clip_region(PangoLayoutLine * line,
gint,
gint, const int *, gint);
extern void gdk_parse_args(gint * argc, gchar * **);
extern GdkPixbuf *gdk_pixbuf_get_from_surface(cairo_surface_t *
surface,
gint, gint, gint, gint);
extern GdkPixbuf *gdk_pixbuf_get_from_window(GdkWindow * window,
gint,
gint, gint, gint);
extern GdkGrabStatus gdk_pointer_grab(GdkWindow * window, gboolean,
GdkEventMask, GdkWindow *,
GdkCursor *, guint32);
extern gboolean gdk_pointer_is_grabbed(void);
extern void gdk_pointer_ungrab(guint32 time_);
extern void gdk_pre_parse_libgtk_only(void);
extern GType gdk_prop_mode_get_type(void);
extern void gdk_property_change(GdkWindow * window, GdkAtom,
GdkAtom, gint,
GdkPropMode, const unsigned char *,
gint);
extern void gdk_property_delete(GdkWindow * window, GdkAtom);
extern gboolean gdk_property_get(GdkWindow * window, GdkAtom,
GdkAtom,
gulong, gulong, gint, GdkAtom *, gint
*,
gint *, gchar * *);
extern GType gdk_property_state_get_type(void);
extern void gdk_query_depths(gint * *depths, gint *);
extern void gdk_query_visual_types(GdkVisualType * *visual_types,
gint *);
extern GType gdk_rectangle_get_type(void);
extern gboolean gdk_rectangle_intersect(const GdkRectangle * src1,
const GdkRectangle *,
GdkRectangle *);
extern void gdk_rectangle_union(const GdkRectangle * src1,
const GdkRectangle *, GdkRectangle *);
extern GdkRGBA *gdk_rgba_copy(const GdkRGBA * rgba);
extern gboolean gdk_rgba_equal(gconstpointer p1, gconstpointer);
extern void gdk_rgba_free(GdkRGBA * rgba);
extern GType gdk_rgba_get_type(void);

```

```

extern guint gdk_rgba_hash(gconstpointer p);
extern gboolean gdk_rgba_parse(GdkRGBA * rgba, const char *);
extern gchar *gdk_rgba_to_string(const GdkRGBA * rgba);
extern GdkWindow *gdk_screen_get_active_window(GdkScreen * screen);
extern GdkScreen *gdk_screen_get_default(void);
extern GdkDisplay *gdk_screen_get_display(GdkScreen * screen);
extern          const          cairo_font_options_t
*gdk_screen_get_font_options(GdkScreen *
                                screen);
extern gint gdk_screen_get_height(GdkScreen * screen);
extern gint gdk_screen_get_height_mm(GdkScreen * screen);
extern gint gdk_screen_get_monitor_at_point(GdkScreen * screen,
gint,
                                gint);
extern gint gdk_screen_get_monitor_at_window(GdkScreen * screen,
                                GdkWindow *);
extern void gdk_screen_get_monitor_geometry(GdkScreen * screen,
gint,
                                GdkRectangle *);
extern gint gdk_screen_get_monitor_height_mm(GdkScreen * screen,
gint);
extern gchar *gdk_screen_get_monitor_plug_name(GdkScreen * screen,
gint);
extern gint gdk_screen_get_monitor_width_mm(GdkScreen * screen,
gint);
extern void gdk_screen_get_monitor_workarea(GdkScreen * screen,
gint,
                                GdkRectangle *);
extern gint gdk_screen_get_n_monitors(GdkScreen * screen);
extern gint gdk_screen_get_number(GdkScreen * screen);
extern gint gdk_screen_get_primary_monitor(GdkScreen * screen);
extern gdouble gdk_screen_get_resolution(GdkScreen * screen);
extern GdkVisual *gdk_screen_get_rgba_visual(GdkScreen * screen);
extern GdkWindow *gdk_screen_get_root_window(GdkScreen * screen);
extern gboolean gdk_screen_get_setting(GdkScreen * screen, const
char *,
                                GValue *);
extern GdkVisual *gdk_screen_get_system_visual(GdkScreen * screen);
extern GList *gdk_screen_get_toplevel_windows(GdkScreen * screen);
extern GType gdk_screen_get_type(void);
extern gint gdk_screen_get_width(GdkScreen * screen);
extern gint gdk_screen_get_width_mm(GdkScreen * screen);
extern GList *gdk_screen_get_window_stack(GdkScreen * screen);
extern gint gdk_screen_height(void);
extern gint gdk_screen_height_mm(void);
extern gboolean gdk_screen_is_composited(GdkScreen * screen);
extern GList *gdk_screen_list_visuals(GdkScreen * screen);
extern gchar *gdk_screen_make_display_name(GdkScreen * screen);
extern void gdk_screen_set_font_options(GdkScreen * screen,
                                const cairo_font_options_t *);
extern void gdk_screen_set_resolution(GdkScreen * screen, gdouble);
extern gint gdk_screen_width(void);
extern gint gdk_screen_width_mm(void);
extern GType gdk_scroll_direction_get_type(void);
extern void gdk_selection_convert(GdkWindow * requestor, GdkAtom,
GdkAtom,
                                guint32);
extern GdkWindow *gdk_selection_owner_get(GdkAtom selection);
extern GdkWindow *gdk_selection_owner_get_for_display(GdkDisplay *
display,
                                GdkAtom);
extern gboolean gdk_selection_owner_set(GdkWindow * owner, GdkAtom,
guint32, gboolean);
extern gboolean gdk_selection_owner_set_for_display(GdkDisplay *
display,
                                GdkWindow *, GdkAtom,

```

```

                                guint32, gboolean);
extern gint gdk_selection_property_get(GdkWindow * requestor,
guchar * *,
                                GdkAtom *, gint *);
extern void gdk_selection_send_notify(GdkWindow * requestor,
GdkAtom,
                                GdkAtom, GdkAtom, guint32);
extern void gdk_selection_send_notify_for_display(GdkDisplay *
display,
                                GdkWindow *, GdkAtom,
                                GdkAtom, GdkAtom,
                                guint32);
extern void gdk_set_double_click_time(guint msec);
extern void gdk_set_program_class(const char *program_class);
extern void gdk_set_show_events(gboolean show_events);
extern GType gdk_setting_action_get_type(void);
extern gboolean gdk_setting_get(const char *name, GValue *);
extern GType gdk_status_get_type(void);
extern void gdk_test_render_sync(GdkWindow * window);
extern gboolean gdk_test_simulate_button(GdkWindow * window, gint,
gint,
                                guint, GdkModifierType,
                                GdkEventType);
extern gboolean gdk_test_simulate_key(GdkWindow * window, gint,
gint,
                                guint, GdkModifierType,
                                GdkEventType);
extern gint gdk_text_property_to_utf8_list_for_display(GdkDisplay
*
                                display, GdkAtom,
                                gint,
                                const unsigned char
                                *, gint,
                                gchar * **);
extern guint gdk_threads_add_idle(GSourceFunc function, gpointer);
extern guint gdk_threads_add_idle_full(gint priority, GSourceFunc,
gpointer, GDestroyNotify);
extern guint gdk_threads_add_timeout(guint interval, GSourceFunc,
gpointer);
extern guint gdk_threads_add_timeout_full(gint priority, guint,
GSourceFunc, gpointer,
GDestroyNotify);
extern guint gdk_threads_add_timeout_seconds(guint interval,
GSourceFunc,
gpointer);
extern guint gdk_threads_add_timeout_seconds_full(gint priority,
gint,
GSourceFunc, gpointer,
GDestroyNotify);
extern void gdk_threads_enter(void);
extern void gdk_threads_init(void);
extern void gdk_threads_leave(void);
extern void gdk_threads_set_lock_functions(GCallback enter_fn,
GCallback);
extern guint gdk_unicode_to_keyval(guint32 wc);
extern gchar *gdk_utf8_to_string_target(const char *str);
extern GType gdk_visibility_state_get_type(void);
extern GdkVisual *gdk_visual_get_best(void);
extern gint gdk_visual_get_best_depth(void);
extern GdkVisualType gdk_visual_get_best_type(void);
extern GdkVisual *gdk_visual_get_best_with_both(gint depth,
GdkVisualType);
extern GdkVisual *gdk_visual_get_best_with_depth(gint depth);
extern GdkVisual *gdk_visual_get_best_with_type(GdkVisualType
visual_type);
extern gint gdk_visual_get_bits_per_rgb(GdkVisual * visual);

```

```

extern void gdk_visual_get_blue_pixel_details(GdkVisual * visual,
                                              guint32 *, gint *, gint *);
extern GdkByteOrder gdk_visual_get_byte_order(GdkVisual * visual);
extern gint gdk_visual_get_colormap_size(GdkVisual * visual);
extern gint gdk_visual_get_depth(GdkVisual * visual);
extern void gdk_visual_get_green_pixel_details(GdkVisual * visual,
                                              guint32 *, gint *, gint *);
extern void gdk_visual_get_red_pixel_details(GdkVisual * visual,
                                              guint32 *,
                                              gint *, gint *);
extern GdkScreen *gdk_visual_get_screen(GdkVisual * visual);
extern GdkVisual *gdk_visual_get_system(void);
extern GType gdk_visual_get_type(void);
extern GdkVisualType gdk_visual_get_visual_type(GdkVisual *
visual);
extern GType gdk_visual_type_get_type(void);
extern void gdk_window_add_filter(GdkWindow * window, GdkFilterFunc,
gpointer);
extern GdkWindow *gdk_window_at_pointer(gint * win_x, gint * win_y);
extern GType gdk_window_attributes_type_get_type(void);
extern void gdk_window_beep(GdkWindow * window);
extern void gdk_window_begin_move_drag(GdkWindow * window, gint,
gint,
gint, guint32);
extern void gdk_window_begin_move_drag_for_device(GdkWindow *
window,
GdkDevice *, gint, gint,
gint, guint32);
extern void gdk_window_begin_paint_rect(GdkWindow * window,
const GdkRectangle *);
extern void gdk_window_begin_paint_region(GdkWindow * window,
const cairo_region_t);
extern void gdk_window_begin_resize_drag(GdkWindow * window,
GdkWindowEdge,
gint, gint, gint, guint32);
extern void gdk_window_begin_resize_drag_for_device(GdkWindow *
window,
GdkWindowEdge,
GdkDevice *, gint,
gint, gint, guint32);
extern void gdk_window_configure_finished(GdkWindow * window);
extern void gdk_window_constrain_size(GdkGeometry * geometry, guint,
gint,
gint, gint *, gint *);
extern void gdk_window_coords_from_parent(GdkWindow * window,
gdouble,
gdouble, gdouble *, gdouble *);
extern void gdk_window_coords_to_parent(GdkWindow * window, gdouble,
gdouble, gdouble *, gdouble *);
extern void gdk_window_create_similar_surface(GdkWindow *
window,
cairo_content_t,
int, int);
extern void gdk_window_deiconify(GdkWindow * window);
extern void gdk_window_destroy(GdkWindow * window);
extern GType gdk_window_edge_get_type(void);
extern void gdk_window_enable_synchronized_configure(GdkWindow *
window);
extern void gdk_window_end_paint(GdkWindow * window);
extern gboolean gdk_window_ensure_native(GdkWindow * window);
extern void gdk_window_flush(GdkWindow * window);
extern void gdk_window_focus(GdkWindow * window, guint32);
extern void gdk_window_freeze_toplevel_updates_libgtk_only(GdkWindow *
window);

```



```

extern void gdk_window_freeze_updates(GdkWindow * window);
extern void gdk_window_fullscreen(GdkWindow * window);
extern void gdk_window_geometry_changed(GdkWindow * window);
extern gboolean gdk_window_get_accept_focus(GdkWindow * window);
extern          cairo_pattern_t
*gdk_window_get_background_pattern(GdkWindow *
                                   window);
extern GList *gdk_window_get_children(GdkWindow * window);
extern cairo_region_t *gdk_window_get_clip_region(GdkWindow *
window);
extern gboolean gdk_window_get_composited(GdkWindow * window);
extern GdkCursor *gdk_window_get_cursor(GdkWindow * window);
extern gboolean gdk_window_get_decorations(GdkWindow * window,
                                           GdkWmDecoration *);
extern GdkCursor *gdk_window_get_device_cursor(GdkWindow * window,
                                                GdkDevice *);
extern GdkEventMask gdk_window_get_device_events(GdkWindow *
window,
                                                  GdkDevice *);
extern GdkWindow *gdk_window_get_device_position(GdkWindow *
window,
                                                  GdkDevice *, gint *,
                                                  gint *,
                                                  GdkModifierType *);
extern GdkDisplay *gdk_window_get_display(GdkWindow * window);
extern GdkDragProtocol gdk_window_get_drag_protocol(GdkWindow *
window,
                                                    GdkWindow * *);
extern GdkWindow *gdk_window_get_effective_parent(GdkWindow *
window);
extern GdkWindow *gdk_window_get_effective_toplevel(GdkWindow *
window);
extern GdkEventMask gdk_window_get_events(GdkWindow * window);
extern gboolean gdk_window_get_focus_on_map(GdkWindow * window);
extern void gdk_window_get_frame_extents(GdkWindow * window,
                                          GdkRectangle *);
extern void gdk_window_get_geometry(GdkWindow * window, gint *,
gint *,
                                gint *, gint *);
extern GdkWindow *gdk_window_get_group(GdkWindow * window);
extern int gdk_window_get_height(GdkWindow * window);
extern gboolean gdk_window_get_modal_hint(GdkWindow * window);
extern gint gdk_window_get_origin(GdkWindow * window, gint *, gint
*);
extern GdkWindow *gdk_window_get_parent(GdkWindow * window);
extern GdkWindow *gdk_window_get_pointer(GdkWindow * window, gint
*,
                                         gint *, GdkModifierType *);
extern void gdk_window_get_position(GdkWindow * window, gint *,
gint *);
extern void gdk_window_get_root_coords(GdkWindow * window, gint,
gint,
                                gint *, gint *);
extern void gdk_window_get_root_origin(GdkWindow * window, gint *,
gint *);
extern GdkScreen *gdk_window_get_screen(GdkWindow * window);
extern GdkEventMask gdk_window_get_source_events(GdkWindow *
window,
                                                  GdkInputSource);
extern GdkWindowState gdk_window_get_state(GdkWindow * window);
extern gboolean gdk_window_get_support_multidevice(GdkWindow *
window);
extern GdkWindow *gdk_window_get_toplevel(GdkWindow * window);
extern GType gdk_window_get_type(void);
extern GdkWindowTypeHint gdk_window_get_type_hint(GdkWindow *
window);

```



```

extern cairo_region_t *gdk_window_get_update_area(GdkWindow *
window);
extern void gdk_window_get_user_data(GdkWindow * window, void **);
extern cairo_region_t *gdk_window_get_visible_region(GdkWindow *
window);
extern GdkVisual *gdk_window_get_visual(GdkWindow * window);
extern int gdk_window_get_width(GdkWindow * window);
extern GdkWindowType gdk_window_get_window_type(GdkWindow *
window);
extern gboolean gdk_window_has_native(GdkWindow * window);
extern void gdk_window_hide(GdkWindow * window);
extern GType gdk_window_hints_get_type(void);
extern void gdk_window_iconify(GdkWindow * window);
extern void gdk_window_input_shape_combine_region(GdkWindow *
window,
                                const cairo_region_t,
                                gint, gint);
extern void gdk_window_invalidate_maybe_recurse(GdkWindow * window,
                                const cairo_region_t,
                                GdkWindowChildFunc,
                                void *);
extern void gdk_window_invalidate_rect(GdkWindow * window,
                                const GdkRectangle *, gboolean);
extern void gdk_window_invalidate_region(GdkWindow * window,
                                const cairo_region_t, gboolean);
extern gboolean gdk_window_is_destroyed(GdkWindow * window);
extern gboolean gdk_window_is_input_only(GdkWindow * window);
extern gboolean gdk_window_is_shaped(GdkWindow * window);
extern gboolean gdk_window_is_viewable(GdkWindow * window);
extern gboolean gdk_window_is_visible(GdkWindow * window);
extern void gdk_window_lower(GdkWindow * window);
extern void gdk_window_maximize(GdkWindow * window);
extern void gdk_window_merge_child_input_shapes(GdkWindow *
window);
extern void gdk_window_merge_child_shapes(GdkWindow * window);
extern void gdk_window_move(GdkWindow * window, gint, gint);
extern void gdk_window_move_region(GdkWindow * window,
                                const cairo_region_t, gint, gint);
extern void gdk_window_move_resize(GdkWindow * window, gint, gint,
gint,
                                gint);
extern GdkWindow *gdk_window_new(GdkWindow * parent, GdkWindowAttr
*,
                                gint);
extern GList *gdk_window_peek_children(GdkWindow * window);
extern void gdk_window_process_all_updates(void);
extern void gdk_window_process_updates(GdkWindow * window,
gboolean);
extern void gdk_window_raise(GdkWindow * window);
extern void gdk_window_register_dnd(GdkWindow * window);
extern void gdk_window_remove_filter(GdkWindow * window,
GdkFilterFunc,
                                gpointer);
extern void gdk_window_reparent(GdkWindow * window, GdkWindow *,
gint,
                                gint);
extern void gdk_window_resize(GdkWindow * window, gint, gint);
extern void gdk_window_restack(GdkWindow * window, GdkWindow *,
gboolean);
extern void gdk_window_scroll(GdkWindow * window, gint, gint);
extern void gdk_window_set_accept_focus(GdkWindow * window,
gboolean);
extern void gdk_window_set_background(GdkWindow * window,
                                const GdkColor *);
extern void gdk_window_set_background_pattern(GdkWindow * window,
                                cairo_pattern_t *);

```

```

extern void gdk_window_set_background_rgba(GdkWindow * window,
GdkRGBA *);
extern void gdk_window_set_child_input_shapes(GdkWindow * window);
extern void gdk_window_set_child_shapes(GdkWindow * window);
extern void gdk_window_set_composited(GdkWindow * window, gboolean);
extern void gdk_window_set_cursor(GdkWindow * window, GdkCursor *);
extern void gdk_window_set_debug_updates(gboolean setting);
extern void gdk_window_set_decorations(GdkWindow * window,
GdkWMDecoration);
extern void gdk_window_set_device_cursor(GdkWindow * window,
GdkDevice *,
GdkCursor *);
extern void gdk_window_set_device_events(GdkWindow * window,
GdkDevice *,
GdkEventMask);
extern void gdk_window_set_events(GdkWindow * window, GdkEventMask);
extern void gdk_window_set_focus_on_map(GdkWindow * window,
gboolean);
extern void gdk_window_set_functions(GdkWindow * window,
GdkWMFunction);
extern void gdk_window_set_geometry_hints(GdkWindow * window,
const GdkGeometry *,
GdkWindowHints);
extern void gdk_window_set_group(GdkWindow * window, GdkWindow *);
extern void gdk_window_set_icon_list(GdkWindow * window, GList *);
extern void gdk_window_set_icon_name(GdkWindow * window, const char
*);
extern void gdk_window_set_keep_above(GdkWindow * window, gboolean);
extern void gdk_window_set_keep_below(GdkWindow * window, gboolean);
extern void gdk_window_set_modal_hint(GdkWindow * window, gboolean);
extern void gdk_window_set_opacity(GdkWindow * window, gdouble);
extern void gdk_window_set_override_redirect(GdkWindow * window,
gboolean);
extern void gdk_window_set_role(GdkWindow * window, const char *);
extern void gdk_window_set_skip_pager_hint(GdkWindow * window,
gboolean);
extern void gdk_window_set_skip_taskbar_hint(GdkWindow * window,
gboolean);
extern void gdk_window_set_source_events(GdkWindow * window,
GdkInputSource, GdkEventMask);
extern void gdk_window_set_startup_id(GdkWindow * window, const
char *);
extern gboolean gdk_window_set_static_gravities(GdkWindow * window,
gboolean);
extern void gdk_window_set_support_multidevice(GdkWindow * window,
gboolean);
extern void gdk_window_set_title(GdkWindow * window, const char *);
extern void gdk_window_set_transient_for(GdkWindow * window,
GdkWindow *);
extern void gdk_window_set_type_hint(GdkWindow * window,
GdkWindowTypeHint);
extern void gdk_window_set_urgency_hint(GdkWindow * window,
gboolean);
extern void gdk_window_set_user_data(GdkWindow * window, gpointer);
extern void gdk_window_shape_combine_region(GdkWindow * window,
const cairo_region_t, gint,
gint);
extern void gdk_window_show(GdkWindow * window);
extern void gdk_window_show_unraised(GdkWindow * window);
extern GType gdk_window_state_get_type(void);
extern void gdk_window_stick(GdkWindow * window);
extern void gdk_window_thaw_toplevel_updates_libgtk_only(GdkWindow
*
window);
extern void gdk_window_thaw_updates(GdkWindow * window);
extern GType gdk_window_type_get_type(void);

```

```

extern GType gdk_window_type_hint_get_type(void);
extern void gdk_window_unfullscreen(GdkWindow * window);
extern void gdk_window_unmaximize(GdkWindow * window);
extern void gdk_window_unstick(GdkWindow * window);
extern GType gdk_window_window_class_get_type(void);
extern void gdk_window_withdraw(GdkWindow * window);
extern GType gdk_wm_decoration_get_type(void);
extern GType gdk_wm_function_get_type(void);

```

7.3.2 gtk-3.0/gdk/gdkbroadway.h

```

#define __GDKBROADWAY_H_INSIDE__
#define GDK_BROADWAY_CURSOR_H__
#define GDK_BROADWAY_DISPLAY_MANAGER_H__
#define GDK_BROADWAY_H__
#define GDK_BROADWAY_VISUAL_H__
#define GDK_BROADWAY_WINDOW_H__
#define GDK_TYPE_BROADWAY_CURSOR
(gdk_broadway_cursor_get_type ())
#define GDK_TYPE_BROADWAY_DISPLAY_MANAGER
(gdk_broadway_display_manager_get_type ())
#define GDK_TYPE_BROADWAY_VISUAL
(gdk_broadway_visual_get_type ())
#define GDK_TYPE_BROADWAY_WINDOW
(gdk_broadway_window_get_type ())
#define GDK_BROADWAY_CURSOR_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_BROADWAY_CURSOR,
GdkBroadwayCursorClass))
#define GDK_BROADWAY_VISUAL_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_BROADWAY_VISUAL,
GdkBroadwayVisualClass))
#define GDK_BROADWAY_WINDOW_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_BROADWAY_WINDOW,
GdkBroadwayWindowClass))
#define GDK_IS_BROADWAY_CURSOR_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GDK_TYPE_BROADWAY_CURSOR))
#define GDK_IS_BROADWAY_VISUAL_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GDK_TYPE_BROADWAY_VISUAL))
#define GDK_IS_BROADWAY_WINDOW_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GDK_TYPE_BROADWAY_WINDOW))
#define GDK_BROADWAY_CURSOR(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object), GDK_TYPE_BROADWAY_CURSOR,
GdkBroadwayCursor))
#define GDK_BROADWAY_DISPLAY_MANAGER(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object),
GDK_TYPE_BROADWAY_DISPLAY_MANAGER, GdkBroadwayDisplayManager))
#define GDK_BROADWAY_VISUAL(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object), GDK_TYPE_BROADWAY_VISUAL,
GdkBroadwayVisual))
#define GDK_BROADWAY_WINDOW(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object), GDK_TYPE_BROADWAY_WINDOW,
GdkBroadwayWindow))
#define GDK_IS_BROADWAY_CURSOR(object)
(G_TYPE_CHECK_INSTANCE_TYPE ((object), GDK_TYPE_BROADWAY_CURSOR))
#define GDK_IS_BROADWAY_VISUAL(object)
(G_TYPE_CHECK_INSTANCE_TYPE ((object), GDK_TYPE_BROADWAY_VISUAL))
#define GDK_IS_BROADWAY_WINDOW(object)
(G_TYPE_CHECK_INSTANCE_TYPE ((object), GDK_TYPE_BROADWAY_WINDOW))
#define GDK_BROADWAY_CURSOR_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GDK_TYPE_BROADWAY_CURSOR,
GdkBroadwayCursorClass))
#define GDK_BROADWAY_VISUAL_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GDK_TYPE_BROADWAY_VISUAL,
GdkBroadwayVisualClass))

```

```

#define GDK_BROADWAY_WINDOW_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GDK_TYPE_BROADWAY_WINDOW,
GdkBroadwayWindowClass))

extern GType gdk_broadway_cursor_get_type(void);
extern GType gdk_broadway_display_manager_get_type(void);
extern guint32 gdk_broadway_get_last_seen_time(GdkWindow * window);
extern GType gdk_broadway_visual_get_type(void);
extern GType gdk_broadway_window_get_type(void);

```

7.3.3 gtk-3.0/gdk/gdkkeysyms-compat.h

```

#define __GDK_KEYSyms_COMPAT_H__
#define GDK_Oslash 0x0d8
#define GDK_ooblique 0x0f8
#define GDK_Ibreve 0x100012c
#define GDK_ibreve 0x100012d
#define GDK_Wcircumflex 0x1000174
#define GDK_wcircumflex 0x1000175
#define GDK_Ycircumflex 0x1000176
#define GDK_ycircumflex 0x1000177
#define GDK_SCHWA 0x100018f
#define GDK_Obarred 0x100019f
#define GDK_Ohorn 0x10001a0
#define GDK_ohorn 0x10001a1
#define GDK_Uhorn 0x10001af
#define GDK_uhorn 0x10001b0
#define GDK_Zstroke 0x10001b5
#define GDK_zstroke 0x10001b6
#define GDK_EZH 0x10001b7
#define GDK_Ocaron 0x10001d1
#define GDK_ocaron 0x10001d2
#define GDK_Gcaron 0x10001e6
#define GDK_gcaron 0x10001e7
#define GDK_schwa 0x1000259
#define GDK_obarred 0x1000275
#define GDK_ezh 0x1000292
#define GDK_Cyrillic_GHE_bar 0x1000492
#define GDK_Cyrillic_ghe_bar 0x1000493
#define GDK_Cyrillic_ZHE_descender 0x1000496
#define GDK_Cyrillic_zhe_descender 0x1000497
#define GDK_Cyrillic_KA_descender 0x100049a
#define GDK_Cyrillic_ka_descender 0x100049b
#define GDK_Cyrillic_KA_vertstroke 0x100049c
#define GDK_Cyrillic_ka_vertstroke 0x100049d
#define GDK_Cyrillic_EN_descender 0x10004a2
#define GDK_Cyrillic_en_descender 0x10004a3
#define GDK_Cyrillic_U_straight 0x10004ae
#define GDK_Cyrillic_u_straight 0x10004af
#define GDK_Cyrillic_U_straight_bar 0x10004b0
#define GDK_Cyrillic_u_straight_bar 0x10004b1
#define GDK_Cyrillic_HA_descender 0x10004b2
#define GDK_Cyrillic_ha_descender 0x10004b3
#define GDK_Cyrillic_CHE_descender 0x10004b6
#define GDK_Cyrillic_che_descender 0x10004b7
#define GDK_Cyrillic_CHE_vertstroke 0x10004b8
#define GDK_Cyrillic_che_vertstroke 0x10004b9
#define GDK_Cyrillic_SHHA 0x10004ba
#define GDK_Cyrillic_shha 0x10004bb
#define GDK_Cyrillic_SCHWA 0x10004d8
#define GDK_Cyrillic_schwa 0x10004d9
#define GDK_Cyrillic_I_macron 0x10004e2
#define GDK_Cyrillic_i_macron 0x10004e3
#define GDK_Cyrillic_O_bar 0x10004e8
#define GDK_Cyrillic_o_bar 0x10004e9

```

```

#define GDK_Cyrillic_U_macron 0x10004ee
#define GDK_Cyrillic_u_macron 0x10004ef
#define GDK_Armenian_AYB 0x1000531
#define GDK_Armenian_BEN 0x1000532
#define GDK_Armenian_GIM 0x1000533
#define GDK_Armenian_DA 0x1000534
#define GDK_Armenian_YECH 0x1000535
#define GDK_Armenian_ZA 0x1000536
#define GDK_Armenian_E 0x1000537
#define GDK_Armenian_AT 0x1000538
#define GDK_Armenian_TO 0x1000539
#define GDK_Armenian_ZHE 0x100053a
#define GDK_Armenian_INI 0x100053b
#define GDK_Armenian_LYUN 0x100053c
#define GDK_Armenian_KHE 0x100053d
#define GDK_Armenian_TSA 0x100053e
#define GDK_Armenian_KEN 0x100053f
#define GDK_Armenian_HO 0x1000540
#define GDK_Armenian_DZA 0x1000541
#define GDK_Armenian_GHAT 0x1000542
#define GDK_Armenian_TCHE 0x1000543
#define GDK_Armenian_MEN 0x1000544
#define GDK_Armenian_HI 0x1000545
#define GDK_Armenian_NU 0x1000546
#define GDK_Armenian_SHA 0x1000547
#define GDK_Armenian_VO 0x1000548
#define GDK_Armenian_CHA 0x1000549
#define GDK_Armenian_PE 0x100054a
#define GDK_Armenian_JE 0x100054b
#define GDK_Armenian_RA 0x100054c
#define GDK_Armenian_SE 0x100054d
#define GDK_Armenian_VEV 0x100054e
#define GDK_Armenian_TYUN 0x100054f
#define GDK_Armenian_RE 0x1000550
#define GDK_Armenian_TSO 0x1000551
#define GDK_Armenian_VYUN 0x1000552
#define GDK_Armenian_PYUR 0x1000553
#define GDK_Armenian_KE 0x1000554
#define GDK_Armenian_O 0x1000555
#define GDK_Armenian_FE 0x1000556
#define GDK_Armenian_apostrophe 0x100055a
#define GDK_Armenian_accent 0x100055b
#define GDK_Armenian_shesht 0x100055b
#define GDK_Armenian_amanak 0x100055c
#define GDK_Armenian_exclam 0x100055c
#define GDK_Armenian_but 0x100055d
#define GDK_Armenian_separation_mark 0x100055d
#define GDK_Armenian_paruyk 0x100055e
#define GDK_Armenian_question 0x100055e
#define GDK_Armenian_ayb 0x1000561
#define GDK_Armenian_ben 0x1000562
#define GDK_Armenian_gim 0x1000563
#define GDK_Armenian_da 0x1000564
#define GDK_Armenian_yech 0x1000565
#define GDK_Armenian_za 0x1000566
#define GDK_Armenian_e 0x1000567
#define GDK_Armenian_at 0x1000568
#define GDK_Armenian_to 0x1000569
#define GDK_Armenian_zhe 0x100056a
#define GDK_Armenian_ini 0x100056b
#define GDK_Armenian_lyun 0x100056c
#define GDK_Armenian_khe 0x100056d
#define GDK_Armenian_tsa 0x100056e
#define GDK_Armenian_ken 0x100056f
#define GDK_Armenian_ho 0x1000570
#define GDK_Armenian_dza 0x1000571

```

```

#define GDK_Armenian_ghat      0x1000572
#define GDK_Armenian_tche      0x1000573
#define GDK_Armenian_men      0x1000574
#define GDK_Armenian_hi 0x1000575
#define GDK_Armenian_nu 0x1000576
#define GDK_Armenian_sha      0x1000577
#define GDK_Armenian_vo 0x1000578
#define GDK_Armenian_cha      0x1000579
#define GDK_Armenian_pe 0x100057a
#define GDK_Armenian_je 0x100057b
#define GDK_Armenian_ra 0x100057c
#define GDK_Armenian_se 0x100057d
#define GDK_Armenian_vev      0x100057e
#define GDK_Armenian_tyun      0x100057f
#define GDK_Armenian_re 0x1000580
#define GDK_Armenian_tso      0x1000581
#define GDK_Armenian_vyun      0x1000582
#define GDK_Armenian_pyur      0x1000583
#define GDK_Armenian_ke 0x1000584
#define GDK_Armenian_o 0x1000585
#define GDK_Armenian_fe 0x1000586
#define GDK_Armenian_ligature_ew      0x1000587
#define GDK_Armenian_full_stop 0x1000589
#define GDK_Armenian_verjaket 0x1000589
#define GDK_Armenian_hyphen 0x100058a
#define GDK_Armenian_yentamna 0x100058a
#define GDK_Arabic_madda_above 0x1000653
#define GDK_Arabic_hamza_above 0x1000654
#define GDK_Arabic_hamza_below 0x1000655
#define GDK_Arabic_0 0x1000660
#define GDK_Arabic_1 0x1000661
#define GDK_Arabic_2 0x1000662
#define GDK_Arabic_3 0x1000663
#define GDK_Arabic_4 0x1000664
#define GDK_Arabic_5 0x1000665
#define GDK_Arabic_6 0x1000666
#define GDK_Arabic_7 0x1000667
#define GDK_Arabic_8 0x1000668
#define GDK_Arabic_9 0x1000669
#define GDK_Arabic_percent 0x100066a
#define GDK_Arabic_superscript_alef      0x1000670
#define GDK_Arabic_tteh 0x1000679
#define GDK_Arabic_peh 0x100067e
#define GDK_Arabic_tcheh      0x1000686
#define GDK_Arabic_ddal 0x1000688
#define GDK_Arabic_rreh 0x1000691
#define GDK_Arabic_jeh 0x1000698
#define GDK_Arabic_veh 0x10006a4
#define GDK_Arabic_keheh      0x10006a9
#define GDK_Arabic_gaf 0x10006af
#define GDK_Arabic_noon_ghunna 0x10006ba
#define GDK_Arabic_heh_doachashmee      0x10006be
#define GDK_Arabic_heh_goal 0x10006c1
#define GDK_Arabic_farsi_yeh 0x10006cc
#define GDK_Farsi_yeh 0x10006cc
#define GDK_Arabic_yeh_baree 0x10006d2
#define GDK_Arabic_fullstop 0x10006d4
#define GDK_Farsi_0 0x10006f0
#define GDK_Farsi_1 0x10006f1
#define GDK_Farsi_2 0x10006f2
#define GDK_Farsi_3 0x10006f3
#define GDK_Farsi_4 0x10006f4
#define GDK_Farsi_5 0x10006f5
#define GDK_Farsi_6 0x10006f6
#define GDK_Farsi_7 0x10006f7
#define GDK_Farsi_8 0x10006f8

```



```

#define GDK_Farsi_9      0x10006f9
#define GDK_Sinh_ng      0x1000d82
#define GDK_Sinh_h2      0x1000d83
#define GDK_Sinh_a       0x1000d85
#define GDK_Sinh_aa      0x1000d86
#define GDK_Sinh_ae      0x1000d87
#define GDK_Sinh_aee     0x1000d88
#define GDK_Sinh_i       0x1000d89
#define GDK_Sinh_ii      0x1000d8a
#define GDK_Sinh_u       0x1000d8b
#define GDK_Sinh_uu      0x1000d8c
#define GDK_Sinh_ri      0x1000d8d
#define GDK_Sinh_rii     0x1000d8e
#define GDK_Sinh_lu      0x1000d8f
#define GDK_Sinh_luu     0x1000d90
#define GDK_Sinh_e       0x1000d91
#define GDK_Sinh_ee      0x1000d92
#define GDK_Sinh_ai      0x1000d93
#define GDK_Sinh_o       0x1000d94
#define GDK_Sinh_oo      0x1000d95
#define GDK_Sinh_au      0x1000d96
#define GDK_Sinh_ka      0x1000d9a
#define GDK_Sinh_kha     0x1000d9b
#define GDK_Sinh_ga      0x1000d9c
#define GDK_Sinh_gha     0x1000d9d
#define GDK_Sinh_ng2     0x1000d9e
#define GDK_Sinh_ng      0x1000d9f
#define GDK_Sinh_ca      0x1000da0
#define GDK_Sinh_cha     0x1000da1
#define GDK_Sinh_ja      0x1000da2
#define GDK_Sinh_jha     0x1000da3
#define GDK_Sinh_nya     0x1000da4
#define GDK_Sinh_jnya    0x1000da5
#define GDK_Sinh_nja     0x1000da6
#define GDK_Sinh_tta     0x1000da7
#define GDK_Sinh_ttha    0x1000da8
#define GDK_Sinh_dda     0x1000da9
#define GDK_Sinh_ddha    0x1000daa
#define GDK_Sinh_nna     0x1000dab
#define GDK_Sinh_ndda    0x1000dac
#define GDK_Sinh_tha     0x1000dad
#define GDK_Sinh_thha    0x1000dae
#define GDK_Sinh_dha     0x1000daf
#define GDK_Sinh_dhha    0x1000db0
#define GDK_Sinh_na      0x1000db1
#define GDK_Sinh_ndha    0x1000db3
#define GDK_Sinh_pa      0x1000db4
#define GDK_Sinh_pha     0x1000db5
#define GDK_Sinh_ba      0x1000db6
#define GDK_Sinh_bha     0x1000db7
#define GDK_Sinh_ma      0x1000db8
#define GDK_Sinh_mba     0x1000db9
#define GDK_Sinh_ya      0x1000dba
#define GDK_Sinh_ra      0x1000dbb
#define GDK_Sinh_la      0x1000dbd
#define GDK_Sinh_va      0x1000dc0
#define GDK_Sinh_sha     0x1000dc1
#define GDK_Sinh_ssha    0x1000dc2
#define GDK_Sinh_sa      0x1000dc3
#define GDK_Sinh_ha      0x1000dc4
#define GDK_Sinh_lla     0x1000dc5
#define GDK_Sinh_fa      0x1000dc6
#define GDK_Sinh_al      0x1000dca
#define GDK_Sinh_aa2     0x1000dcf
#define GDK_Sinh_ae2     0x1000dd0
#define GDK_Sinh_aee2    0x1000dd1

```

```

#define GDK_Sinh_i2      0x1000dd2
#define GDK_Sinh_ii2     0x1000dd3
#define GDK_Sinh_u2      0x1000dd4
#define GDK_Sinh_uu2     0x1000dd6
#define GDK_Sinh_ru2     0x1000dd8
#define GDK_Sinh_e2      0x1000dd9
#define GDK_Sinh_ee2     0x1000dda
#define GDK_Sinh_ai2     0x1000ddb
#define GDK_Sinh_o2      0x1000ddc
#define GDK_Sinh_oo2     0x1000ddd
#define GDK_Sinh_au2     0x1000dde
#define GDK_Sinh_lu2     0x1000ddf
#define GDK_Sinh_ruu2    0x1000df2
#define GDK_Sinh_luu2    0x1000df3
#define GDK_Sinh_kunddaliya 0x1000df4
#define GDK_Georgian_an 0x10010d0
#define GDK_Georgian_ban 0x10010d1
#define GDK_Georgian_gan 0x10010d2
#define GDK_Georgian_don 0x10010d3
#define GDK_Georgian_en 0x10010d4
#define GDK_Georgian_vin 0x10010d5
#define GDK_Georgian_zen 0x10010d6
#define GDK_Georgian_tan 0x10010d7
#define GDK_Georgian_in 0x10010d8
#define GDK_Georgian_kan 0x10010d9
#define GDK_Georgian_las 0x10010da
#define GDK_Georgian_man 0x10010db
#define GDK_Georgian_nar 0x10010dc
#define GDK_Georgian_on 0x10010dd
#define GDK_Georgian_par 0x10010de
#define GDK_Georgian_zhar 0x10010df
#define GDK_Georgian_rae 0x10010e0
#define GDK_Georgian_san 0x10010e1
#define GDK_Georgian_tar 0x10010e2
#define GDK_Georgian_un 0x10010e3
#define GDK_Georgian_phar 0x10010e4
#define GDK_Georgian_khar 0x10010e5
#define GDK_Georgian_ghan 0x10010e6
#define GDK_Georgian_gar 0x10010e7
#define GDK_Georgian_shin 0x10010e8
#define GDK_Georgian_chin 0x10010e9
#define GDK_Georgian_can 0x10010ea
#define GDK_Georgian_jil 0x10010eb
#define GDK_Georgian_cil 0x10010ec
#define GDK_Georgian_char 0x10010ed
#define GDK_Georgian_xan 0x10010ee
#define GDK_Georgian_jhan 0x10010ef
#define GDK_Georgian_hae 0x10010f0
#define GDK_Georgian_he 0x10010f1
#define GDK_Georgian_hie 0x10010f2
#define GDK_Georgian_we 0x10010f3
#define GDK_Georgian_har 0x10010f4
#define GDK_Georgian_hoe 0x10010f5
#define GDK_Georgian_fi 0x10010f6
#define GDK_Babovedot 0x1001e02
#define GDK_babovedot 0x1001e03
#define GDK_Dabovedot 0x1001e0a
#define GDK_dabovedot 0x1001e0b
#define GDK_Fabovedot 0x1001e1e
#define GDK_fabovedot 0x1001e1f
#define GDK_Lbelowdot 0x1001e36
#define GDK_lbelowdot 0x1001e37
#define GDK_Mabovedot 0x1001e40
#define GDK_mabovedot 0x1001e41
#define GDK_Pabovedot 0x1001e56
#define GDK_pabovedot 0x1001e57

```



```

#define GDK_Sabovedot 0x1001e60
#define GDK_sabovedot 0x1001e61
#define GDK_Tabovedot 0x1001e6a
#define GDK_tabovedot 0x1001e6b
#define GDK_Wgrave 0x1001e80
#define GDK_wgrave 0x1001e81
#define GDK_Wacute 0x1001e82
#define GDK_wacute 0x1001e83
#define GDK_Wdiaeresis 0x1001e84
#define GDK_wdiaeresis 0x1001e85
#define GDK_Xabovedot 0x1001e8a
#define GDK_xabovedot 0x1001e8b
#define GDK_Abelowdot 0x1001ea0
#define GDK_abelowdot 0x1001ea1
#define GDK_Ahook 0x1001ea2
#define GDK_ahook 0x1001ea3
#define GDK_Acircumflexacute 0x1001ea4
#define GDK_acircumflexacute 0x1001ea5
#define GDK_Acircumflexgrave 0x1001ea6
#define GDK_acircumflexgrave 0x1001ea7
#define GDK_Acircumflexhook 0x1001ea8
#define GDK_acircumflexhook 0x1001ea9
#define GDK_Acircumflextilde 0x1001eaa
#define GDK_acircumflextilde 0x1001eab
#define GDK_Acircumflexbelowdot 0x1001eac
#define GDK_acircumflexbelowdot 0x1001ead
#define GDK_Abreveacute 0x1001eae
#define GDK_abreveacute 0x1001eaf
#define GDK_Abrevegrave 0x1001eb0
#define GDK_abrevegrave 0x1001eb1
#define GDK_Abrevehook 0x1001eb2
#define GDK_abrevehook 0x1001eb3
#define GDK_Abrevetilde 0x1001eb4
#define GDK_abrevetilde 0x1001eb5
#define GDK_Abrevebelowdot 0x1001eb6
#define GDK_abrevebelowdot 0x1001eb7
#define GDK_Ebelowdot 0x1001eb8
#define GDK_ebelowdot 0x1001eb9
#define GDK_Ehook 0x1001eba
#define GDK_ehook 0x1001ebb
#define GDK_Etilde 0x1001ebc
#define GDK_etilde 0x1001ebd
#define GDK_Ecircumflexacute 0x1001ebe
#define GDK_ecircumflexacute 0x1001ebf
#define GDK_Ecircumflexgrave 0x1001ec0
#define GDK_ecircumflexgrave 0x1001ec1
#define GDK_Ecircumflexhook 0x1001ec2
#define GDK_ecircumflexhook 0x1001ec3
#define GDK_Ecircumflextilde 0x1001ec4
#define GDK_ecircumflextilde 0x1001ec5
#define GDK_Ecircumflexbelowdot 0x1001ec6
#define GDK_ecircumflexbelowdot 0x1001ec7
#define GDK_Ihook 0x1001ec8
#define GDK_ihook 0x1001ec9
#define GDK_Ibelowdot 0x1001eca
#define GDK_ibelowdot 0x1001ecb
#define GDK_Obelowdot 0x1001ecc
#define GDK_obelowdot 0x1001ecd
#define GDK_Ohook 0x1001ece
#define GDK_ohook 0x1001ecf
#define GDK_Ocircumflexacute 0x1001ed0
#define GDK_ocircumflexacute 0x1001ed1
#define GDK_Ocircumflexgrave 0x1001ed2
#define GDK_ocircumflexgrave 0x1001ed3
#define GDK_Ocircumflexhook 0x1001ed4
#define GDK_ocircumflexhook 0x1001ed5

```

```

#define GDK_Ocircumflextilde 0x1001ed6
#define GDK_ocircumflextilde 0x1001ed7
#define GDK_Ocircumflexbelowdot 0x1001ed8
#define GDK_ocircumflexbelowdot 0x1001ed9
#define GDK_Ohornacute 0x1001eda
#define GDK_ohornacute 0x1001edb
#define GDK_Ohorngrave 0x1001edc
#define GDK_ohorngrave 0x1001edd
#define GDK_Ohornhook 0x1001ede
#define GDK_ohornhook 0x1001edf
#define GDK_Ohorntilde 0x1001ee0
#define GDK_ohorntilde 0x1001ee1
#define GDK_Ohornbelowdot 0x1001ee2
#define GDK_ohornbelowdot 0x1001ee3
#define GDK_Ubelowdot 0x1001ee4
#define GDK_ubelowdot 0x1001ee5
#define GDK_Uhook 0x1001ee6
#define GDK_uhook 0x1001ee7
#define GDK_Uhornacute 0x1001ee8
#define GDK_uhornacute 0x1001ee9
#define GDK_Uhorngrave 0x1001eea
#define GDK_uhorngrave 0x1001eeb
#define GDK_Uhornhook 0x1001eec
#define GDK_uhornhook 0x1001eed
#define GDK_Uhorntilde 0x1001eee
#define GDK_uhorntilde 0x1001eef
#define GDK_Uhornbelowdot 0x1001ef0
#define GDK_uhornbelowdot 0x1001ef1
#define GDK_Ygrave 0x1001ef2
#define GDK_ygrave 0x1001ef3
#define GDK_Ybelowdot 0x1001ef4
#define GDK_ybelowdot 0x1001ef5
#define GDK_Yhook 0x1001ef6
#define GDK_yhook 0x1001ef7
#define GDK_Ytilde 0x1001ef8
#define GDK_ytilde 0x1001ef9
#define GDK_zerosuperior 0x1002070
#define GDK_foursuperior 0x1002074
#define GDK_fivesuperior 0x1002075
#define GDK_sixsuperior 0x1002076
#define GDK_sevensuperior 0x1002077
#define GDK_eightsuperior 0x1002078
#define GDK_ninesuperior 0x1002079
#define GDK_zerosubscript 0x1002080
#define GDK_onesubscript 0x1002081
#define GDK_twosubscript 0x1002082
#define GDK_threesubscript 0x1002083
#define GDK_foursubscript 0x1002084
#define GDK_fivesubscript 0x1002085
#define GDK_sixsubscript 0x1002086
#define GDK_sevensubscript 0x1002087
#define GDK_eightsubscript 0x1002088
#define GDK_ninesubscript 0x1002089
#define GDK_partdifferential 0x1002202
#define GDK_emptyset 0x1002205
#define GDK_elementof 0x1002208
#define GDK_notelementof 0x1002209
#define GDK_containsas 0x100220b
#define GDK_squareroot 0x100221a
#define GDK_cuberoot 0x100221b
#define GDK_fourthroot 0x100221c
#define GDK_dintegral 0x100222c
#define GDK_tintegral 0x100222d
#define GDK_because 0x1002235
#define GDK_notapprox 0x1002247
#define GDK_approx 0x1002248

```

```

#define GDK_notidentical      0x1002262
#define GDK_stricteq          0x1002263
#define GDK_braille_blank    0x1002800
#define GDK_braille_dots_1   0x1002801
#define GDK_braille_dots_2   0x1002802
#define GDK_braille_dots_12  0x1002803
#define GDK_braille_dots_3   0x1002804
#define GDK_braille_dots_13  0x1002805
#define GDK_braille_dots_23  0x1002806
#define GDK_braille_dots_123 0x1002807
#define GDK_braille_dots_4   0x1002808
#define GDK_braille_dots_14  0x1002809
#define GDK_braille_dots_24  0x100280a
#define GDK_braille_dots_124 0x100280b
#define GDK_braille_dots_34  0x100280c
#define GDK_braille_dots_134 0x100280d
#define GDK_braille_dots_234 0x100280e
#define GDK_braille_dots_1234 0x100280f
#define GDK_braille_dots_5   0x1002810
#define GDK_braille_dots_15  0x1002811
#define GDK_braille_dots_25  0x1002812
#define GDK_braille_dots_125 0x1002813
#define GDK_braille_dots_35  0x1002814
#define GDK_braille_dots_135 0x1002815
#define GDK_braille_dots_235 0x1002816
#define GDK_braille_dots_1235 0x1002817
#define GDK_braille_dots_45  0x1002818
#define GDK_braille_dots_145 0x1002819
#define GDK_braille_dots_245 0x100281a
#define GDK_braille_dots_1245 0x100281b
#define GDK_braille_dots_345 0x100281c
#define GDK_braille_dots_1345 0x100281d
#define GDK_braille_dots_2345 0x100281e
#define GDK_braille_dots_12345 0x100281f
#define GDK_braille_dots_6   0x1002820
#define GDK_braille_dots_16  0x1002821
#define GDK_braille_dots_26  0x1002822
#define GDK_braille_dots_126 0x1002823
#define GDK_braille_dots_36  0x1002824
#define GDK_braille_dots_136 0x1002825
#define GDK_braille_dots_236 0x1002826
#define GDK_braille_dots_1236 0x1002827
#define GDK_braille_dots_46  0x1002828
#define GDK_braille_dots_146 0x1002829
#define GDK_braille_dots_246 0x100282a
#define GDK_braille_dots_1246 0x100282b
#define GDK_braille_dots_346 0x100282c
#define GDK_braille_dots_1346 0x100282d
#define GDK_braille_dots_2346 0x100282e
#define GDK_braille_dots_12346 0x100282f
#define GDK_braille_dots_56  0x1002830
#define GDK_braille_dots_156 0x1002831
#define GDK_braille_dots_256 0x1002832
#define GDK_braille_dots_1256 0x1002833
#define GDK_braille_dots_356 0x1002834
#define GDK_braille_dots_1356 0x1002835
#define GDK_braille_dots_2356 0x1002836
#define GDK_braille_dots_12356 0x1002837
#define GDK_braille_dots_456 0x1002838
#define GDK_braille_dots_1456 0x1002839
#define GDK_braille_dots_2456 0x100283a
#define GDK_braille_dots_12456 0x100283b
#define GDK_braille_dots_3456 0x100283c
#define GDK_braille_dots_13456 0x100283d
#define GDK_braille_dots_23456 0x100283e
#define GDK_braille_dots_123456 0x100283f

```

```

#define GDK_braille_dots_7      0x1002840
#define GDK_braille_dots_17     0x1002841
#define GDK_braille_dots_27     0x1002842
#define GDK_braille_dots_127    0x1002843
#define GDK_braille_dots_37     0x1002844
#define GDK_braille_dots_137    0x1002845
#define GDK_braille_dots_237    0x1002846
#define GDK_braille_dots_1237   0x1002847
#define GDK_braille_dots_47     0x1002848
#define GDK_braille_dots_147    0x1002849
#define GDK_braille_dots_247    0x100284a
#define GDK_braille_dots_1247   0x100284b
#define GDK_braille_dots_347    0x100284c
#define GDK_braille_dots_1347   0x100284d
#define GDK_braille_dots_2347   0x100284e
#define GDK_braille_dots_12347  0x100284f
#define GDK_braille_dots_57     0x1002850
#define GDK_braille_dots_157    0x1002851
#define GDK_braille_dots_257    0x1002852
#define GDK_braille_dots_1257   0x1002853
#define GDK_braille_dots_357    0x1002854
#define GDK_braille_dots_1357   0x1002855
#define GDK_braille_dots_2357   0x1002856
#define GDK_braille_dots_12357  0x1002857
#define GDK_braille_dots_457    0x1002858
#define GDK_braille_dots_1457   0x1002859
#define GDK_braille_dots_2457   0x100285a
#define GDK_braille_dots_12457  0x100285b
#define GDK_braille_dots_3457   0x100285c
#define GDK_braille_dots_13457  0x100285d
#define GDK_braille_dots_23457  0x100285e
#define GDK_braille_dots_123457 0x100285f
#define GDK_braille_dots_67     0x1002860
#define GDK_braille_dots_167    0x1002861
#define GDK_braille_dots_267    0x1002862
#define GDK_braille_dots_1267   0x1002863
#define GDK_braille_dots_367    0x1002864
#define GDK_braille_dots_1367   0x1002865
#define GDK_braille_dots_2367   0x1002866
#define GDK_braille_dots_12367  0x1002867
#define GDK_braille_dots_467    0x1002868
#define GDK_braille_dots_1467   0x1002869
#define GDK_braille_dots_2467   0x100286a
#define GDK_braille_dots_12467  0x100286b
#define GDK_braille_dots_3467   0x100286c
#define GDK_braille_dots_13467  0x100286d
#define GDK_braille_dots_23467  0x100286e
#define GDK_braille_dots_123467 0x100286f
#define GDK_braille_dots_567    0x1002870
#define GDK_braille_dots_1567   0x1002871
#define GDK_braille_dots_2567   0x1002872
#define GDK_braille_dots_12567  0x1002873
#define GDK_braille_dots_3567   0x1002874
#define GDK_braille_dots_13567  0x1002875
#define GDK_braille_dots_23567  0x1002876
#define GDK_braille_dots_123567 0x1002877
#define GDK_braille_dots_4567   0x1002878
#define GDK_braille_dots_14567  0x1002879
#define GDK_braille_dots_24567  0x100287a
#define GDK_braille_dots_124567 0x100287b
#define GDK_braille_dots_34567  0x100287c
#define GDK_braille_dots_134567 0x100287d
#define GDK_braille_dots_234567 0x100287e
#define GDK_braille_dots_1234567 0x100287f
#define GDK_braille_dots_8      0x1002880
#define GDK_braille_dots_18     0x1002881

```

```

#define GDK_braille_dots_28      0x1002882
#define GDK_braille_dots_128     0x1002883
#define GDK_braille_dots_38      0x1002884
#define GDK_braille_dots_138     0x1002885
#define GDK_braille_dots_238     0x1002886
#define GDK_braille_dots_1238    0x1002887
#define GDK_braille_dots_48      0x1002888
#define GDK_braille_dots_148     0x1002889
#define GDK_braille_dots_248     0x100288a
#define GDK_braille_dots_1248    0x100288b
#define GDK_braille_dots_348     0x100288c
#define GDK_braille_dots_1348    0x100288d
#define GDK_braille_dots_2348    0x100288e
#define GDK_braille_dots_12348   0x100288f
#define GDK_braille_dots_58      0x1002890
#define GDK_braille_dots_158     0x1002891
#define GDK_braille_dots_258     0x1002892
#define GDK_braille_dots_1258    0x1002893
#define GDK_braille_dots_358     0x1002894
#define GDK_braille_dots_1358    0x1002895
#define GDK_braille_dots_2358    0x1002896
#define GDK_braille_dots_12358   0x1002897
#define GDK_braille_dots_458     0x1002898
#define GDK_braille_dots_1458    0x1002899
#define GDK_braille_dots_2458    0x100289a
#define GDK_braille_dots_12458   0x100289b
#define GDK_braille_dots_3458    0x100289c
#define GDK_braille_dots_13458   0x100289d
#define GDK_braille_dots_23458   0x100289e
#define GDK_braille_dots_123458  0x100289f
#define GDK_braille_dots_68      0x10028a0
#define GDK_braille_dots_168     0x10028a1
#define GDK_braille_dots_268     0x10028a2
#define GDK_braille_dots_1268    0x10028a3
#define GDK_braille_dots_368     0x10028a4
#define GDK_braille_dots_1368    0x10028a5
#define GDK_braille_dots_2368    0x10028a6
#define GDK_braille_dots_12368   0x10028a7
#define GDK_braille_dots_468     0x10028a8
#define GDK_braille_dots_1468    0x10028a9
#define GDK_braille_dots_2468    0x10028aa
#define GDK_braille_dots_12468   0x10028ab
#define GDK_braille_dots_3468    0x10028ac
#define GDK_braille_dots_13468   0x10028ad
#define GDK_braille_dots_23468   0x10028ae
#define GDK_braille_dots_123468  0x10028af
#define GDK_braille_dots_568     0x10028b0
#define GDK_braille_dots_1568    0x10028b1
#define GDK_braille_dots_2568    0x10028b2
#define GDK_braille_dots_12568   0x10028b3
#define GDK_braille_dots_3568    0x10028b4
#define GDK_braille_dots_13568   0x10028b5
#define GDK_braille_dots_23568   0x10028b6
#define GDK_braille_dots_123568  0x10028b7
#define GDK_braille_dots_4568    0x10028b8
#define GDK_braille_dots_14568   0x10028b9
#define GDK_braille_dots_24568   0x10028ba
#define GDK_braille_dots_124568  0x10028bb
#define GDK_braille_dots_34568   0x10028bc
#define GDK_braille_dots_134568  0x10028bd
#define GDK_braille_dots_234568  0x10028be
#define GDK_braille_dots_1234568 0x10028bf
#define GDK_braille_dots_78      0x10028c0
#define GDK_braille_dots_178     0x10028c1
#define GDK_braille_dots_278     0x10028c2
#define GDK_braille_dots_1278    0x10028c3

```

```

#define GDK_braille_dots_378 0x10028c4
#define GDK_braille_dots_1378 0x10028c5
#define GDK_braille_dots_2378 0x10028c6
#define GDK_braille_dots_12378 0x10028c7
#define GDK_braille_dots_478 0x10028c8
#define GDK_braille_dots_1478 0x10028c9
#define GDK_braille_dots_2478 0x10028ca
#define GDK_braille_dots_12478 0x10028cb
#define GDK_braille_dots_3478 0x10028cc
#define GDK_braille_dots_13478 0x10028cd
#define GDK_braille_dots_23478 0x10028ce
#define GDK_braille_dots_123478 0x10028cf
#define GDK_braille_dots_578 0x10028d0
#define GDK_braille_dots_1578 0x10028d1
#define GDK_braille_dots_2578 0x10028d2
#define GDK_braille_dots_12578 0x10028d3
#define GDK_braille_dots_3578 0x10028d4
#define GDK_braille_dots_13578 0x10028d5
#define GDK_braille_dots_23578 0x10028d6
#define GDK_braille_dots_123578 0x10028d7
#define GDK_braille_dots_4578 0x10028d8
#define GDK_braille_dots_14578 0x10028d9
#define GDK_braille_dots_24578 0x10028da
#define GDK_braille_dots_124578 0x10028db
#define GDK_braille_dots_34578 0x10028dc
#define GDK_braille_dots_134578 0x10028dd
#define GDK_braille_dots_234578 0x10028de
#define GDK_braille_dots_1234578 0x10028df
#define GDK_braille_dots_678 0x10028e0
#define GDK_braille_dots_1678 0x10028e1
#define GDK_braille_dots_2678 0x10028e2
#define GDK_braille_dots_12678 0x10028e3
#define GDK_braille_dots_3678 0x10028e4
#define GDK_braille_dots_13678 0x10028e5
#define GDK_braille_dots_23678 0x10028e6
#define GDK_braille_dots_123678 0x10028e7
#define GDK_braille_dots_4678 0x10028e8
#define GDK_braille_dots_14678 0x10028e9
#define GDK_braille_dots_24678 0x10028ea
#define GDK_braille_dots_124678 0x10028eb
#define GDK_braille_dots_34678 0x10028ec
#define GDK_braille_dots_134678 0x10028ed
#define GDK_braille_dots_234678 0x10028ee
#define GDK_braille_dots_1234678 0x10028ef
#define GDK_braille_dots_5678 0x10028f0
#define GDK_braille_dots_15678 0x10028f1
#define GDK_braille_dots_25678 0x10028f2
#define GDK_braille_dots_125678 0x10028f3
#define GDK_braille_dots_35678 0x10028f4
#define GDK_braille_dots_135678 0x10028f5
#define GDK_braille_dots_235678 0x10028f6
#define GDK_braille_dots_1235678 0x10028f7
#define GDK_braille_dots_45678 0x10028f8
#define GDK_braille_dots_145678 0x10028f9
#define GDK_braille_dots_245678 0x10028fa
#define GDK_braille_dots_1245678 0x10028fb
#define GDK_braille_dots_345678 0x10028fc
#define GDK_braille_dots_1345678 0x10028fd
#define GDK_braille_dots_2345678 0x10028fe
#define GDK_braille_dots_12345678 0x10028ff
#define GDK_Switch_VT_1 0x1008fe01
#define GDK_Switch_VT_2 0x1008fe02
#define GDK_Switch_VT_3 0x1008fe03
#define GDK_Switch_VT_4 0x1008fe04
#define GDK_Switch_VT_5 0x1008fe05
#define GDK_Switch_VT_6 0x1008fe06

```



```

#define GDK_Switch_VT_7 0x1008fe07
#define GDK_Switch_VT_8 0x1008fe08
#define GDK_Switch_VT_9 0x1008fe09
#define GDK_Switch_VT_10 0x1008fe0a
#define GDK_Switch_VT_11 0x1008fe0b
#define GDK_Switch_VT_12 0x1008fe0c
#define GDK_Ungrab 0x1008fe20
#define GDK_ClearGrab 0x1008fe21
#define GDK_Next_VMode 0x1008fe22
#define GDK_Prev_VMode 0x1008fe23
#define GDK_LogWindowTree 0x1008fe24
#define GDK_LogGrabInfo 0x1008fe25
#define GDK_ModeLock 0x1008ff01
#define GDK_MonBrightnessUp 0x1008ff02
#define GDK_MonBrightnessDown 0x1008ff03
#define GDK_KbdLightOnOff 0x1008ff04
#define GDK_KbdBrightnessUp 0x1008ff05
#define GDK_KbdBrightnessDown 0x1008ff06
#define GDK_Standby 0x1008ff10
#define GDK_AudioLowerVolume 0x1008ff11
#define GDK_AudioMute 0x1008ff12
#define GDK_AudioRaiseVolume 0x1008ff13
#define GDK_AudioPlay 0x1008ff14
#define GDK_AudioStop 0x1008ff15
#define GDK_AudioPrev 0x1008ff16
#define GDK_AudioNext 0x1008ff17
#define GDK_HomePage 0x1008ff18
#define GDK_Mail 0x1008ff19
#define GDK_Start 0x1008ff1a
#define GDK_Search 0x1008ff1b
#define GDK_AudioRecord 0x1008ff1c
#define GDK_Calculator 0x1008ff1d
#define GDK_Memo 0x1008ff1e
#define GDK_ToDoList 0x1008ff1f
#define GDK_Calendar 0x1008ff20
#define GDK_PowerDown 0x1008ff21
#define GDK_ContrastAdjust 0x1008ff22
#define GDK_RockerUp 0x1008ff23
#define GDK_RockerDown 0x1008ff24
#define GDK_RockerEnter 0x1008ff25
#define GDK_Back 0x1008ff26
#define GDK_Forward 0x1008ff27
#define GDK_Stop 0x1008ff28
#define GDK_Refresh 0x1008ff29
#define GDK_PowerOff 0x1008ff2a
#define GDK_WakeUp 0x1008ff2b
#define GDK_Eject 0x1008ff2c
#define GDK_ScreenSaver 0x1008ff2d
#define GDK_WWW 0x1008ff2e
#define GDK_Sleep 0x1008ff2f
#define GDK_Favorites 0x1008ff30
#define GDK_AudioPause 0x1008ff31
#define GDK_AudioMedia 0x1008ff32
#define GDK_MyComputer 0x1008ff33
#define GDK_VendorHome 0x1008ff34
#define GDK_LightBulb 0x1008ff35
#define GDK_Shop 0x1008ff36
#define GDK_History 0x1008ff37
#define GDK_OpenURL 0x1008ff38
#define GDK_AddFavorite 0x1008ff39
#define GDK_HotLinks 0x1008ff3a
#define GDK_BrightnessAdjust 0x1008ff3b
#define GDK_Finance 0x1008ff3c
#define GDK_Community 0x1008ff3d
#define GDK_AudioRewind 0x1008ff3e
#define GDK_BackForward 0x1008ff3f

```

```

#define GDK_Launch0      0x1008ff40
#define GDK_Launch1      0x1008ff41
#define GDK_Launch2      0x1008ff42
#define GDK_Launch3      0x1008ff43
#define GDK_Launch4      0x1008ff44
#define GDK_Launch5      0x1008ff45
#define GDK_Launch6      0x1008ff46
#define GDK_Launch7      0x1008ff47
#define GDK_Launch8      0x1008ff48
#define GDK_Launch9      0x1008ff49
#define GDK_LaunchA      0x1008ff4a
#define GDK_LaunchB      0x1008ff4b
#define GDK_LaunchC      0x1008ff4c
#define GDK_LaunchD      0x1008ff4d
#define GDK_LaunchE      0x1008ff4e
#define GDK_LaunchF      0x1008ff4f
#define GDK_ApplicationLeft 0x1008ff50
#define GDK_ApplicationRight 0x1008ff51
#define GDK_Book          0x1008ff52
#define GDK_CD            0x1008ff53
#define GDK_WindowClear 0x1008ff55
#define GDK_Close         0x1008ff56
#define GDK_Copy          0x1008ff57
#define GDK_Cut           0x1008ff58
#define GDK_Display       0x1008ff59
#define GDK_DOS           0x1008ff5a
#define GDK_Documents     0x1008ff5b
#define GDK_Excel         0x1008ff5c
#define GDK_Explorer      0x1008ff5d
#define GDK_Game          0x1008ff5e
#define GDK_Go            0x1008ff5f
#define GDK_iTouch       0x1008ff60
#define GDK_LogOff        0x1008ff61
#define GDK_Market        0x1008ff62
#define GDK_Meeting       0x1008ff63
#define GDK_MenuKB        0x1008ff65
#define GDK_MenuPB        0x1008ff66
#define GDK_MySites       0x1008ff67
#define GDK_New           0x1008ff68
#define GDK_News          0x1008ff69
#define GDK_OfficeHome    0x1008ff6a
#define GDK_Open          0x1008ff6b
#define GDK_Option        0x1008ff6c
#define GDK_Paste         0x1008ff6d
#define GDK_Phone         0x1008ff6e
#define GDK_Reply         0x1008ff72
#define GDK_Reload        0x1008ff73
#define GDK_RotateWindows 0x1008ff74
#define GDK_RotationPB    0x1008ff75
#define GDK_RotationKB    0x1008ff76
#define GDK_Save          0x1008ff77
#define GDK_ScrollUp      0x1008ff78
#define GDK_ScrollDown    0x1008ff79
#define GDK_ScrollClick   0x1008ff7a
#define GDK_Send          0x1008ff7b
#define GDK_Spell         0x1008ff7c
#define GDK_SplitScreen   0x1008ff7d
#define GDK_Support       0x1008ff7e
#define GDK_TaskPane      0x1008ff7f
#define GDK_Terminal      0x1008ff80
#define GDK_Tools         0x1008ff81
#define GDK_Travel        0x1008ff82
#define GDK_UserPB        0x1008ff84
#define GDK_User1KB       0x1008ff85
#define GDK_User2KB       0x1008ff86
#define GDK_Video         0x1008ff87

```



```

#define GDK_WheelButton 0x1008ff88
#define GDK_Word 0x1008ff89
#define GDK_Xfer 0x1008ff8a
#define GDK_ZoomIn 0x1008ff8b
#define GDK_ZoomOut 0x1008ff8c
#define GDK_Away 0x1008ff8d
#define GDK_Messenger 0x1008ff8e
#define GDK_WebCam 0x1008ff8f
#define GDK_MailForward 0x1008ff90
#define GDK_Pictures 0x1008ff91
#define GDK_Music 0x1008ff92
#define GDK_Battery 0x1008ff93
#define GDK_Bluetooth 0x1008ff94
#define GDK_WLAN 0x1008ff95
#define GDK_UWB 0x1008ff96
#define GDK_AudioForward 0x1008ff97
#define GDK_AudioRepeat 0x1008ff98
#define GDK_AudioRandomPlay 0x1008ff99
#define GDK_Subtitle 0x1008ff9a
#define GDK_AudioCycleTrack 0x1008ff9b
#define GDK_CycleAngle 0x1008ff9c
#define GDK_FrameBack 0x1008ff9d
#define GDK_FrameForward 0x1008ff9e
#define GDK_Time 0x1008ff9f
#define GDK_SelectButton 0x1008ffa0
#define GDK_View 0x1008ffa1
#define GDK_TopMenu 0x1008ffa2
#define GDK_Red 0x1008ffa3
#define GDK_Green 0x1008ffa4
#define GDK_Yellow 0x1008ffa5
#define GDK_Blue 0x1008ffa6
#define GDK_Suspend 0x1008ffa7
#define GDK_Hibernate 0x1008ffa8
#define GDK_TouchpadToggle 0x1008ffa9
#define GDK_TouchpadOn 0x1008ffb0
#define GDK_TouchpadOff 0x1008ffb1
#define GDK_permille 0xad5
#define GDK_ISO_Level5_Shift 0xfe11
#define GDK_ISO_Level5_Latch 0xfe12
#define GDK_ISO_Level5_Lock 0xfe13
#define GDK_dead_perispomeni 0xfe53
#define GDK_dead_stroke 0xfe63
#define GDK_dead_abovecomma 0xfe64
#define GDK_dead_psili 0xfe64
#define GDK_dead_aboverversedcomma 0xfe65
#define GDK_dead_dasia 0xfe65
#define GDK_dead_doublegrave 0xfe66
#define GDK_dead_belowring 0xfe67
#define GDK_dead_belowmacron 0xfe68
#define GDK_dead_belowcircumflex 0xfe69
#define GDK_dead_belowtilde 0xfe6a
#define GDK_dead_belowbreve 0xfe6b
#define GDK_dead_belowdiaeresis 0xfe6c
#define GDK_dead_invertedbreve 0xfe6d
#define GDK_dead_belowcomma 0xfe6e
#define GDK_dead_currency 0xfe6f
#define GDK_dead_a 0xfe80
#define GDK_dead_A 0xfe81
#define GDK_dead_e 0xfe82
#define GDK_dead_E 0xfe83
#define GDK_dead_i 0xfe84
#define GDK_dead_I 0xfe85
#define GDK_dead_o 0xfe86
#define GDK_dead_O 0xfe87
#define GDK_dead_u 0xfe88
#define GDK_dead_U 0xfe89

```

```

#define GDK_dead_small_schwa 0xfe8a
#define GDK_dead_capital_schwa 0xfe8b
#define GDK_dead_greek 0xfe8c
#define GDK_ch 0xfea0
#define GDK_Ch 0xfea1
#define GDK_CH 0xfea2
#define GDK_c_h 0xfea3
#define GDK_C_h 0xfea4
#define GDK_C_H 0xfea5
#define GDK_braille_dot_1 0xffff1
#define GDK_braille_dot_2 0xffff2
#define GDK_braille_dot_3 0xffff3
#define GDK_braille_dot_4 0xffff4
#define GDK_braille_dot_5 0xffff5
#define GDK_braille_dot_6 0xffff6
#define GDK_braille_dot_7 0xffff7
#define GDK_braille_dot_8 0xffff8
#define GDK_braille_dot_9 0xffff9
#define GDK_braille_dot_10 0xffffa

```

7.3.4 gtk-3.0/gdk/gdkkeysyms.h

```

#define _GDK_KEYSyms_H_
#define GDK_KEY_space 0x020
#define GDK_KEY_exclam 0x021
#define GDK_KEY_quotedbl 0x022
#define GDK_KEY_numbersign 0x023
#define GDK_KEY_dollar 0x024
#define GDK_KEY_percent 0x025
#define GDK_KEY_ampersand 0x026
#define GDK_KEY_apostrophe 0x027
#define GDK_KEY_quoteright 0x027
#define GDK_KEY_parenleft 0x028
#define GDK_KEY_parenright 0x029
#define GDK_KEY_asterisk 0x02a
#define GDK_KEY_plus 0x02b
#define GDK_KEY_comma 0x02c
#define GDK_KEY_minus 0x02d
#define GDK_KEY_period 0x02e
#define GDK_KEY_slash 0x02f
#define GDK_KEY_0 0x030
#define GDK_KEY_1 0x031
#define GDK_KEY_2 0x032
#define GDK_KEY_3 0x033
#define GDK_KEY_4 0x034
#define GDK_KEY_5 0x035
#define GDK_KEY_6 0x036
#define GDK_KEY_7 0x037
#define GDK_KEY_8 0x038
#define GDK_KEY_9 0x039
#define GDK_KEY_colon 0x03a
#define GDK_KEY_semicolon 0x03b
#define GDK_KEY_less 0x03c
#define GDK_KEY_equal 0x03d
#define GDK_KEY_greater 0x03e
#define GDK_KEY_question 0x03f
#define GDK_KEY_at 0x040
#define GDK_KEY_A 0x041
#define GDK_KEY_B 0x042
#define GDK_KEY_C 0x043
#define GDK_KEY_D 0x044
#define GDK_KEY_E 0x045
#define GDK_KEY_F 0x046
#define GDK_KEY_G 0x047
#define GDK_KEY_H 0x048

```

```

#define GDK_KEY_I      0x049
#define GDK_KEY_J      0x04a
#define GDK_KEY_K      0x04b
#define GDK_KEY_L      0x04c
#define GDK_KEY_M      0x04d
#define GDK_KEY_N      0x04e
#define GDK_KEY_O      0x04f
#define GDK_KEY_P      0x050
#define GDK_KEY_Q      0x051
#define GDK_KEY_R      0x052
#define GDK_KEY_S      0x053
#define GDK_KEY_T      0x054
#define GDK_KEY_U      0x055
#define GDK_KEY_V      0x056
#define GDK_KEY_W      0x057
#define GDK_KEY_X      0x058
#define GDK_KEY_Y      0x059
#define GDK_KEY_Z      0x05a
#define GDK_KEY_bracketleft 0x05b
#define GDK_KEY_backslash 0x05c
#define GDK_KEY_bracketright 0x05d
#define GDK_KEY_asciicircum 0x05e
#define GDK_KEY_underscore 0x05f
#define GDK_KEY_grave 0x060
#define GDK_KEY_quoteleft 0x060
#define GDK_KEY_a      0x061
#define GDK_KEY_b      0x062
#define GDK_KEY_c      0x063
#define GDK_KEY_d      0x064
#define GDK_KEY_e      0x065
#define GDK_KEY_f      0x066
#define GDK_KEY_g      0x067
#define GDK_KEY_h      0x068
#define GDK_KEY_i      0x069
#define GDK_KEY_j      0x06a
#define GDK_KEY_k      0x06b
#define GDK_KEY_l      0x06c
#define GDK_KEY_m      0x06d
#define GDK_KEY_n      0x06e
#define GDK_KEY_o      0x06f
#define GDK_KEY_p      0x070
#define GDK_KEY_q      0x071
#define GDK_KEY_r      0x072
#define GDK_KEY_s      0x073
#define GDK_KEY_t      0x074
#define GDK_KEY_u      0x075
#define GDK_KEY_v      0x076
#define GDK_KEY_w      0x077
#define GDK_KEY_x      0x078
#define GDK_KEY_y      0x079
#define GDK_KEY_z      0x07a
#define GDK_KEY_braceleft 0x07b
#define GDK_KEY_bar 0x07c
#define GDK_KEY_braceright 0x07d
#define GDK_KEY_asciitilde 0x07e
#define GDK_KEY_nobreakspace 0x0a0
#define GDK_KEY_exclamdown 0x0a1
#define GDK_KEY_cent 0x0a2
#define GDK_KEY_sterling 0x0a3
#define GDK_KEY_currency 0x0a4
#define GDK_KEY_yen 0x0a5
#define GDK_KEY_brokenbar 0x0a6
#define GDK_KEY_section 0x0a7
#define GDK_KEY_diaeresis 0x0a8
#define GDK_KEY_copyright 0x0a9
#define GDK_KEY_ordfeminine 0x0aa

```

```

#define GDK_KEY_guillemotleft 0x0ab
#define GDK_KEY_notsign 0x0ac
#define GDK_KEY_hyphen 0x0ad
#define GDK_KEY_registered 0x0ae
#define GDK_KEY_macron 0x0af
#define GDK_KEY_degree 0x0b0
#define GDK_KEY_plusminus 0x0b1
#define GDK_KEY_twosuperior 0x0b2
#define GDK_KEY_threesuperior 0x0b3
#define GDK_KEY_acute 0x0b4
#define GDK_KEY_mu 0x0b5
#define GDK_KEY_paragraph 0x0b6
#define GDK_KEY_periodcentered 0x0b7
#define GDK_KEY_cedilla 0x0b8
#define GDK_KEY_onesuperior 0x0b9
#define GDK_KEY_masculine 0x0ba
#define GDK_KEY_guillemotright 0x0bb
#define GDK_KEY_onequarter 0x0bc
#define GDK_KEY_onehalf 0x0bd
#define GDK_KEY_threequarters 0x0be
#define GDK_KEY_questiondown 0x0bf
#define GDK_KEY_Agrave 0x0c0
#define GDK_KEY_Aacute 0x0c1
#define GDK_KEY_Acircumflex 0x0c2
#define GDK_KEY_Atilde 0x0c3
#define GDK_KEY_Adiaeresis 0x0c4
#define GDK_KEY_Aring 0x0c5
#define GDK_KEY_AE 0x0c6
#define GDK_KEY_Ccedilla 0x0c7
#define GDK_KEY_Egrave 0x0c8
#define GDK_KEY_Eacute 0x0c9
#define GDK_KEY_Ecircumflex 0x0ca
#define GDK_KEY_Ediaeresis 0x0cb
#define GDK_KEY_Igrave 0x0cc
#define GDK_KEY_Iacute 0x0cd
#define GDK_KEY_Icircumflex 0x0ce
#define GDK_KEY_Idiaeresis 0x0cf
#define GDK_KEY_ETH 0x0d0
#define GDK_KEY_Eth 0x0d0
#define GDK_KEY_Ntilde 0x0d1
#define GDK_KEY_Ograve 0x0d2
#define GDK_KEY_Oacute 0x0d3
#define GDK_KEY_Ocircumflex 0x0d4
#define GDK_KEY_Otilde 0x0d5
#define GDK_KEY_Odiaeresis 0x0d6
#define GDK_KEY_multiply 0x0d7
#define GDK_KEY_Ooblique 0x0d8
#define GDK_KEY_Oslash 0x0d8
#define GDK_KEY_Ugrave 0x0d9
#define GDK_KEY_Uacute 0x0da
#define GDK_KEY_Ucircumflex 0x0db
#define GDK_KEY_Udiaeresis 0x0dc
#define GDK_KEY_Yacute 0x0dd
#define GDK_KEY_THORN 0x0de
#define GDK_KEY_Thorn 0x0de
#define GDK_KEY_ssharp 0x0df
#define GDK_KEY_agrave 0x0e0
#define GDK_KEY_aacute 0x0e1
#define GDK_KEY_acircumflex 0x0e2
#define GDK_KEY_atilde 0x0e3
#define GDK_KEY_adiaeresis 0x0e4
#define GDK_KEY_aring 0x0e5
#define GDK_KEY_ae 0x0e6
#define GDK_KEY_ccedilla 0x0e7
#define GDK_KEY_egrave 0x0e8
#define GDK_KEY_eacute 0x0e9

```

```

#define GDK_KEY_ecircumflex 0x0ea
#define GDK_KEY_ediaeresis 0x0eb
#define GDK_KEY_igrave 0x0ec
#define GDK_KEY_iacute 0x0ed
#define GDK_KEY_icircumflex 0x0ee
#define GDK_KEY_idiaeresis 0x0ef
#define GDK_KEY_eth 0x0f0
#define GDK_KEY_ntilde 0x0f1
#define GDK_KEY_ograve 0x0f2
#define GDK_KEY_oacute 0x0f3
#define GDK_KEY_ocircumflex 0x0f4
#define GDK_KEY_otilde 0x0f5
#define GDK_KEY_odiaeresis 0x0f6
#define GDK_KEY_division 0x0f7
#define GDK_KEY_oblique 0x0f8
#define GDK_KEY_oshlash 0x0f8
#define GDK_KEY_ugrave 0x0f9
#define GDK_KEY_uacute 0x0fa
#define GDK_KEY_ucircumflex 0x0fb
#define GDK_KEY_udiaeresis 0x0fc
#define GDK_KEY_yacute 0x0fd
#define GDK_KEY_thorn 0x0fe
#define GDK_KEY_ydiaeresis 0x0ff
#define GDK_KEY_Ibreve 0x100012c
#define GDK_KEY_ibreve 0x100012d
#define GDK_KEY_Wcircumflex 0x1000174
#define GDK_KEY_wcircumflex 0x1000175
#define GDK_KEY_Ycircumflex 0x1000176
#define GDK_KEY_ycircumflex 0x1000177
#define GDK_KEY_SCHWA 0x100018f
#define GDK_KEY_Obarred 0x100019f
#define GDK_KEY_Ohorn 0x10001a0
#define GDK_KEY_ohorn 0x10001a1
#define GDK_KEY_Uhorn 0x10001af
#define GDK_KEY_uhorn 0x10001b0
#define GDK_KEY_Zstroke 0x10001b5
#define GDK_KEY_zstroke 0x10001b6
#define GDK_KEY_EZH 0x10001b7
#define GDK_KEY_Ocaron 0x10001d1
#define GDK_KEY_ocaron 0x10001d2
#define GDK_KEY_Gcaron 0x10001e6
#define GDK_KEY_gcaron 0x10001e7
#define GDK_KEY_schwa 0x1000259
#define GDK_KEY_obarred 0x1000275
#define GDK_KEY_ezh 0x1000292
#define GDK_KEY_Cyrillic_GHE_bar 0x1000492
#define GDK_KEY_Cyrillic_ghe_bar 0x1000493
#define GDK_KEY_Cyrillic_ZHE_descender 0x1000496
#define GDK_KEY_Cyrillic_zhe_descender 0x1000497
#define GDK_KEY_Cyrillic_KA_descender 0x100049a
#define GDK_KEY_Cyrillic_ka_descender 0x100049b
#define GDK_KEY_Cyrillic_KA_vertstroke 0x100049c
#define GDK_KEY_Cyrillic_ka_vertstroke 0x100049d
#define GDK_KEY_Cyrillic_EN_descender 0x10004a2
#define GDK_KEY_Cyrillic_en_descender 0x10004a3
#define GDK_KEY_Cyrillic_U_straight 0x10004ae
#define GDK_KEY_Cyrillic_u_straight 0x10004af
#define GDK_KEY_Cyrillic_U_straight_bar 0x10004b0
#define GDK_KEY_Cyrillic_u_straight_bar 0x10004b1
#define GDK_KEY_Cyrillic_HA_descender 0x10004b2
#define GDK_KEY_Cyrillic_ha_descender 0x10004b3
#define GDK_KEY_Cyrillic_CHE_descender 0x10004b6
#define GDK_KEY_Cyrillic_che_descender 0x10004b7
#define GDK_KEY_Cyrillic_CHE_vertstroke 0x10004b8
#define GDK_KEY_Cyrillic_che_vertstroke 0x10004b9
#define GDK_KEY_Cyrillic_SHHA 0x10004ba

```

```

#define GDK_KEY_Cyrillic_shha 0x10004bb
#define GDK_KEY_Cyrillic_SCHWA 0x10004d8
#define GDK_KEY_Cyrillic_schwa 0x10004d9
#define GDK_KEY_Cyrillic_I_macron 0x10004e2
#define GDK_KEY_Cyrillic_i_macron 0x10004e3
#define GDK_KEY_Cyrillic_O_bar 0x10004e8
#define GDK_KEY_Cyrillic_o_bar 0x10004e9
#define GDK_KEY_Cyrillic_U_macron 0x10004ee
#define GDK_KEY_Cyrillic_u_macron 0x10004ef
#define GDK_KEY_Armenian_AYB 0x1000531
#define GDK_KEY_Armenian_BEN 0x1000532
#define GDK_KEY_Armenian_GIM 0x1000533
#define GDK_KEY_Armenian_DA 0x1000534
#define GDK_KEY_Armenian_YECH 0x1000535
#define GDK_KEY_Armenian_ZA 0x1000536
#define GDK_KEY_Armenian_E 0x1000537
#define GDK_KEY_Armenian_AT 0x1000538
#define GDK_KEY_Armenian_TO 0x1000539
#define GDK_KEY_Armenian_ZHE 0x100053a
#define GDK_KEY_Armenian_INI 0x100053b
#define GDK_KEY_Armenian_LYUN 0x100053c
#define GDK_KEY_Armenian_KHE 0x100053d
#define GDK_KEY_Armenian_TSA 0x100053e
#define GDK_KEY_Armenian_KEN 0x100053f
#define GDK_KEY_Armenian_HO 0x1000540
#define GDK_KEY_Armenian_DZA 0x1000541
#define GDK_KEY_Armenian_GHAT 0x1000542
#define GDK_KEY_Armenian_TCHE 0x1000543
#define GDK_KEY_Armenian_MEN 0x1000544
#define GDK_KEY_Armenian_HI 0x1000545
#define GDK_KEY_Armenian_NU 0x1000546
#define GDK_KEY_Armenian_SHA 0x1000547
#define GDK_KEY_Armenian_VO 0x1000548
#define GDK_KEY_Armenian_CHA 0x1000549
#define GDK_KEY_Armenian_PE 0x100054a
#define GDK_KEY_Armenian_JE 0x100054b
#define GDK_KEY_Armenian_RA 0x100054c
#define GDK_KEY_Armenian_SE 0x100054d
#define GDK_KEY_Armenian_VEV 0x100054e
#define GDK_KEY_Armenian_TYUN 0x100054f
#define GDK_KEY_Armenian_RE 0x1000550
#define GDK_KEY_Armenian_TSO 0x1000551
#define GDK_KEY_Armenian_VYUN 0x1000552
#define GDK_KEY_Armenian_PYUR 0x1000553
#define GDK_KEY_Armenian_KE 0x1000554
#define GDK_KEY_Armenian_O 0x1000555
#define GDK_KEY_Armenian_FE 0x1000556
#define GDK_KEY_Armenian_apostrophe 0x100055a
#define GDK_KEY_Armenian_accent 0x100055b
#define GDK_KEY_Armenian_shesht 0x100055b
#define GDK_KEY_Armenian_amanak 0x100055c
#define GDK_KEY_Armenian_exclam 0x100055c
#define GDK_KEY_Armenian_but 0x100055d
#define GDK_KEY_Armenian_separation_mark 0x100055d
#define GDK_KEY_Armenian_paruyk 0x100055e
#define GDK_KEY_Armenian_question 0x100055e
#define GDK_KEY_Armenian_ayb 0x1000561
#define GDK_KEY_Armenian_ben 0x1000562
#define GDK_KEY_Armenian_gim 0x1000563
#define GDK_KEY_Armenian_da 0x1000564
#define GDK_KEY_Armenian_yech 0x1000565
#define GDK_KEY_Armenian_za 0x1000566
#define GDK_KEY_Armenian_e 0x1000567
#define GDK_KEY_Armenian_at 0x1000568
#define GDK_KEY_Armenian_to 0x1000569
#define GDK_KEY_Armenian_zhe 0x100056a

```

```

#define GDK_KEY_Armenian_ini 0x100056b
#define GDK_KEY_Armenian_lyun 0x100056c
#define GDK_KEY_Armenian_khe 0x100056d
#define GDK_KEY_Armenian_tsa 0x100056e
#define GDK_KEY_Armenian_ken 0x100056f
#define GDK_KEY_Armenian_ho 0x1000570
#define GDK_KEY_Armenian_dza 0x1000571
#define GDK_KEY_Armenian_ghat 0x1000572
#define GDK_KEY_Armenian_tche 0x1000573
#define GDK_KEY_Armenian_men 0x1000574
#define GDK_KEY_Armenian_hi 0x1000575
#define GDK_KEY_Armenian_nu 0x1000576
#define GDK_KEY_Armenian_sha 0x1000577
#define GDK_KEY_Armenian_vo 0x1000578
#define GDK_KEY_Armenian_cha 0x1000579
#define GDK_KEY_Armenian_pe 0x100057a
#define GDK_KEY_Armenian_je 0x100057b
#define GDK_KEY_Armenian_ra 0x100057c
#define GDK_KEY_Armenian_se 0x100057d
#define GDK_KEY_Armenian_vev 0x100057e
#define GDK_KEY_Armenian_tyun 0x100057f
#define GDK_KEY_Armenian_re 0x1000580
#define GDK_KEY_Armenian_tso 0x1000581
#define GDK_KEY_Armenian_vyun 0x1000582
#define GDK_KEY_Armenian_pyur 0x1000583
#define GDK_KEY_Armenian_ke 0x1000584
#define GDK_KEY_Armenian_o 0x1000585
#define GDK_KEY_Armenian_fe 0x1000586
#define GDK_KEY_Armenian_ligature_ew 0x1000587
#define GDK_KEY_Armenian_full_stop 0x1000589
#define GDK_KEY_Armenian_verjaket 0x1000589
#define GDK_KEY_Armenian_hyphen 0x100058a
#define GDK_KEY_Armenian_yentamna 0x100058a
#define GDK_KEY_Arabic_madda_above 0x1000653
#define GDK_KEY_Arabic_hamza_above 0x1000654
#define GDK_KEY_Arabic_hamza_below 0x1000655
#define GDK_KEY_Arabic_0 0x1000660
#define GDK_KEY_Arabic_1 0x1000661
#define GDK_KEY_Arabic_2 0x1000662
#define GDK_KEY_Arabic_3 0x1000663
#define GDK_KEY_Arabic_4 0x1000664
#define GDK_KEY_Arabic_5 0x1000665
#define GDK_KEY_Arabic_6 0x1000666
#define GDK_KEY_Arabic_7 0x1000667
#define GDK_KEY_Arabic_8 0x1000668
#define GDK_KEY_Arabic_9 0x1000669
#define GDK_KEY_Arabic_percent 0x100066a
#define GDK_KEY_Arabic_superscript_alef 0x1000670
#define GDK_KEY_Arabic_tteh 0x1000679
#define GDK_KEY_Arabic_peh 0x100067e
#define GDK_KEY_Arabic_tcheh 0x1000686
#define GDK_KEY_Arabic_ddal 0x1000688
#define GDK_KEY_Arabic_rreh 0x1000691
#define GDK_KEY_Arabic_jeh 0x1000698
#define GDK_KEY_Arabic_veh 0x10006a4
#define GDK_KEY_Arabic_keheh 0x10006a9
#define GDK_KEY_Arabic_gaf 0x10006af
#define GDK_KEY_Arabic_noon_ghunna 0x10006ba
#define GDK_KEY_Arabic_heh_doachashmee 0x10006be
#define GDK_KEY_Arabic_heh_goal 0x10006c1
#define GDK_KEY_Arabic_farsi_yeh 0x10006cc
#define GDK_KEY_Farsi_yeh 0x10006cc
#define GDK_KEY_Arabic_yeh_baree 0x10006d2
#define GDK_KEY_Arabic_fullstop 0x10006d4
#define GDK_KEY_Farsi_0 0x10006f0
#define GDK_KEY_Farsi_1 0x10006f1

```



```

#define GDK_KEY_Farsi_2 0x10006f2
#define GDK_KEY_Farsi_3 0x10006f3
#define GDK_KEY_Farsi_4 0x10006f4
#define GDK_KEY_Farsi_5 0x10006f5
#define GDK_KEY_Farsi_6 0x10006f6
#define GDK_KEY_Farsi_7 0x10006f7
#define GDK_KEY_Farsi_8 0x10006f8
#define GDK_KEY_Farsi_9 0x10006f9
#define GDK_KEY_Sinh_ng 0x1000d82
#define GDK_KEY_Sinh_h2 0x1000d83
#define GDK_KEY_Sinh_a 0x1000d85
#define GDK_KEY_Sinh_aa 0x1000d86
#define GDK_KEY_Sinh_ae 0x1000d87
#define GDK_KEY_Sinh_aee 0x1000d88
#define GDK_KEY_Sinh_i 0x1000d89
#define GDK_KEY_Sinh_ii 0x1000d8a
#define GDK_KEY_Sinh_u 0x1000d8b
#define GDK_KEY_Sinh_uu 0x1000d8c
#define GDK_KEY_Sinh_ri 0x1000d8d
#define GDK_KEY_Sinh_rii 0x1000d8e
#define GDK_KEY_Sinh_lu 0x1000d8f
#define GDK_KEY_Sinh_luu 0x1000d90
#define GDK_KEY_Sinh_e 0x1000d91
#define GDK_KEY_Sinh_ee 0x1000d92
#define GDK_KEY_Sinh_ai 0x1000d93
#define GDK_KEY_Sinh_o 0x1000d94
#define GDK_KEY_Sinh_oo 0x1000d95
#define GDK_KEY_Sinh_au 0x1000d96
#define GDK_KEY_Sinh_ka 0x1000d9a
#define GDK_KEY_Sinh_kha 0x1000d9b
#define GDK_KEY_Sinh_ga 0x1000d9c
#define GDK_KEY_Sinh_gha 0x1000d9d
#define GDK_KEY_Sinh_ng2 0x1000d9e
#define GDK_KEY_Sinh_nga 0x1000d9f
#define GDK_KEY_Sinh_ca 0x1000da0
#define GDK_KEY_Sinh_cha 0x1000da1
#define GDK_KEY_Sinh_ja 0x1000da2
#define GDK_KEY_Sinh_jha 0x1000da3
#define GDK_KEY_Sinh_nya 0x1000da4
#define GDK_KEY_Sinh_jnya 0x1000da5
#define GDK_KEY_Sinh_nja 0x1000da6
#define GDK_KEY_Sinh_tta 0x1000da7
#define GDK_KEY_Sinh_ttha 0x1000da8
#define GDK_KEY_Sinh_dda 0x1000da9
#define GDK_KEY_Sinh_ddha 0x1000daa
#define GDK_KEY_Sinh_nna 0x1000dab
#define GDK_KEY_Sinh_ndda 0x1000dac
#define GDK_KEY_Sinh_tha 0x1000dad
#define GDK_KEY_Sinh_thha 0x1000dae
#define GDK_KEY_Sinh_dha 0x1000daf
#define GDK_KEY_Sinh_dhha 0x1000db0
#define GDK_KEY_Sinh_na 0x1000db1
#define GDK_KEY_Sinh_ndha 0x1000db3
#define GDK_KEY_Sinh_pa 0x1000db4
#define GDK_KEY_Sinh pha 0x1000db5
#define GDK_KEY_Sinh_ba 0x1000db6
#define GDK_KEY_Sinh_bha 0x1000db7
#define GDK_KEY_Sinh_ma 0x1000db8
#define GDK_KEY_Sinh_mba 0x1000db9
#define GDK_KEY_Sinh_ya 0x1000dba
#define GDK_KEY_Sinh_ra 0x1000dbb
#define GDK_KEY_Sinh_la 0x1000dbd
#define GDK_KEY_Sinh_va 0x1000dc0
#define GDK_KEY_Sinh_sha 0x1000dc1
#define GDK_KEY_Sinh_ssha 0x1000dc2
#define GDK_KEY_Sinh_sa 0x1000dc3

```



```

#define GDK_KEY_Sinh_ha 0x1000dc4
#define GDK_KEY_Sinh_lla 0x1000dc5
#define GDK_KEY_Sinh_fa 0x1000dc6
#define GDK_KEY_Sinh_al 0x1000dca
#define GDK_KEY_Sinh_aa2 0x1000dcf
#define GDK_KEY_Sinh_ae2 0x1000dd0
#define GDK_KEY_Sinh_aee2 0x1000dd1
#define GDK_KEY_Sinh_i2 0x1000dd2
#define GDK_KEY_Sinh_ii2 0x1000dd3
#define GDK_KEY_Sinh_u2 0x1000dd4
#define GDK_KEY_Sinh_uu2 0x1000dd6
#define GDK_KEY_Sinh_ru2 0x1000dd8
#define GDK_KEY_Sinh_e2 0x1000dd9
#define GDK_KEY_Sinh_ee2 0x1000dda
#define GDK_KEY_Sinh_ai2 0x1000ddb
#define GDK_KEY_Sinh_o2 0x1000ddc
#define GDK_KEY_Sinh_oo2 0x1000ddd
#define GDK_KEY_Sinh_au2 0x1000dde
#define GDK_KEY_Sinh_lu2 0x1000ddf
#define GDK_KEY_Sinh_ruu2 0x1000df2
#define GDK_KEY_Sinh_luu2 0x1000df3
#define GDK_KEY_Sinh_kunddaliya 0x1000df4
#define GDK_KEY_Georgian_an 0x10010d0
#define GDK_KEY_Georgian_ban 0x10010d1
#define GDK_KEY_Georgian_gan 0x10010d2
#define GDK_KEY_Georgian_don 0x10010d3
#define GDK_KEY_Georgian_en 0x10010d4
#define GDK_KEY_Georgian_vin 0x10010d5
#define GDK_KEY_Georgian_zen 0x10010d6
#define GDK_KEY_Georgian_tan 0x10010d7
#define GDK_KEY_Georgian_in 0x10010d8
#define GDK_KEY_Georgian_kan 0x10010d9
#define GDK_KEY_Georgian_las 0x10010da
#define GDK_KEY_Georgian_man 0x10010db
#define GDK_KEY_Georgian_nar 0x10010dc
#define GDK_KEY_Georgian_on 0x10010dd
#define GDK_KEY_Georgian_par 0x10010de
#define GDK_KEY_Georgian_zhar 0x10010df
#define GDK_KEY_Georgian_rae 0x10010e0
#define GDK_KEY_Georgian_san 0x10010e1
#define GDK_KEY_Georgian_tar 0x10010e2
#define GDK_KEY_Georgian_un 0x10010e3
#define GDK_KEY_Georgian_phar 0x10010e4
#define GDK_KEY_Georgian_khar 0x10010e5
#define GDK_KEY_Georgian_ghan 0x10010e6
#define GDK_KEY_Georgian_qar 0x10010e7
#define GDK_KEY_Georgian_shin 0x10010e8
#define GDK_KEY_Georgian_chin 0x10010e9
#define GDK_KEY_Georgian_can 0x10010ea
#define GDK_KEY_Georgian_jil 0x10010eb
#define GDK_KEY_Georgian_cil 0x10010ec
#define GDK_KEY_Georgian_char 0x10010ed
#define GDK_KEY_Georgian_xan 0x10010ee
#define GDK_KEY_Georgian_jhan 0x10010ef
#define GDK_KEY_Georgian_hae 0x10010f0
#define GDK_KEY_Georgian_he 0x10010f1
#define GDK_KEY_Georgian_hie 0x10010f2
#define GDK_KEY_Georgian_we 0x10010f3
#define GDK_KEY_Georgian_har 0x10010f4
#define GDK_KEY_Georgian_hoe 0x10010f5
#define GDK_KEY_Georgian_fi 0x10010f6
#define GDK_KEY_Babovedot 0x1001e02
#define GDK_KEY_babovedot 0x1001e03
#define GDK_KEY_Dabovedot 0x1001e0a
#define GDK_KEY_dabovedot 0x1001e0b
#define GDK_KEY_Fabovedot 0x1001e1e

```

```

#define GDK_KEY_fabovedot 0x1001e1f
#define GDK_KEY_lbelowdot 0x1001e36
#define GDK_KEY_lbelowdot 0x1001e37
#define GDK_KEY_Mabovedot 0x1001e40
#define GDK_KEY_mabovedot 0x1001e41
#define GDK_KEY_Pabovedot 0x1001e56
#define GDK_KEY_pabovedot 0x1001e57
#define GDK_KEY_Sabovedot 0x1001e60
#define GDK_KEY_sabovedot 0x1001e61
#define GDK_KEY_Tabovedot 0x1001e6a
#define GDK_KEY_tabovedot 0x1001e6b
#define GDK_KEY_Wgrave 0x1001e80
#define GDK_KEY_wgrave 0x1001e81
#define GDK_KEY_Wacute 0x1001e82
#define GDK_KEY_wacute 0x1001e83
#define GDK_KEY_Wdiaeresis 0x1001e84
#define GDK_KEY_wdiaeresis 0x1001e85
#define GDK_KEY_Xabovedot 0x1001e8a
#define GDK_KEY_xabovedot 0x1001e8b
#define GDK_KEY_Abelowdot 0x1001ea0
#define GDK_KEY_abelowdot 0x1001ea1
#define GDK_KEY_Ahook 0x1001ea2
#define GDK_KEY_ahook 0x1001ea3
#define GDK_KEY_Acircumflexacute 0x1001ea4
#define GDK_KEY_acircumflexacute 0x1001ea5
#define GDK_KEY_Acircumflexgrave 0x1001ea6
#define GDK_KEY_acircumflexgrave 0x1001ea7
#define GDK_KEY_Acircumflexhook 0x1001ea8
#define GDK_KEY_acircumflexhook 0x1001ea9
#define GDK_KEY_Acircumflextilde 0x1001eaa
#define GDK_KEY_acircumflextilde 0x1001eab
#define GDK_KEY_Acircumflexbelowdot 0x1001eac
#define GDK_KEY_acircumflexbelowdot 0x1001ead
#define GDK_KEY_Abreveacute 0x1001eae
#define GDK_KEY_abreveacute 0x1001eaf
#define GDK_KEY_Abrevegrave 0x1001eb0
#define GDK_KEY_abrevegrave 0x1001eb1
#define GDK_KEY_Abrevehook 0x1001eb2
#define GDK_KEY_abrevehook 0x1001eb3
#define GDK_KEY_Abrevetilde 0x1001eb4
#define GDK_KEY_abrevetilde 0x1001eb5
#define GDK_KEY_Abrevebelowdot 0x1001eb6
#define GDK_KEY_abrevebelowdot 0x1001eb7
#define GDK_KEY_Ebelowdot 0x1001eb8
#define GDK_KEY_ebelowdot 0x1001eb9
#define GDK_KEY_Ehook 0x1001eba
#define GDK_KEY_ehook 0x1001ebb
#define GDK_KEY_Etilde 0x1001ebc
#define GDK_KEY_etilde 0x1001ebd
#define GDK_KEY_Ecircumflexacute 0x1001ebe
#define GDK_KEY_ecircumflexacute 0x1001ebf
#define GDK_KEY_Ecircumflexgrave 0x1001ec0
#define GDK_KEY_ecircumflexgrave 0x1001ec1
#define GDK_KEY_Ecircumflexhook 0x1001ec2
#define GDK_KEY_ecircumflexhook 0x1001ec3
#define GDK_KEY_Ecircumflextilde 0x1001ec4
#define GDK_KEY_ecircumflextilde 0x1001ec5
#define GDK_KEY_Ecircumflexbelowdot 0x1001ec6
#define GDK_KEY_ecircumflexbelowdot 0x1001ec7
#define GDK_KEY_Ihook 0x1001ec8
#define GDK_KEY_ihook 0x1001ec9
#define GDK_KEY_Ibelowdot 0x1001eca
#define GDK_KEY_ibelowdot 0x1001ecb
#define GDK_KEY_Obelowdot 0x1001ecc
#define GDK_KEY_obelowdot 0x1001ecd
#define GDK_KEY_Ohook 0x1001ece

```

```

#define GDK_KEY_ohook 0x1001ecf
#define GDK_KEY_Ocircumflexacute 0x1001ed0
#define GDK_KEY_ocircumflexacute 0x1001ed1
#define GDK_KEY_Ocircumflexgrave 0x1001ed2
#define GDK_KEY_ocircumflexgrave 0x1001ed3
#define GDK_KEY_Ocircumflexhook 0x1001ed4
#define GDK_KEY_ocircumflexhook 0x1001ed5
#define GDK_KEY_Ocircumflextilde 0x1001ed6
#define GDK_KEY_ocircumflextilde 0x1001ed7
#define GDK_KEY_Ocircumflexbelowdot 0x1001ed8
#define GDK_KEY_ocircumflexbelowdot 0x1001ed9
#define GDK_KEY_Ohornacute 0x1001eda
#define GDK_KEY_ohornacute 0x1001edb
#define GDK_KEY_Ohorngrave 0x1001edc
#define GDK_KEY_ohorngrave 0x1001edd
#define GDK_KEY_Ohornhook 0x1001ede
#define GDK_KEY_ohornhook 0x1001edf
#define GDK_KEY_Ohorntilde 0x1001ee0
#define GDK_KEY_ohorntilde 0x1001ee1
#define GDK_KEY_Ohornbelowdot 0x1001ee2
#define GDK_KEY_ohornbelowdot 0x1001ee3
#define GDK_KEY_Ubelowdot 0x1001ee4
#define GDK_KEY_ubelowdot 0x1001ee5
#define GDK_KEY_Uhook 0x1001ee6
#define GDK_KEY_uhook 0x1001ee7
#define GDK_KEY_Uhornacute 0x1001ee8
#define GDK_KEY_uhornacute 0x1001ee9
#define GDK_KEY_Uhorngrave 0x1001eea
#define GDK_KEY_uhorngrave 0x1001eeb
#define GDK_KEY_Uhornhook 0x1001eec
#define GDK_KEY_uhornhook 0x1001eed
#define GDK_KEY_Uhorntilde 0x1001eee
#define GDK_KEY_uhorntilde 0x1001eef
#define GDK_KEY_Uhornbelowdot 0x1001ef0
#define GDK_KEY_uhornbelowdot 0x1001ef1
#define GDK_KEY_Ygrave 0x1001ef2
#define GDK_KEY_ygrave 0x1001ef3
#define GDK_KEY_Ybelowdot 0x1001ef4
#define GDK_KEY_ybelowdot 0x1001ef5
#define GDK_KEY_Yhook 0x1001ef6
#define GDK_KEY_yhook 0x1001ef7
#define GDK_KEY_Ytilde 0x1001ef8
#define GDK_KEY_ytilde 0x1001ef9
#define GDK_KEY_zerosuperior 0x1002070
#define GDK_KEY_foursuperior 0x1002074
#define GDK_KEY_fivesuperior 0x1002075
#define GDK_KEY_sixsuperior 0x1002076
#define GDK_KEY_sevensuperior 0x1002077
#define GDK_KEY_eightsuperior 0x1002078
#define GDK_KEY_ninesuperior 0x1002079
#define GDK_KEY_zerosubscript 0x1002080
#define GDK_KEY_onesubscript 0x1002081
#define GDK_KEY_twosubscript 0x1002082
#define GDK_KEY_threesubscript 0x1002083
#define GDK_KEY_foursubscript 0x1002084
#define GDK_KEY_fivesubscript 0x1002085
#define GDK_KEY_sixsubscript 0x1002086
#define GDK_KEY_sevensubscript 0x1002087
#define GDK_KEY_eightsubscript 0x1002088
#define GDK_KEY_ninesubscript 0x1002089
#define GDK_KEY_EcuSign 0x10020a0
#define GDK_KEY_ColonSign 0x10020a1
#define GDK_KEY_CruzeiroSign 0x10020a2
#define GDK_KEY_FFrancSign 0x10020a3
#define GDK_KEY_LiraSign 0x10020a4
#define GDK_KEY_MillSign 0x10020a5

```

```

#define GDK_KEY_NairaSign 0x10020a6
#define GDK_KEY_PesetaSign 0x10020a7
#define GDK_KEY_RupeeSign 0x10020a8
#define GDK_KEY_WonSign 0x10020a9
#define GDK_KEY_NewSheqelSign 0x10020aa
#define GDK_KEY_DongSign 0x10020ab
#define GDK_KEY_partdifferential 0x1002202
#define GDK_KEY_emptyset 0x1002205
#define GDK_KEY_elementof 0x1002208
#define GDK_KEY_notelementof 0x1002209
#define GDK_KEY_containsas 0x100220b
#define GDK_KEY_squareroot 0x100221a
#define GDK_KEY_cuberoot 0x100221b
#define GDK_KEY_fourthroot 0x100221c
#define GDK_KEY_dintegral 0x100222c
#define GDK_KEY_tintegral 0x100222d
#define GDK_KEY_because 0x1002235
#define GDK_KEY_notapproxq 0x1002247
#define GDK_KEY_approxq 0x1002248
#define GDK_KEY_notidentical 0x1002262
#define GDK_KEY_stricteq 0x1002263
#define GDK_KEY_braille_blank 0x1002800
#define GDK_KEY_braille_dots_1 0x1002801
#define GDK_KEY_braille_dots_2 0x1002802
#define GDK_KEY_braille_dots_12 0x1002803
#define GDK_KEY_braille_dots_3 0x1002804
#define GDK_KEY_braille_dots_13 0x1002805
#define GDK_KEY_braille_dots_23 0x1002806
#define GDK_KEY_braille_dots_123 0x1002807
#define GDK_KEY_braille_dots_4 0x1002808
#define GDK_KEY_braille_dots_14 0x1002809
#define GDK_KEY_braille_dots_24 0x100280a
#define GDK_KEY_braille_dots_124 0x100280b
#define GDK_KEY_braille_dots_34 0x100280c
#define GDK_KEY_braille_dots_134 0x100280d
#define GDK_KEY_braille_dots_234 0x100280e
#define GDK_KEY_braille_dots_1234 0x100280f
#define GDK_KEY_braille_dots_5 0x1002810
#define GDK_KEY_braille_dots_15 0x1002811
#define GDK_KEY_braille_dots_25 0x1002812
#define GDK_KEY_braille_dots_125 0x1002813
#define GDK_KEY_braille_dots_35 0x1002814
#define GDK_KEY_braille_dots_135 0x1002815
#define GDK_KEY_braille_dots_235 0x1002816
#define GDK_KEY_braille_dots_1235 0x1002817
#define GDK_KEY_braille_dots_45 0x1002818
#define GDK_KEY_braille_dots_145 0x1002819
#define GDK_KEY_braille_dots_245 0x100281a
#define GDK_KEY_braille_dots_1245 0x100281b
#define GDK_KEY_braille_dots_345 0x100281c
#define GDK_KEY_braille_dots_1345 0x100281d
#define GDK_KEY_braille_dots_2345 0x100281e
#define GDK_KEY_braille_dots_12345 0x100281f
#define GDK_KEY_braille_dots_6 0x1002820
#define GDK_KEY_braille_dots_16 0x1002821
#define GDK_KEY_braille_dots_26 0x1002822
#define GDK_KEY_braille_dots_126 0x1002823
#define GDK_KEY_braille_dots_36 0x1002824
#define GDK_KEY_braille_dots_136 0x1002825
#define GDK_KEY_braille_dots_236 0x1002826
#define GDK_KEY_braille_dots_1236 0x1002827
#define GDK_KEY_braille_dots_46 0x1002828
#define GDK_KEY_braille_dots_146 0x1002829
#define GDK_KEY_braille_dots_246 0x100282a
#define GDK_KEY_braille_dots_1246 0x100282b
#define GDK_KEY_braille_dots_346 0x100282c

```

```

#define GDK_KEY_braille_dots_1346      0x100282d
#define GDK_KEY_braille_dots_2346      0x100282e
#define GDK_KEY_braille_dots_12346     0x100282f
#define GDK_KEY_braille_dots_56 0x1002830
#define GDK_KEY_braille_dots_156       0x1002831
#define GDK_KEY_braille_dots_256       0x1002832
#define GDK_KEY_braille_dots_1256      0x1002833
#define GDK_KEY_braille_dots_356       0x1002834
#define GDK_KEY_braille_dots_1356      0x1002835
#define GDK_KEY_braille_dots_2356      0x1002836
#define GDK_KEY_braille_dots_12356     0x1002837
#define GDK_KEY_braille_dots_456       0x1002838
#define GDK_KEY_braille_dots_1456      0x1002839
#define GDK_KEY_braille_dots_2456      0x100283a
#define GDK_KEY_braille_dots_12456     0x100283b
#define GDK_KEY_braille_dots_3456      0x100283c
#define GDK_KEY_braille_dots_13456     0x100283d
#define GDK_KEY_braille_dots_23456     0x100283e
#define GDK_KEY_braille_dots_123456    0x100283f
#define GDK_KEY_braille_dots_7 0x1002840
#define GDK_KEY_braille_dots_17 0x1002841
#define GDK_KEY_braille_dots_27 0x1002842
#define GDK_KEY_braille_dots_127       0x1002843
#define GDK_KEY_braille_dots_37 0x1002844
#define GDK_KEY_braille_dots_137       0x1002845
#define GDK_KEY_braille_dots_237       0x1002846
#define GDK_KEY_braille_dots_1237      0x1002847
#define GDK_KEY_braille_dots_47 0x1002848
#define GDK_KEY_braille_dots_147       0x1002849
#define GDK_KEY_braille_dots_247       0x100284a
#define GDK_KEY_braille_dots_1247      0x100284b
#define GDK_KEY_braille_dots_347       0x100284c
#define GDK_KEY_braille_dots_1347      0x100284d
#define GDK_KEY_braille_dots_2347      0x100284e
#define GDK_KEY_braille_dots_12347     0x100284f
#define GDK_KEY_braille_dots_57 0x1002850
#define GDK_KEY_braille_dots_157       0x1002851
#define GDK_KEY_braille_dots_257       0x1002852
#define GDK_KEY_braille_dots_1257      0x1002853
#define GDK_KEY_braille_dots_357       0x1002854
#define GDK_KEY_braille_dots_1357      0x1002855
#define GDK_KEY_braille_dots_2357      0x1002856
#define GDK_KEY_braille_dots_12357     0x1002857
#define GDK_KEY_braille_dots_457       0x1002858
#define GDK_KEY_braille_dots_1457      0x1002859
#define GDK_KEY_braille_dots_2457      0x100285a
#define GDK_KEY_braille_dots_12457     0x100285b
#define GDK_KEY_braille_dots_3457      0x100285c
#define GDK_KEY_braille_dots_13457     0x100285d
#define GDK_KEY_braille_dots_23457     0x100285e
#define GDK_KEY_braille_dots_123457    0x100285f
#define GDK_KEY_braille_dots_67 0x1002860
#define GDK_KEY_braille_dots_167       0x1002861
#define GDK_KEY_braille_dots_267       0x1002862
#define GDK_KEY_braille_dots_1267      0x1002863
#define GDK_KEY_braille_dots_367       0x1002864
#define GDK_KEY_braille_dots_1367      0x1002865
#define GDK_KEY_braille_dots_2367      0x1002866
#define GDK_KEY_braille_dots_12367     0x1002867
#define GDK_KEY_braille_dots_467       0x1002868
#define GDK_KEY_braille_dots_1467      0x1002869
#define GDK_KEY_braille_dots_2467      0x100286a
#define GDK_KEY_braille_dots_12467     0x100286b
#define GDK_KEY_braille_dots_3467      0x100286c
#define GDK_KEY_braille_dots_13467     0x100286d
#define GDK_KEY_braille_dots_23467     0x100286e

```

```

#define GDK_KEY_braille_dots_123467 0x100286f
#define GDK_KEY_braille_dots_567 0x1002870
#define GDK_KEY_braille_dots_1567 0x1002871
#define GDK_KEY_braille_dots_2567 0x1002872
#define GDK_KEY_braille_dots_12567 0x1002873
#define GDK_KEY_braille_dots_3567 0x1002874
#define GDK_KEY_braille_dots_13567 0x1002875
#define GDK_KEY_braille_dots_23567 0x1002876
#define GDK_KEY_braille_dots_123567 0x1002877
#define GDK_KEY_braille_dots_4567 0x1002878
#define GDK_KEY_braille_dots_14567 0x1002879
#define GDK_KEY_braille_dots_24567 0x100287a
#define GDK_KEY_braille_dots_124567 0x100287b
#define GDK_KEY_braille_dots_34567 0x100287c
#define GDK_KEY_braille_dots_134567 0x100287d
#define GDK_KEY_braille_dots_234567 0x100287e
#define GDK_KEY_braille_dots_1234567 0x100287f
#define GDK_KEY_braille_dots_8 0x1002880
#define GDK_KEY_braille_dots_18 0x1002881
#define GDK_KEY_braille_dots_28 0x1002882
#define GDK_KEY_braille_dots_128 0x1002883
#define GDK_KEY_braille_dots_38 0x1002884
#define GDK_KEY_braille_dots_138 0x1002885
#define GDK_KEY_braille_dots_238 0x1002886
#define GDK_KEY_braille_dots_1238 0x1002887
#define GDK_KEY_braille_dots_48 0x1002888
#define GDK_KEY_braille_dots_148 0x1002889
#define GDK_KEY_braille_dots_248 0x100288a
#define GDK_KEY_braille_dots_1248 0x100288b
#define GDK_KEY_braille_dots_348 0x100288c
#define GDK_KEY_braille_dots_1348 0x100288d
#define GDK_KEY_braille_dots_2348 0x100288e
#define GDK_KEY_braille_dots_12348 0x100288f
#define GDK_KEY_braille_dots_58 0x1002890
#define GDK_KEY_braille_dots_158 0x1002891
#define GDK_KEY_braille_dots_258 0x1002892
#define GDK_KEY_braille_dots_1258 0x1002893
#define GDK_KEY_braille_dots_358 0x1002894
#define GDK_KEY_braille_dots_1358 0x1002895
#define GDK_KEY_braille_dots_2358 0x1002896
#define GDK_KEY_braille_dots_12358 0x1002897
#define GDK_KEY_braille_dots_458 0x1002898
#define GDK_KEY_braille_dots_1458 0x1002899
#define GDK_KEY_braille_dots_2458 0x100289a
#define GDK_KEY_braille_dots_12458 0x100289b
#define GDK_KEY_braille_dots_3458 0x100289c
#define GDK_KEY_braille_dots_13458 0x100289d
#define GDK_KEY_braille_dots_23458 0x100289e
#define GDK_KEY_braille_dots_123458 0x100289f
#define GDK_KEY_braille_dots_68 0x10028a0
#define GDK_KEY_braille_dots_168 0x10028a1
#define GDK_KEY_braille_dots_268 0x10028a2
#define GDK_KEY_braille_dots_1268 0x10028a3
#define GDK_KEY_braille_dots_368 0x10028a4
#define GDK_KEY_braille_dots_1368 0x10028a5
#define GDK_KEY_braille_dots_2368 0x10028a6
#define GDK_KEY_braille_dots_12368 0x10028a7
#define GDK_KEY_braille_dots_468 0x10028a8
#define GDK_KEY_braille_dots_1468 0x10028a9
#define GDK_KEY_braille_dots_2468 0x10028aa
#define GDK_KEY_braille_dots_12468 0x10028ab
#define GDK_KEY_braille_dots_3468 0x10028ac
#define GDK_KEY_braille_dots_13468 0x10028ad
#define GDK_KEY_braille_dots_23468 0x10028ae
#define GDK_KEY_braille_dots_123468 0x10028af
#define GDK_KEY_braille_dots_568 0x10028b0

```



```

#define GDK_KEY_braille_dots_1568 0x10028b1
#define GDK_KEY_braille_dots_2568 0x10028b2
#define GDK_KEY_braille_dots_12568 0x10028b3
#define GDK_KEY_braille_dots_3568 0x10028b4
#define GDK_KEY_braille_dots_13568 0x10028b5
#define GDK_KEY_braille_dots_23568 0x10028b6
#define GDK_KEY_braille_dots_123568 0x10028b7
#define GDK_KEY_braille_dots_4568 0x10028b8
#define GDK_KEY_braille_dots_14568 0x10028b9
#define GDK_KEY_braille_dots_24568 0x10028ba
#define GDK_KEY_braille_dots_124568 0x10028bb
#define GDK_KEY_braille_dots_34568 0x10028bc
#define GDK_KEY_braille_dots_134568 0x10028bd
#define GDK_KEY_braille_dots_234568 0x10028be
#define GDK_KEY_braille_dots_1234568 0x10028bf
#define GDK_KEY_braille_dots_78 0x10028c0
#define GDK_KEY_braille_dots_178 0x10028c1
#define GDK_KEY_braille_dots_278 0x10028c2
#define GDK_KEY_braille_dots_1278 0x10028c3
#define GDK_KEY_braille_dots_378 0x10028c4
#define GDK_KEY_braille_dots_1378 0x10028c5
#define GDK_KEY_braille_dots_2378 0x10028c6
#define GDK_KEY_braille_dots_12378 0x10028c7
#define GDK_KEY_braille_dots_478 0x10028c8
#define GDK_KEY_braille_dots_1478 0x10028c9
#define GDK_KEY_braille_dots_2478 0x10028ca
#define GDK_KEY_braille_dots_12478 0x10028cb
#define GDK_KEY_braille_dots_3478 0x10028cc
#define GDK_KEY_braille_dots_13478 0x10028cd
#define GDK_KEY_braille_dots_23478 0x10028ce
#define GDK_KEY_braille_dots_123478 0x10028cf
#define GDK_KEY_braille_dots_578 0x10028d0
#define GDK_KEY_braille_dots_1578 0x10028d1
#define GDK_KEY_braille_dots_2578 0x10028d2
#define GDK_KEY_braille_dots_12578 0x10028d3
#define GDK_KEY_braille_dots_3578 0x10028d4
#define GDK_KEY_braille_dots_13578 0x10028d5
#define GDK_KEY_braille_dots_23578 0x10028d6
#define GDK_KEY_braille_dots_123578 0x10028d7
#define GDK_KEY_braille_dots_4578 0x10028d8
#define GDK_KEY_braille_dots_14578 0x10028d9
#define GDK_KEY_braille_dots_24578 0x10028da
#define GDK_KEY_braille_dots_124578 0x10028db
#define GDK_KEY_braille_dots_34578 0x10028dc
#define GDK_KEY_braille_dots_134578 0x10028dd
#define GDK_KEY_braille_dots_234578 0x10028de
#define GDK_KEY_braille_dots_1234578 0x10028df
#define GDK_KEY_braille_dots_678 0x10028e0
#define GDK_KEY_braille_dots_1678 0x10028e1
#define GDK_KEY_braille_dots_2678 0x10028e2
#define GDK_KEY_braille_dots_12678 0x10028e3
#define GDK_KEY_braille_dots_3678 0x10028e4
#define GDK_KEY_braille_dots_13678 0x10028e5
#define GDK_KEY_braille_dots_23678 0x10028e6
#define GDK_KEY_braille_dots_123678 0x10028e7
#define GDK_KEY_braille_dots_4678 0x10028e8
#define GDK_KEY_braille_dots_14678 0x10028e9
#define GDK_KEY_braille_dots_24678 0x10028ea
#define GDK_KEY_braille_dots_124678 0x10028eb
#define GDK_KEY_braille_dots_34678 0x10028ec
#define GDK_KEY_braille_dots_134678 0x10028ed
#define GDK_KEY_braille_dots_234678 0x10028ee
#define GDK_KEY_braille_dots_1234678 0x10028ef
#define GDK_KEY_braille_dots_5678 0x10028f0
#define GDK_KEY_braille_dots_15678 0x10028f1
#define GDK_KEY_braille_dots_25678 0x10028f2

```

```

#define GDK_KEY_braille_dots_125678 0x10028f3
#define GDK_KEY_braille_dots_35678 0x10028f4
#define GDK_KEY_braille_dots_135678 0x10028f5
#define GDK_KEY_braille_dots_235678 0x10028f6
#define GDK_KEY_braille_dots_1235678 0x10028f7
#define GDK_KEY_braille_dots_45678 0x10028f8
#define GDK_KEY_braille_dots_145678 0x10028f9
#define GDK_KEY_braille_dots_245678 0x10028fa
#define GDK_KEY_braille_dots_1245678 0x10028fb
#define GDK_KEY_braille_dots_345678 0x10028fc
#define GDK_KEY_braille_dots_1345678 0x10028fd
#define GDK_KEY_braille_dots_2345678 0x10028fe
#define GDK_KEY_braille_dots_12345678 0x10028ff
#define GDK_KEY_Switch_VT_1 0x1008fe01
#define GDK_KEY_Switch_VT_2 0x1008fe02
#define GDK_KEY_Switch_VT_3 0x1008fe03
#define GDK_KEY_Switch_VT_4 0x1008fe04
#define GDK_KEY_Switch_VT_5 0x1008fe05
#define GDK_KEY_Switch_VT_6 0x1008fe06
#define GDK_KEY_Switch_VT_7 0x1008fe07
#define GDK_KEY_Switch_VT_8 0x1008fe08
#define GDK_KEY_Switch_VT_9 0x1008fe09
#define GDK_KEY_Switch_VT_10 0x1008fe0a
#define GDK_KEY_Switch_VT_11 0x1008fe0b
#define GDK_KEY_Switch_VT_12 0x1008fe0c
#define GDK_KEY_Ungrab 0x1008fe20
#define GDK_KEY_ClearGrab 0x1008fe21
#define GDK_KEY_Next_VMode 0x1008fe22
#define GDK_KEY_Prev_VMode 0x1008fe23
#define GDK_KEY_LogWindowTree 0x1008fe24
#define GDK_KEY_LogGrabInfo 0x1008fe25
#define GDK_KEY_ModeLock 0x1008ff01
#define GDK_KEY_MonBrightnessUp 0x1008ff02
#define GDK_KEY_MonBrightnessDown 0x1008ff03
#define GDK_KEY_KbdLightOnOff 0x1008ff04
#define GDK_KEY_KbdBrightnessUp 0x1008ff05
#define GDK_KEY_KbdBrightnessDown 0x1008ff06
#define GDK_KEY_Standby 0x1008ff10
#define GDK_KEY_AudioLowerVolume 0x1008ff11
#define GDK_KEY_AudioMute 0x1008ff12
#define GDK_KEY_AudioRaiseVolume 0x1008ff13
#define GDK_KEY_AudioPlay 0x1008ff14
#define GDK_KEY_AudioStop 0x1008ff15
#define GDK_KEY_AudioPrev 0x1008ff16
#define GDK_KEY_AudioNext 0x1008ff17
#define GDK_KEY_HomePage 0x1008ff18
#define GDK_KEY_Mail 0x1008ff19
#define GDK_KEY_Start 0x1008ff1a
#define GDK_KEY_Search 0x1008ff1b
#define GDK_KEY_AudioRecord 0x1008ff1c
#define GDK_KEY_Calculator 0x1008ff1d
#define GDK_KEY_Memo 0x1008ff1e
#define GDK_KEY_ToDoList 0x1008ff1f
#define GDK_KEY_Calendar 0x1008ff20
#define GDK_KEY_PowerDown 0x1008ff21
#define GDK_KEY_ContrastAdjust 0x1008ff22
#define GDK_KEY_RockerUp 0x1008ff23
#define GDK_KEY_RockerDown 0x1008ff24
#define GDK_KEY_RockerEnter 0x1008ff25
#define GDK_KEY_Back 0x1008ff26
#define GDK_KEY_Forward 0x1008ff27
#define GDK_KEY_Stop 0x1008ff28
#define GDK_KEY_Refresh 0x1008ff29
#define GDK_KEY_PowerOff 0x1008ff2a
#define GDK_KEY_WakeUp 0x1008ff2b
#define GDK_KEY_Eject 0x1008ff2c

```



```

#define GDK_KEY_ScreenSaver 0x1008ff2d
#define GDK_KEY_WWW 0x1008ff2e
#define GDK_KEY_Sleep 0x1008ff2f
#define GDK_KEY_Favorites 0x1008ff30
#define GDK_KEY_AudioPause 0x1008ff31
#define GDK_KEY_AudioMedia 0x1008ff32
#define GDK_KEY_MyComputer 0x1008ff33
#define GDK_KEY_VendorHome 0x1008ff34
#define GDK_KEY_LightBulb 0x1008ff35
#define GDK_KEY_Shop 0x1008ff36
#define GDK_KEY_History 0x1008ff37
#define GDK_KEY_OpenURL 0x1008ff38
#define GDK_KEY_AddFavorite 0x1008ff39
#define GDK_KEY_HotLinks 0x1008ff3a
#define GDK_KEY_BrightnessAdjust 0x1008ff3b
#define GDK_KEY_Finance 0x1008ff3c
#define GDK_KEY_Community 0x1008ff3d
#define GDK_KEY_AudioRewind 0x1008ff3e
#define GDK_KEY_BackForward 0x1008ff3f
#define GDK_KEY_Launch0 0x1008ff40
#define GDK_KEY_Launch1 0x1008ff41
#define GDK_KEY_Launch2 0x1008ff42
#define GDK_KEY_Launch3 0x1008ff43
#define GDK_KEY_Launch4 0x1008ff44
#define GDK_KEY_Launch5 0x1008ff45
#define GDK_KEY_Launch6 0x1008ff46
#define GDK_KEY_Launch7 0x1008ff47
#define GDK_KEY_Launch8 0x1008ff48
#define GDK_KEY_Launch9 0x1008ff49
#define GDK_KEY_LaunchA 0x1008ff4a
#define GDK_KEY_LaunchB 0x1008ff4b
#define GDK_KEY_LaunchC 0x1008ff4c
#define GDK_KEY_LaunchD 0x1008ff4d
#define GDK_KEY_LaunchE 0x1008ff4e
#define GDK_KEY_LaunchF 0x1008ff4f
#define GDK_KEY_ApplicationLeft 0x1008ff50
#define GDK_KEY_ApplicationRight 0x1008ff51
#define GDK_KEY_Book 0x1008ff52
#define GDK_KEY_CD 0x1008ff53
#define GDK_KEY_WindowClear 0x1008ff55
#define GDK_KEY_Close 0x1008ff56
#define GDK_KEY_Copy 0x1008ff57
#define GDK_KEY_Cut 0x1008ff58
#define GDK_KEY_Display 0x1008ff59
#define GDK_KEY_DOS 0x1008ff5a
#define GDK_KEY_Documents 0x1008ff5b
#define GDK_KEY_Excel 0x1008ff5c
#define GDK_KEY_Explorer 0x1008ff5d
#define GDK_KEY_Game 0x1008ff5e
#define GDK_KEY_Go 0x1008ff5f
#define GDK_KEY_iTouch 0x1008ff60
#define GDK_KEY_LogOff 0x1008ff61
#define GDK_KEY_Market 0x1008ff62
#define GDK_KEY_Meeting 0x1008ff63
#define GDK_KEY_MenuKB 0x1008ff65
#define GDK_KEY_MenuPB 0x1008ff66
#define GDK_KEY_MySites 0x1008ff67
#define GDK_KEY_New 0x1008ff68
#define GDK_KEY_News 0x1008ff69
#define GDK_KEY_OfficeHome 0x1008ff6a
#define GDK_KEY_Open 0x1008ff6b
#define GDK_KEY_Option 0x1008ff6c
#define GDK_KEY_Paste 0x1008ff6d
#define GDK_KEY_Phone 0x1008ff6e
#define GDK_KEY_Reply 0x1008ff72
#define GDK_KEY_Reload 0x1008ff73

```

```

#define GDK_KEY_RotateWindows 0x1008ff74
#define GDK_KEY_RotationPB 0x1008ff75
#define GDK_KEY_RotationKB 0x1008ff76
#define GDK_KEY_Save 0x1008ff77
#define GDK_KEY_ScrollUp 0x1008ff78
#define GDK_KEY_ScrollDown 0x1008ff79
#define GDK_KEY_ScrollClick 0x1008ff7a
#define GDK_KEY_Send 0x1008ff7b
#define GDK_KEY_Spell 0x1008ff7c
#define GDK_KEY_SplitScreen 0x1008ff7d
#define GDK_KEY_Support 0x1008ff7e
#define GDK_KEY_TaskPane 0x1008ff7f
#define GDK_KEY_Terminal 0x1008ff80
#define GDK_KEY_Tools 0x1008ff81
#define GDK_KEY_Travel 0x1008ff82
#define GDK_KEY_UserPB 0x1008ff84
#define GDK_KEY_User1KB 0x1008ff85
#define GDK_KEY_User2KB 0x1008ff86
#define GDK_KEY_Video 0x1008ff87
#define GDK_KEY_WheelButton 0x1008ff88
#define GDK_KEY_Word 0x1008ff89
#define GDK_KEY_Xfer 0x1008ff8a
#define GDK_KEY_ZoomIn 0x1008ff8b
#define GDK_KEY_ZoomOut 0x1008ff8c
#define GDK_KEY_Away 0x1008ff8d
#define GDK_KEY_Messenger 0x1008ff8e
#define GDK_KEY_WebCam 0x1008ff8f
#define GDK_KEY_MailForward 0x1008ff90
#define GDK_KEY_Pictures 0x1008ff91
#define GDK_KEY_Music 0x1008ff92
#define GDK_KEY_Battery 0x1008ff93
#define GDK_KEY_Bluetooth 0x1008ff94
#define GDK_KEY_WLAN 0x1008ff95
#define GDK_KEY_UWB 0x1008ff96
#define GDK_KEY_AudioForward 0x1008ff97
#define GDK_KEY_AudioRepeat 0x1008ff98
#define GDK_KEY_AudioRandomPlay 0x1008ff99
#define GDK_KEY_Subtitle 0x1008ff9a
#define GDK_KEY_AudioCycleTrack 0x1008ff9b
#define GDK_KEY_CycleAngle 0x1008ff9c
#define GDK_KEY_FrameBack 0x1008ff9d
#define GDK_KEY_FrameForward 0x1008ff9e
#define GDK_KEY_Fime 0x1008ff9f
#define GDK_KEY_SelectButton 0x1008ffa0
#define GDK_KEY_View 0x1008ffa1
#define GDK_KEY_TopMenu 0x1008ffa2
#define GDK_KEY_Red 0x1008ffa3
#define GDK_KEY_Green 0x1008ffa4
#define GDK_KEY_Yellow 0x1008ffa5
#define GDK_KEY_Blue 0x1008ffa6
#define GDK_KEY_Suspend 0x1008ffa7
#define GDK_KEY_Hibernate 0x1008ffa8
#define GDK_KEY_TouchpadToggle 0x1008ffa9
#define GDK_KEY_TouchpadOn 0x1008ffb0
#define GDK_KEY_TouchpadOff 0x1008ffb1
#define GDK_KEY_OE 0x13bc
#define GDK_KEY_oe 0x13bd
#define GDK_KEY_Ydiaeresis 0x13be
#define GDK_KEY_Aogonek 0x1a1
#define GDK_KEY_breve 0x1a2
#define GDK_KEY_Lstroke 0x1a3
#define GDK_KEY_Lcaron 0x1a5
#define GDK_KEY_Sacute 0x1a6
#define GDK_KEY_Scaron 0x1a9
#define GDK_KEY_Scedilla 0x1aa
#define GDK_KEY_Tcaron 0x1ab

```

```

#define GDK_KEY_Zacute 0x1ac
#define GDK_KEY_Zcaron 0x1ae
#define GDK_KEY_Zabovedot 0x1af
#define GDK_KEY_aogonek 0x1b1
#define GDK_KEY_ogonek 0x1b2
#define GDK_KEY_lstroke 0x1b3
#define GDK_KEY_lcaron 0x1b5
#define GDK_KEY_sacute 0x1b6
#define GDK_KEY_caron 0x1b7
#define GDK_KEY_scaron 0x1b9
#define GDK_KEY_scedilla 0x1ba
#define GDK_KEY_tcaron 0x1bb
#define GDK_KEY_zacute 0x1bc
#define GDK_KEY_doubleacute 0x1bd
#define GDK_KEY_zcaron 0x1be
#define GDK_KEY_zabovedot 0x1bf
#define GDK_KEY_Racute 0x1c0
#define GDK_KEY_Abreve 0x1c3
#define GDK_KEY_Lacute 0x1c5
#define GDK_KEY_Cacute 0x1c6
#define GDK_KEY_Ccaron 0x1c8
#define GDK_KEY_Eogonek 0x1ca
#define GDK_KEY_Ecaron 0x1cc
#define GDK_KEY_Dcaron 0x1cf
#define GDK_KEY_Dstroke 0x1d0
#define GDK_KEY_Nacute 0x1d1
#define GDK_KEY_Ncaron 0x1d2
#define GDK_KEY_Odoubleacute 0x1d5
#define GDK_KEY_Rcaron 0x1d8
#define GDK_KEY_Uring 0x1d9
#define GDK_KEY_Udoubleacute 0x1db
#define GDK_KEY_Tcedilla 0x1de
#define GDK_KEY_racute 0x1e0
#define GDK_KEY_abreve 0x1e3
#define GDK_KEY_lacute 0x1e5
#define GDK_KEY_cacute 0x1e6
#define GDK_KEY_ccaron 0x1e8
#define GDK_KEY_eogonek 0x1ea
#define GDK_KEY_ecaron 0x1ec
#define GDK_KEY_dcaron 0x1ef
#define GDK_KEY_dstroke 0x1f0
#define GDK_KEY_nacute 0x1f1
#define GDK_KEY_ncaron 0x1f2
#define GDK_KEY_odoubleacute 0x1f5
#define GDK_KEY_rcaron 0x1f8
#define GDK_KEY_uring 0x1f9
#define GDK_KEY_udoubleacute 0x1fb
#define GDK_KEY_tcedilla 0x1fe
#define GDK_KEY_abovedot 0x1ff
#define GDK_KEY_EuroSign 0x20ac
#define GDK_KEY_Hstroke 0x2a1
#define GDK_KEY_Hcircumflex 0x2a6
#define GDK_KEY_Iabovedot 0x2a9
#define GDK_KEY_Gbreve 0x2ab
#define GDK_KEY_Jcircumflex 0x2ac
#define GDK_KEY_hstroke 0x2b1
#define GDK_KEY_hcircumflex 0x2b6
#define GDK_KEY_idotless 0x2b9
#define GDK_KEY_gbreve 0x2bb
#define GDK_KEY_jcircumflex 0x2bc
#define GDK_KEY_Cabovedot 0x2c5
#define GDK_KEY_Ccircumflex 0x2c6
#define GDK_KEY_Gabovedot 0x2d5
#define GDK_KEY_Gcircumflex 0x2d8
#define GDK_KEY_Ubreve 0x2dd
#define GDK_KEY_Scircumflex 0x2de

```

```

#define GDK_KEY_cabovedot 0x2e5
#define GDK_KEY_ccircumflex 0x2e6
#define GDK_KEY_gabovedot 0x2f5
#define GDK_KEY_gcircumflex 0x2f8
#define GDK_KEY_ubreve 0x2fd
#define GDK_KEY_scircumflex 0x2fe
#define GDK_KEY_kappa 0x3a2
#define GDK_KEY_kra 0x3a2
#define GDK_KEY_Rcedilla 0x3a3
#define GDK_KEY_Itilde 0x3a5
#define GDK_KEY_Lcedilla 0x3a6
#define GDK_KEY_Emacron 0x3aa
#define GDK_KEY_Gcedilla 0x3ab
#define GDK_KEY_Tslash 0x3ac
#define GDK_KEY_rcedilla 0x3b3
#define GDK_KEY_itylde 0x3b5
#define GDK_KEY_lcedilla 0x3b6
#define GDK_KEY_emacron 0x3ba
#define GDK_KEY_gcedilla 0x3bb
#define GDK_KEY_tslash 0x3bc
#define GDK_KEY_ENG 0x3bd
#define GDK_KEY_eng 0x3bf
#define GDK_KEY_Amacron 0x3c0
#define GDK_KEY_Iogonek 0x3c7
#define GDK_KEY_Eabovedot 0x3cc
#define GDK_KEY_Imacron 0x3cf
#define GDK_KEY_Ncedilla 0x3d1
#define GDK_KEY_Omacron 0x3d2
#define GDK_KEY_Kcedilla 0x3d3
#define GDK_KEY_Uogonek 0x3d9
#define GDK_KEY_Utilde 0x3dd
#define GDK_KEY_Umacron 0x3de
#define GDK_KEY_amacron 0x3e0
#define GDK_KEY_iogonek 0x3e7
#define GDK_KEY_eabovedot 0x3ec
#define GDK_KEY_imacron 0x3ef
#define GDK_KEY_ncedilla 0x3f1
#define GDK_KEY_omacron 0x3f2
#define GDK_KEY_kcedilla 0x3f3
#define GDK_KEY_uogonek 0x3f9
#define GDK_KEY_utyilde 0x3fd
#define GDK_KEY_umacron 0x3fe
#define GDK_KEY_overline 0x47e
#define GDK_KEY_kana_fullstop 0x4a1
#define GDK_KEY_kana_openingbracket 0x4a2
#define GDK_KEY_kana_closingbracket 0x4a3
#define GDK_KEY_kana_comma 0x4a4
#define GDK_KEY_kana_conjunctive 0x4a5
#define GDK_KEY_kana_middledot 0x4a5
#define GDK_KEY_kana_WO 0x4a6
#define GDK_KEY_kana_a 0x4a7
#define GDK_KEY_kana_i 0x4a8
#define GDK_KEY_kana_u 0x4a9
#define GDK_KEY_kana_e 0x4aa
#define GDK_KEY_kana_o 0x4ab
#define GDK_KEY_kana_ya 0x4ac
#define GDK_KEY_kana_yu 0x4ad
#define GDK_KEY_kana_yo 0x4ae
#define GDK_KEY_kana_tsu 0x4af
#define GDK_KEY_kana_tu 0x4af
#define GDK_KEY_prolongedsound 0x4b0
#define GDK_KEY_kana_A 0x4b1
#define GDK_KEY_kana_I 0x4b2
#define GDK_KEY_kana_U 0x4b3
#define GDK_KEY_kana_E 0x4b4
#define GDK_KEY_kana_O 0x4b5

```

```

#define GDK_KEY_kana_KA 0x4b6
#define GDK_KEY_kana_KI 0x4b7
#define GDK_KEY_kana_KU 0x4b8
#define GDK_KEY_kana_KE 0x4b9
#define GDK_KEY_kana_KO 0x4ba
#define GDK_KEY_kana_SA 0x4bb
#define GDK_KEY_kana_SHI 0x4bc
#define GDK_KEY_kana_SU 0x4bd
#define GDK_KEY_kana_SE 0x4be
#define GDK_KEY_kana_SO 0x4bf
#define GDK_KEY_kana_TA 0x4c0
#define GDK_KEY_kana_CHI 0x4c1
#define GDK_KEY_kana_TI 0x4c1
#define GDK_KEY_kana_TSU 0x4c2
#define GDK_KEY_kana_TU 0x4c2
#define GDK_KEY_kana_TE 0x4c3
#define GDK_KEY_kana_TO 0x4c4
#define GDK_KEY_kana_NA 0x4c5
#define GDK_KEY_kana_NI 0x4c6
#define GDK_KEY_kana_NU 0x4c7
#define GDK_KEY_kana_NE 0x4c8
#define GDK_KEY_kana_NO 0x4c9
#define GDK_KEY_kana_HA 0x4ca
#define GDK_KEY_kana_HI 0x4cb
#define GDK_KEY_kana_FU 0x4cc
#define GDK_KEY_kana_HU 0x4cc
#define GDK_KEY_kana_HE 0x4cd
#define GDK_KEY_kana_HO 0x4ce
#define GDK_KEY_kana_MA 0x4cf
#define GDK_KEY_kana_MI 0x4d0
#define GDK_KEY_kana_MU 0x4d1
#define GDK_KEY_kana_ME 0x4d2
#define GDK_KEY_kana_MO 0x4d3
#define GDK_KEY_kana_YA 0x4d4
#define GDK_KEY_kana_YU 0x4d5
#define GDK_KEY_kana_YO 0x4d6
#define GDK_KEY_kana_RA 0x4d7
#define GDK_KEY_kana_RI 0x4d8
#define GDK_KEY_kana_RU 0x4d9
#define GDK_KEY_kana_RE 0x4da
#define GDK_KEY_kana_RO 0x4db
#define GDK_KEY_kana_WA 0x4dc
#define GDK_KEY_kana_N 0x4dd
#define GDK_KEY_voicedsound 0x4de
#define GDK_KEY_semivoicedsound 0x4df
#define GDK_KEY_Arabic_comma 0x5ac
#define GDK_KEY_Arabic_semicolon 0x5bb
#define GDK_KEY_Arabic_question_mark 0x5bf
#define GDK_KEY_Arabic_hamza 0x5c1
#define GDK_KEY_Arabic_maddaonalef 0x5c2
#define GDK_KEY_Arabic_hamzaonalef 0x5c3
#define GDK_KEY_Arabic_hamzaonwaw 0x5c4
#define GDK_KEY_Arabic_hamzaunderalef 0x5c5
#define GDK_KEY_Arabic_hamzaonyeh 0x5c6
#define GDK_KEY_Arabic_alef 0x5c7
#define GDK_KEY_Arabic_beh 0x5c8
#define GDK_KEY_Arabic_tehmarbuta 0x5c9
#define GDK_KEY_Arabic_teh 0x5ca
#define GDK_KEY_Arabic_theh 0x5cb
#define GDK_KEY_Arabic_jeem 0x5cc
#define GDK_KEY_Arabic_hah 0x5cd
#define GDK_KEY_Arabic_khah 0x5ce
#define GDK_KEY_Arabic_dal 0x5cf
#define GDK_KEY_Arabic_thal 0x5d0
#define GDK_KEY_Arabic_ra 0x5d1
#define GDK_KEY_Arabic_zain 0x5d2

```

```

#define GDK_KEY_Arabic_seen      0x5d3
#define GDK_KEY_Arabic_sheen    0x5d4
#define GDK_KEY_Arabic_sad      0x5d5
#define GDK_KEY_Arabic_dad      0x5d6
#define GDK_KEY_Arabic_tah      0x5d7
#define GDK_KEY_Arabic_zah      0x5d8
#define GDK_KEY_Arabic_ain      0x5d9
#define GDK_KEY_Arabic_ghain    0x5da
#define GDK_KEY_Arabic_tatweel  0x5e0
#define GDK_KEY_Arabic_feh      0x5e1
#define GDK_KEY_Arabic_qaf      0x5e2
#define GDK_KEY_Arabic_kaf      0x5e3
#define GDK_KEY_Arabic_lam      0x5e4
#define GDK_KEY_Arabic_meem      0x5e5
#define GDK_KEY_Arabic_noon      0x5e6
#define GDK_KEY_Arabic_ha       0x5e7
#define GDK_KEY_Arabic_heh      0x5e7
#define GDK_KEY_Arabic_waw      0x5e8
#define GDK_KEY_Arabic_alefmaksura 0x5e9
#define GDK_KEY_Arabic_yeh      0x5ea
#define GDK_KEY_Arabic_fathatan 0x5eb
#define GDK_KEY_Arabic_dammatan 0x5ec
#define GDK_KEY_Arabic_kasratan 0x5ed
#define GDK_KEY_Arabic_fatha    0x5ee
#define GDK_KEY_Arabic_damma    0x5ef
#define GDK_KEY_Arabic_kasra    0x5f0
#define GDK_KEY_Arabic_shadda   0x5f1
#define GDK_KEY_Arabic_sukun    0x5f2
#define GDK_KEY_Serbian_dje     0x6a1
#define GDK_KEY_Macedonia_gje   0x6a2
#define GDK_KEY_Cyrillic_io     0x6a3
#define GDK_KEY_Ukrainian_ie    0x6a4
#define GDK_KEY_Ukranian_je     0x6a4
#define GDK_KEY_Macedonia_dse   0x6a5
#define GDK_KEY_Ukrainian_i     0x6a6
#define GDK_KEY_Ukranian_i      0x6a6
#define GDK_KEY_Ukrainian_yi    0x6a7
#define GDK_KEY_Ukranian_yi     0x6a7
#define GDK_KEY_Cyrillic_je     0x6a8
#define GDK_KEY_Serbian_je      0x6a8
#define GDK_KEY_Cyrillic_lje    0x6a9
#define GDK_KEY_Serbian_lje     0x6a9
#define GDK_KEY_Cyrillic_nje    0x6aa
#define GDK_KEY_Serbian_nje     0x6aa
#define GDK_KEY_Serbian_tshe    0x6ab
#define GDK_KEY_Macedonia_kje   0x6ac
#define GDK_KEY_Ukrainian_ghe_with_upturn 0x6ad
#define GDK_KEY_Byelorussian_shortu 0x6ae
#define GDK_KEY_Cyrillic_dzhe   0x6af
#define GDK_KEY_Serbian_dze     0x6af
#define GDK_KEY_numerosign      0x6b0
#define GDK_KEY_Serbian_DJE     0x6b1
#define GDK_KEY_Macedonia_GJE   0x6b2
#define GDK_KEY_Cyrillic_IO     0x6b3
#define GDK_KEY_Ukrainian_IE    0x6b4
#define GDK_KEY_Ukranian_JE     0x6b4
#define GDK_KEY_Macedonia_DSE   0x6b5
#define GDK_KEY_Ukrainian_I     0x6b6
#define GDK_KEY_Ukranian_I      0x6b6
#define GDK_KEY_Ukrainian_YI    0x6b7
#define GDK_KEY_Ukranian_YI     0x6b7
#define GDK_KEY_Cyrillic_JE     0x6b8
#define GDK_KEY_Serbian_JE      0x6b8
#define GDK_KEY_Cyrillic_LJE    0x6b9
#define GDK_KEY_Serbian_LJE     0x6b9
#define GDK_KEY_Cyrillic_NJE    0x6ba

```

```

#define GDK_KEY_Serbian_NJE      0x6ba
#define GDK_KEY_Serbian_TSHE     0x6bb
#define GDK_KEY_Macedonia_KJE    0x6bc
#define GDK_KEY_Ukrainian_GHE_WITH_UPTURN 0x6bd
#define GDK_KEY_Byelorussian_SHORTU 0x6be
#define GDK_KEY_Cyrillic_DZHE    0x6bf
#define GDK_KEY_Serbian_DZE      0x6bf
#define GDK_KEY_Cyrillic_yu      0x6c0
#define GDK_KEY_Cyrillic_a       0x6c1
#define GDK_KEY_Cyrillic_be      0x6c2
#define GDK_KEY_Cyrillic_tse     0x6c3
#define GDK_KEY_Cyrillic_de      0x6c4
#define GDK_KEY_Cyrillic_ie      0x6c5
#define GDK_KEY_Cyrillic_ef      0x6c6
#define GDK_KEY_Cyrillic_ghe     0x6c7
#define GDK_KEY_Cyrillic_ha      0x6c8
#define GDK_KEY_Cyrillic_i       0x6c9
#define GDK_KEY_Cyrillic_shorti  0x6ca
#define GDK_KEY_Cyrillic_ka      0x6cb
#define GDK_KEY_Cyrillic_el      0x6cc
#define GDK_KEY_Cyrillic_em      0x6cd
#define GDK_KEY_Cyrillic_en      0x6ce
#define GDK_KEY_Cyrillic_o       0x6cf
#define GDK_KEY_Cyrillic_pe      0x6d0
#define GDK_KEY_Cyrillic_ya      0x6d1
#define GDK_KEY_Cyrillic_er      0x6d2
#define GDK_KEY_Cyrillic_es      0x6d3
#define GDK_KEY_Cyrillic_te      0x6d4
#define GDK_KEY_Cyrillic_u       0x6d5
#define GDK_KEY_Cyrillic_zhe     0x6d6
#define GDK_KEY_Cyrillic_ve      0x6d7
#define GDK_KEY_Cyrillic_softsign 0x6d8
#define GDK_KEY_Cyrillic_yeru    0x6d9
#define GDK_KEY_Cyrillic_ze      0x6da
#define GDK_KEY_Cyrillic_sha     0x6db
#define GDK_KEY_Cyrillic_e       0x6dc
#define GDK_KEY_Cyrillic_shcha   0x6dd
#define GDK_KEY_Cyrillic_che     0x6de
#define GDK_KEY_Cyrillic_hardsign 0x6df
#define GDK_KEY_Cyrillic_YU      0x6e0
#define GDK_KEY_Cyrillic_A       0x6e1
#define GDK_KEY_Cyrillic_BE      0x6e2
#define GDK_KEY_Cyrillic_TSE     0x6e3
#define GDK_KEY_Cyrillic_DE      0x6e4
#define GDK_KEY_Cyrillic_IE      0x6e5
#define GDK_KEY_Cyrillic_EF      0x6e6
#define GDK_KEY_Cyrillic_GHE     0x6e7
#define GDK_KEY_Cyrillic_HA      0x6e8
#define GDK_KEY_Cyrillic_I       0x6e9
#define GDK_KEY_Cyrillic_SHORTI  0x6ea
#define GDK_KEY_Cyrillic_KA      0x6eb
#define GDK_KEY_Cyrillic_EL      0x6ec
#define GDK_KEY_Cyrillic_EM      0x6ed
#define GDK_KEY_Cyrillic_EN      0x6ee
#define GDK_KEY_Cyrillic_O       0x6ef
#define GDK_KEY_Cyrillic_PE      0x6f0
#define GDK_KEY_Cyrillic_YA      0x6f1
#define GDK_KEY_Cyrillic_ER      0x6f2
#define GDK_KEY_Cyrillic_ES      0x6f3
#define GDK_KEY_Cyrillic_TE      0x6f4
#define GDK_KEY_Cyrillic_U       0x6f5
#define GDK_KEY_Cyrillic_ZHE     0x6f6
#define GDK_KEY_Cyrillic_VE      0x6f7
#define GDK_KEY_Cyrillic_SOFTSIGN 0x6f8
#define GDK_KEY_Cyrillic_YERU    0x6f9
#define GDK_KEY_Cyrillic_ZE      0x6fa

```



```

#define GDK_KEY_Cyrillic_SHA      0x6fb
#define GDK_KEY_Cyrillic_E        0x6fc
#define GDK_KEY_Cyrillic_SHCHA    0x6fd
#define GDK_KEY_Cyrillic_CHE      0x6fe
#define GDK_KEY_Cyrillic_HARDSIGN 0x6ff
#define GDK_KEY_Greek_ALPHAaccent 0x7a1
#define GDK_KEY_Greek_EPSILONaccent 0x7a2
#define GDK_KEY_Greek_ETAaccent 0x7a3
#define GDK_KEY_Greek_IOTAaccent 0x7a4
#define GDK_KEY_Greek_IOTAdiaeresis 0x7a5
#define GDK_KEY_Greek_IOTAdieresis 0x7a5
#define GDK_KEY_Greek_OMICRONaccent 0x7a7
#define GDK_KEY_Greek_UPSILONaccent 0x7a8
#define GDK_KEY_Greek_UPSILONdieresis 0x7a9
#define GDK_KEY_Greek_OMEGAaccent 0x7ab
#define GDK_KEY_Greek_accentdieresis 0x7ae
#define GDK_KEY_Greek_horizbar 0x7af
#define GDK_KEY_Greek_alphaaccent 0x7b1
#define GDK_KEY_Greek_epsilonaccent 0x7b2
#define GDK_KEY_Greek_etaaccent 0x7b3
#define GDK_KEY_Greek_iotaaccent 0x7b4
#define GDK_KEY_Greek_iotadieresis 0x7b5
#define GDK_KEY_Greek_iotaaccentdieresis 0x7b6
#define GDK_KEY_Greek_omicronaccent 0x7b7
#define GDK_KEY_Greek_upsilonaccent 0x7b8
#define GDK_KEY_Greek_upsilondieresis 0x7b9
#define GDK_KEY_Greek_upsilonaccentdieresis 0x7ba
#define GDK_KEY_Greek_omegaaccent 0x7bb
#define GDK_KEY_Greek_ALPHA 0x7c1
#define GDK_KEY_Greek_BETA 0x7c2
#define GDK_KEY_Greek_GAMMA 0x7c3
#define GDK_KEY_Greek_DELTA 0x7c4
#define GDK_KEY_Greek_EPSILON 0x7c5
#define GDK_KEY_Greek_ZETA 0x7c6
#define GDK_KEY_Greek_ETA 0x7c7
#define GDK_KEY_Greek_THETA 0x7c8
#define GDK_KEY_Greek_IOTA 0x7c9
#define GDK_KEY_Greek_KAPPA 0x7ca
#define GDK_KEY_Greek_LAMBDA 0x7cb
#define GDK_KEY_Greek_LAMDA 0x7cb
#define GDK_KEY_Greek_MU 0x7cc
#define GDK_KEY_Greek_NU 0x7cd
#define GDK_KEY_Greek_XI 0x7ce
#define GDK_KEY_Greek_OMICRON 0x7cf
#define GDK_KEY_Greek_PI 0x7d0
#define GDK_KEY_Greek_RHO 0x7d1
#define GDK_KEY_Greek_SIGMA 0x7d2
#define GDK_KEY_Greek_TAU 0x7d4
#define GDK_KEY_Greek_UPSILON 0x7d5
#define GDK_KEY_Greek_PHI 0x7d6
#define GDK_KEY_Greek_CHI 0x7d7
#define GDK_KEY_Greek_PSI 0x7d8
#define GDK_KEY_Greek_OMEGA 0x7d9
#define GDK_KEY_Greek_alpha 0x7e1
#define GDK_KEY_Greek_beta 0x7e2
#define GDK_KEY_Greek_gamma 0x7e3
#define GDK_KEY_Greek_delta 0x7e4
#define GDK_KEY_Greek_epsilon 0x7e5
#define GDK_KEY_Greek_zeta 0x7e6
#define GDK_KEY_Greek_eta 0x7e7
#define GDK_KEY_Greek_theta 0x7e8
#define GDK_KEY_Greek_iota 0x7e9
#define GDK_KEY_Greek_kappa 0x7ea
#define GDK_KEY_Greek_lambda 0x7eb
#define GDK_KEY_Greek_lamda 0x7eb
#define GDK_KEY_Greek_mu 0x7ec

```



```

#define GDK_KEY_Greek_nu      0x7ed
#define GDK_KEY_Greek_xi      0x7ee
#define GDK_KEY_Greek_omicron 0x7ef
#define GDK_KEY_Greek_pi      0x7f0
#define GDK_KEY_Greek_rho      0x7f1
#define GDK_KEY_Greek_sigma    0x7f2
#define GDK_KEY_Greek_finalsmallsigma 0x7f3
#define GDK_KEY_Greek_tau      0x7f4
#define GDK_KEY_Greek_upsilon  0x7f5
#define GDK_KEY_Greek_phi      0x7f6
#define GDK_KEY_Greek_chi      0x7f7
#define GDK_KEY_Greek_psi      0x7f8
#define GDK_KEY_Greek_omega    0x7f9
#define GDK_KEY_leftradical    0x8a1
#define GDK_KEY_topleftradical 0x8a2
#define GDK_KEY_horizconnector 0x8a3
#define GDK_KEY_topintegral     0x8a4
#define GDK_KEY_botintegral     0x8a5
#define GDK_KEY_vertconnector   0x8a6
#define GDK_KEY_topleftsqbracket 0x8a7
#define GDK_KEY_botleftsqbracket 0x8a8
#define GDK_KEY_toprightsqbracket 0x8a9
#define GDK_KEY_botrightsqbracket 0x8aa
#define GDK_KEY_topleftparens  0x8ab
#define GDK_KEY_botleftparens  0x8ac
#define GDK_KEY_toprightparens 0x8ad
#define GDK_KEY_botrightparens 0x8ae
#define GDK_KEY_leftmiddlecurlybrace 0x8af
#define GDK_KEY_rightmiddlecurlybrace 0x8b0
#define GDK_KEY_topleftsummation 0x8b1
#define GDK_KEY_botleftsummation 0x8b2
#define GDK_KEY_topvertsummationconnector 0x8b3
#define GDK_KEY_botvertsummationconnector 0x8b4
#define GDK_KEY_toprightsummation 0x8b5
#define GDK_KEY_botrightsummation 0x8b6
#define GDK_KEY_rightmiddlesummation 0x8b7
#define GDK_KEY_lessthanequal 0x8bc
#define GDK_KEY_notequal 0x8bd
#define GDK_KEY_greaterequal 0x8be
#define GDK_KEY_integral 0x8bf
#define GDK_KEY_therefore 0x8c0
#define GDK_KEY_variation 0x8c1
#define GDK_KEY_infinity 0x8c2
#define GDK_KEY_nabla 0x8c5
#define GDK_KEY_approximate 0x8c8
#define GDK_KEY_similarequal 0x8c9
#define GDK_KEY_ifonlyif 0x8cd
#define GDK_KEY_implies 0x8ce
#define GDK_KEY_identical 0x8cf
#define GDK_KEY_radical 0x8d6
#define GDK_KEY_includedin 0x8da
#define GDK_KEY_includes 0x8db
#define GDK_KEY_intersection 0x8dc
#define GDK_KEY_union 0x8dd
#define GDK_KEY_logicaland 0x8de
#define GDK_KEY_logicalor 0x8df
#define GDK_KEY_partialderivative 0x8ef
#define GDK_KEY_function 0x8f6
#define GDK_KEY_leftarrow 0x8fb
#define GDK_KEY_uparrow 0x8fc
#define GDK_KEY_rightarrow 0x8fd
#define GDK_KEY_downdarrow 0x8fe
#define GDK_KEY_blank 0x9df
#define GDK_KEY_soliddiamond 0x9e0
#define GDK_KEY_checkerboard 0x9e1
#define GDK_KEY_ht 0x9e2

```

```

#define GDK_KEY_ff      0x9e3
#define GDK_KEY_cr      0x9e4
#define GDK_KEY_lf      0x9e5
#define GDK_KEY_nl      0x9e8
#define GDK_KEY_vt      0x9e9
#define GDK_KEY_lowrightcorner 0x9ea
#define GDK_KEY_uprightcorner 0x9eb
#define GDK_KEY_upleftcorner 0x9ec
#define GDK_KEY_lowleftcorner 0x9ed
#define GDK_KEY_crossinglines 0x9ee
#define GDK_KEY_horizlinescan1 0x9ef
#define GDK_KEY_horizlinescan3 0x9f0
#define GDK_KEY_horizlinescan5 0x9f1
#define GDK_KEY_horizlinescan7 0x9f2
#define GDK_KEY_horizlinescan9 0x9f3
#define GDK_KEY_leftt 0x9f4
#define GDK_KEY_rightt 0x9f5
#define GDK_KEY_bott 0x9f6
#define GDK_KEY_topt 0x9f7
#define GDK_KEY_vertbar 0x9f8
#define GDK_KEY_emspace 0xaa1
#define GDK_KEY_enspace 0xaa2
#define GDK_KEY_em3space 0xaa3
#define GDK_KEY_em4space 0xaa4
#define GDK_KEY_digitspace 0xaa5
#define GDK_KEY_punctspace 0xaa6
#define GDK_KEY_thinspace 0xaa7
#define GDK_KEY_hairspace 0xaa8
#define GDK_KEY_emdash 0xaa9
#define GDK_KEY_endash 0aaa
#define GDK_KEY_signifblank 0aac
#define GDK_KEY_ellipsis 0aae
#define GDK_KEY_doubbaselinedot 0aaef
#define GDK_KEY_onethird 0xab0
#define GDK_KEY_twothirds 0xab1
#define GDK_KEY_onefifth 0xab2
#define GDK_KEY_twofifths 0xab3
#define GDK_KEY_threefifths 0xab4
#define GDK_KEY_fourfifths 0xab5
#define GDK_KEY_onesixth 0xab6
#define GDK_KEY_fivesixths 0xab7
#define GDK_KEY_careof 0xab8
#define GDK_KEY_rigdash 0abb
#define GDK_KEY_leftanglebracket 0xabc
#define GDK_KEY_decimalpoint 0abd
#define GDK_KEY_rightanglebracket 0xabe
#define GDK_KEY_marker 0abf
#define GDK_KEY_oneeighth 0xac3
#define GDK_KEY_threeeighths 0xac4
#define GDK_KEY_fiveeighths 0xac5
#define GDK_KEY_seveneighths 0xac6
#define GDK_KEY_trademark 0xac9
#define GDK_KEY_signaturemark 0aca
#define GDK_KEY_trademarkincircle 0xacb
#define GDK_KEY_leftopentriangle 0xacc
#define GDK_KEY_rightopentriangle 0xacd
#define GDK_KEY_emopencircle 0ace
#define GDK_KEY_emopenrectangle 0acf
#define GDK_KEY_leftsinglequotemark 0xad0
#define GDK_KEY_rightsinglequotemark 0xad1
#define GDK_KEY_leftdoublequotemark 0xad2
#define GDK_KEY_rightdoublequotemark 0xad3
#define GDK_KEY_prescription 0xad4
#define GDK_KEY_permille 0xad5
#define GDK_KEY_minutes 0xad6
#define GDK_KEY_seconds 0xad7

```

```

#define GDK_KEY_latincross      0xad9
#define GDK_KEY_hexagram       0xada
#define GDK_KEY_filledrectbullet 0xadb
#define GDK_KEY_filledlefttribullet 0xadc
#define GDK_KEY_filledrighttribullet 0xadd
#define GDK_KEY_emfilledcircle 0xade
#define GDK_KEY_emfilledrect 0xadf
#define GDK_KEY_enopencircbullet 0xae0
#define GDK_KEY_enopensquarebullet 0xae1
#define GDK_KEY_openrectbullet 0xae2
#define GDK_KEY_opentribulletup 0xae3
#define GDK_KEY_opentribulletdown 0xae4
#define GDK_KEY_openstar 0xae5
#define GDK_KEY_enfilledcircbullet 0xae6
#define GDK_KEY_enfilledsqbullet 0xae7
#define GDK_KEY_filledtribulletup 0xae8
#define GDK_KEY_filledtribulletdown 0xae9
#define GDK_KEY_leftpointer 0xaea
#define GDK_KEY_rightpointer 0xaeb
#define GDK_KEY_club 0xaec
#define GDK_KEY_diamond 0xaed
#define GDK_KEY_heart 0xaee
#define GDK_KEY_maltesecross 0xaf0
#define GDK_KEY_dagger 0xaf1
#define GDK_KEY_doubledagger 0xaf2
#define GDK_KEY_checkmark 0xaf3
#define GDK_KEY_ballotcross 0xaf4
#define GDK_KEY_musicalsharp 0xaf5
#define GDK_KEY_musicalflat 0xaf6
#define GDK_KEY_malesymbol 0xaf7
#define GDK_KEY_femalesymbol 0xaf8
#define GDK_KEY_telephone 0xaf9
#define GDK_KEY_telephonerecorder 0xafa
#define GDK_KEY_phonographcopyright 0xafb
#define GDK_KEY_caret 0xafc
#define GDK_KEY_singlelowquotemark 0xafd
#define GDK_KEY_doublelowquotemark 0xafe
#define GDK_KEY_cursor 0xaff
#define GDK_KEY_leftcaret 0xba3
#define GDK_KEY_rightcaret 0xba6
#define GDK_KEY_downcaret 0xba8
#define GDK_KEY_upcaret 0xba9
#define GDK_KEY_overbar 0xbc0
#define GDK_KEY_downtack 0xbc2
#define GDK_KEY_upshoe 0xbc3
#define GDK_KEY_downstile 0xbc4
#define GDK_KEY_underbar 0xbc6
#define GDK_KEY_jot 0xbca
#define GDK_KEY_quad 0xbcc
#define GDK_KEY_uptack 0xbce
#define GDK_KEY_circle 0xbcf
#define GDK_KEY_upstile 0xbd3
#define GDK_KEY_downshoe 0xbd6
#define GDK_KEY_rightshoe 0xbd8
#define GDK_KEY_leftshoe 0xbda
#define GDK_KEY_lefttack 0xbdc
#define GDK_KEY_righttack 0xbfc
#define GDK_KEY_hebrew_doublelowline 0xcdf
#define GDK_KEY_hebrew_aleph 0xce0
#define GDK_KEY_hebrew_bet 0xce1
#define GDK_KEY_hebrew_beth 0xce1
#define GDK_KEY_hebrew_gimel 0xce2
#define GDK_KEY_hebrew_gimmel 0xce2
#define GDK_KEY_hebrew_dalet 0xce3
#define GDK_KEY_hebrew_daleth 0xce3
#define GDK_KEY_hebrew_he 0xce4

```

```

#define GDK_KEY_hebrew_waw      0xce5
#define GDK_KEY_hebrew_zain     0xce6
#define GDK_KEY_hebrew_zayin    0xce6
#define GDK_KEY_hebrew_chet     0xce7
#define GDK_KEY_hebrew_het      0xce7
#define GDK_KEY_hebrew_tet      0xce8
#define GDK_KEY_hebrew_teth     0xce8
#define GDK_KEY_hebrew_yod      0xce9
#define GDK_KEY_hebrew_finalkaph 0xcea
#define GDK_KEY_hebrew_kaph     0xceb
#define GDK_KEY_hebrew_lamed    0xcec
#define GDK_KEY_hebrew_finalmem 0xcxed
#define GDK_KEY_hebrew_mem      0xcee
#define GDK_KEY_hebrew_finalnun 0xccef
#define GDK_KEY_hebrew_nun      0xcff0
#define GDK_KEY_hebrew_samech   0xcff1
#define GDK_KEY_hebrew_samekh   0xcff1
#define GDK_KEY_hebrew_ayin     0xcff2
#define GDK_KEY_hebrew_finalpe  0xcff3
#define GDK_KEY_hebrew_pe       0xcff4
#define GDK_KEY_hebrew_finalzade 0xcff5
#define GDK_KEY_hebrew_finalzadi 0xcff5
#define GDK_KEY_hebrew_zade     0xcff6
#define GDK_KEY_hebrew_zadi     0xcff6
#define GDK_KEY_hebrew_kuf      0xcff7
#define GDK_KEY_hebrew_qoph     0xcff7
#define GDK_KEY_hebrew_resh     0xcff8
#define GDK_KEY_hebrew_shin     0xcff9
#define GDK_KEY_hebrew_taf      0xcffa
#define GDK_KEY_hebrew_taw      0xcffa
#define GDK_KEY_Thai_kokai      0xda1
#define GDK_KEY_Thai_khokhai    0xda2
#define GDK_KEY_Thai_khokhuat   0xda3
#define GDK_KEY_Thai_khokhwai   0xda4
#define GDK_KEY_Thai_khokhon    0xda5
#define GDK_KEY_Thai_khorakhang 0xda6
#define GDK_KEY_Thai_ngongu     0xda7
#define GDK_KEY_Thai_chochan    0xda8
#define GDK_KEY_Thai_choching   0xda9
#define GDK_KEY_Thai_chochang   0xdaa
#define GDK_KEY_Thai_soso       0xdab
#define GDK_KEY_Thai_chochoe    0xdac
#define GDK_KEY_Thai_yoying     0xdad
#define GDK_KEY_Thai_dochada    0xdae
#define GDK_KEY_Thai_topatak    0xdaf
#define GDK_KEY_Thai_thothan    0xdb0
#define GDK_KEY_Thai_thonangmontho 0xdb1
#define GDK_KEY_Thai_thophuthao 0xdb2
#define GDK_KEY_Thai_nonen      0xdb3
#define GDK_KEY_Thai_dodek      0xdb4
#define GDK_KEY_Thai_totao      0xdb5
#define GDK_KEY_Thai_thothung   0xdb6
#define GDK_KEY_Thai_thothahan  0xdb7
#define GDK_KEY_Thai_thothong   0xdb8
#define GDK_KEY_Thai_nonu       0xdb9
#define GDK_KEY_Thai_bobaimai    0xdba
#define GDK_KEY_Thai_popla      0xdbb
#define GDK_KEY_Thai_phophung    0xdbc
#define GDK_KEY_Thai_fofa       0xdbd
#define GDK_KEY_Thai_phophan     0dbde
#define GDK_KEY_Thai_fofan      0dbdf
#define GDK_KEY_Thai_phosamphao 0xdc0
#define GDK_KEY_Thai_moma       0xdc1
#define GDK_KEY_Thai_yoyak      0xdc2
#define GDK_KEY_Thai_rorua      0xdc3
#define GDK_KEY_Thai_ru         0xdc4

```

```

#define GDK_KEY_Thai_loling      0xdc5
#define GDK_KEY_Thai_lu         0xdc6
#define GDK_KEY_Thai_wowaen     0xdc7
#define GDK_KEY_Thai_sosala     0xdc8
#define GDK_KEY_Thai_sorusi     0xdc9
#define GDK_KEY_Thai_sosua      0xdca
#define GDK_KEY_Thai_hohip      0xdcb
#define GDK_KEY_Thai_lochula    0xdcc
#define GDK_KEY_Thai_oang       0xdcd
#define GDK_KEY_Thai_honokhuk   0xdce
#define GDK_KEY_Thai_paiyannoi  0xdcf
#define GDK_KEY_Thai_saraa      0xdd0
#define GDK_KEY_Thai_maihanakat 0xdd1
#define GDK_KEY_Thai_saraaa     0xdd2
#define GDK_KEY_Thai_saraam     0xdd3
#define GDK_KEY_Thai_sarai      0xdd4
#define GDK_KEY_Thai_saraii     0xdd5
#define GDK_KEY_Thai_saraue     0xdd6
#define GDK_KEY_Thai_sarauee    0xdd7
#define GDK_KEY_Thai_sarau      0xdd8
#define GDK_KEY_Thai_sarauu     0xdd9
#define GDK_KEY_Thai_phinthu    0xdda
#define GDK_KEY_Thai_maihanakat_maitho 0xdde
#define GDK_KEY_Thai_baht       0xddf
#define GDK_KEY_Thai_sarae      0xde0
#define GDK_KEY_Thai_saraae     0xde1
#define GDK_KEY_Thai_sarao      0xde2
#define GDK_KEY_Thai_saraaimaimuan 0xde3
#define GDK_KEY_Thai_saraaimaimalai 0xde4
#define GDK_KEY_Thai_lakkhangyao 0xde5
#define GDK_KEY_Thai_maiyamok   0xde6
#define GDK_KEY_Thai_maitaikhui 0xde7
#define GDK_KEY_Thai_maiek      0xde8
#define GDK_KEY_Thai_maitho     0xde9
#define GDK_KEY_Thai_maitri     0xdea
#define GDK_KEY_Thai_maichattawa 0xdeb
#define GDK_KEY_Thai_thanthakhath 0xdec
#define GDK_KEY_Thai_nikhahit   0xded
#define GDK_KEY_Thai_leksun     0xdf0
#define GDK_KEY_Thai_leknung    0xdf1
#define GDK_KEY_Thai_leksong    0xdf2
#define GDK_KEY_Thai_leksam     0xdf3
#define GDK_KEY_Thai_leksi      0xdf4
#define GDK_KEY_Thai_lekha      0xdf5
#define GDK_KEY_Thai_lekhok     0xdf6
#define GDK_KEY_Thai_lekchet    0xdf7
#define GDK_KEY_Thai_lekpaet    0xdf8
#define GDK_KEY_Thai_lekkao     0xdf9
#define GDK_KEY_Hangul_Kiyeog    0xea1
#define GDK_KEY_Hangul_SsangKiyeog 0xea2
#define GDK_KEY_Hangul_KiyeogSios 0xea3
#define GDK_KEY_Hangul_Nieun     0xea4
#define GDK_KEY_Hangul_NieunJieuj 0xea5
#define GDK_KEY_Hangul_NieunHieuh 0xea6
#define GDK_KEY_Hangul_Dikeud    0xea7
#define GDK_KEY_Hangul_SsangDikeud 0xea8
#define GDK_KEY_Hangul_Rieul     0xea9
#define GDK_KEY_Hangul_RieulKiyeog 0xeaa
#define GDK_KEY_Hangul_RieulMieum 0xeab
#define GDK_KEY_Hangul_RieulPieub 0xeac
#define GDK_KEY_Hangul_RieulSios 0xead
#define GDK_KEY_Hangul_RieulTieut 0xaeae
#define GDK_KEY_Hangul_RieulPhieuf 0xaeaf
#define GDK_KEY_Hangul_RieulHieuh 0xeb0
#define GDK_KEY_Hangul_Mieum     0xeb1
#define GDK_KEY_Hangul_Pieub     0xeb2

```

```

#define GDK_KEY_Hangul_SsangPieub 0xeb3
#define GDK_KEY_Hangul_PieubSios 0xeb4
#define GDK_KEY_Hangul_Sios 0xeb5
#define GDK_KEY_Hangul_SsangSios 0xeb6
#define GDK_KEY_Hangul_Ieung 0xeb7
#define GDK_KEY_Hangul_Jieuj 0xeb8
#define GDK_KEY_Hangul_SsangJieuj 0xeb9
#define GDK_KEY_Hangul_Cieuc 0xeba
#define GDK_KEY_Hangul_Khieug 0xebb
#define GDK_KEY_Hangul_Tieut 0xebc
#define GDK_KEY_Hangul_Phieuf 0xebd
#define GDK_KEY_Hangul_Hieuh 0ebe
#define GDK_KEY_Hangul_A 0xebf
#define GDK_KEY_Hangul_AE 0xec0
#define GDK_KEY_Hangul_YA 0xec1
#define GDK_KEY_Hangul_YAE 0xec2
#define GDK_KEY_Hangul_EO 0xec3
#define GDK_KEY_Hangul_E 0xec4
#define GDK_KEY_Hangul_YEO 0xec5
#define GDK_KEY_Hangul_YE 0xec6
#define GDK_KEY_Hangul_O 0xec7
#define GDK_KEY_Hangul_WA 0xec8
#define GDK_KEY_Hangul_WAE 0xec9
#define GDK_KEY_Hangul_OE 0xeca
#define GDK_KEY_Hangul_YO 0xecb
#define GDK_KEY_Hangul_U 0xecc
#define GDK_KEY_Hangul_WEO 0xecd
#define GDK_KEY_Hangul_WE 0xece
#define GDK_KEY_Hangul_WI 0xecf
#define GDK_KEY_Hangul_YU 0xed0
#define GDK_KEY_Hangul_EU 0xed1
#define GDK_KEY_Hangul_YI 0xed2
#define GDK_KEY_Hangul_I 0xed3
#define GDK_KEY_Hangul_J_Kiyeog 0xed4
#define GDK_KEY_Hangul_J_SsangKiyeog 0xed5
#define GDK_KEY_Hangul_J_KiyeogSios 0xed6
#define GDK_KEY_Hangul_J_Nieun 0xed7
#define GDK_KEY_Hangul_J_NieunJieuj 0xed8
#define GDK_KEY_Hangul_J_NieunHieuh 0xed9
#define GDK_KEY_Hangul_J_Dikeud 0xeda
#define GDK_KEY_Hangul_J_Rieul 0xedb
#define GDK_KEY_Hangul_J_RieulKiyeog 0xedc
#define GDK_KEY_Hangul_J_RieulMieum 0xedd
#define GDK_KEY_Hangul_J_RieulPieub 0xede
#define GDK_KEY_Hangul_J_RieulSios 0xedf
#define GDK_KEY_Hangul_J_RieulTieut 0xee0
#define GDK_KEY_Hangul_J_RieulPhieuf 0xee1
#define GDK_KEY_Hangul_J_RieulHieuh 0xee2
#define GDK_KEY_Hangul_J_Mieum 0xee3
#define GDK_KEY_Hangul_J_Pieub 0xee4
#define GDK_KEY_Hangul_J_PieubSios 0xee5
#define GDK_KEY_Hangul_J_Sios 0xee6
#define GDK_KEY_Hangul_J_SsangSios 0xee7
#define GDK_KEY_Hangul_J_Ieung 0xee8
#define GDK_KEY_Hangul_J_Jieuj 0xee9
#define GDK_KEY_Hangul_J_Cieuc 0xeea
#define GDK_KEY_Hangul_J_Khieug 0xeeb
#define GDK_KEY_Hangul_J_Tieut 0xeec
#define GDK_KEY_Hangul_J_Phieuf 0xeed
#define GDK_KEY_Hangul_J_Hieuh 0xeee
#define GDK_KEY_Hangul_RieulYeorinHieuh 0xeef
#define GDK_KEY_Hangul_SunkyeongeumMieum 0xef0
#define GDK_KEY_Hangul_SunkyeongeumPieub 0xef1
#define GDK_KEY_Hangul_PanSios 0xef2
#define GDK_KEY_Hangul_KkogjiDalrinIeung 0xef3
#define GDK_KEY_Hangul_SunkyeongeumPhieuf 0xef4

```

```

#define GDK_KEY_Hangul_YeorinHieuh 0xef5
#define GDK_KEY_Hangul_AraeA 0xef6
#define GDK_KEY_Hangul_AraeAE 0xef7
#define GDK_KEY_Hangul_J_PanSios 0xef8
#define GDK_KEY_Hangul_J_KkogjiDalrinJeung 0xef9
#define GDK_KEY_Hangul_J_YeorinHieuh 0xefa
#define GDK_KEY_Korean_Won 0xeff
#define GDK_KEY_3270_Duplicate 0xfd01
#define GDK_KEY_3270_FieldMark 0xfd02
#define GDK_KEY_3270_Right2 0xfd03
#define GDK_KEY_3270_Left2 0xfd04
#define GDK_KEY_3270_BackTab 0xfd05
#define GDK_KEY_3270_EraseEOF 0xfd06
#define GDK_KEY_3270_EraseInput 0xfd07
#define GDK_KEY_3270_Reset 0xfd08
#define GDK_KEY_3270_Quit 0xfd09
#define GDK_KEY_3270_PA1 0xfd0a
#define GDK_KEY_3270_PA2 0xfd0b
#define GDK_KEY_3270_PA3 0xfd0c
#define GDK_KEY_3270_Test 0xfd0d
#define GDK_KEY_3270_Attn 0xfd0e
#define GDK_KEY_3270_CursorBlink 0xfd0f
#define GDK_KEY_3270_AltCursor 0xfd10
#define GDK_KEY_3270_KeyClick 0xfd11
#define GDK_KEY_3270_Jump 0xfd12
#define GDK_KEY_3270_Ident 0xfd13
#define GDK_KEY_3270_Rule 0xfd14
#define GDK_KEY_3270_Copy 0xfd15
#define GDK_KEY_3270_Play 0xfd16
#define GDK_KEY_3270_Setup 0xfd17
#define GDK_KEY_3270_Record 0xfd18
#define GDK_KEY_3270_ChangeScreen 0xfd19
#define GDK_KEY_3270_DeleteWord 0xfd1a
#define GDK_KEY_3270_ExSelect 0xfd1b
#define GDK_KEY_3270_CursorSelect 0xfd1c
#define GDK_KEY_3270_PrintScreen 0xfd1d
#define GDK_KEY_3270_Enter 0xfd1e
#define GDK_KEY_ISO_Lock 0xfe01
#define GDK_KEY_ISO_Level2_Latch 0xfe02
#define GDK_KEY_ISO_Level3_Shift 0xfe03
#define GDK_KEY_ISO_Level3_Latch 0xfe04
#define GDK_KEY_ISO_Level3_Lock 0xfe05
#define GDK_KEY_ISO_Group_Latch 0xfe06
#define GDK_KEY_ISO_Group_Lock 0xfe07
#define GDK_KEY_ISO_Next_Group 0xfe08
#define GDK_KEY_ISO_Next_Group_Lock 0xfe09
#define GDK_KEY_ISO_Prev_Group 0xfe0a
#define GDK_KEY_ISO_Prev_Group_Lock 0xfe0b
#define GDK_KEY_ISO_First_Group 0xfe0c
#define GDK_KEY_ISO_First_Group_Lock 0xfe0d
#define GDK_KEY_ISO_Last_Group 0xfe0e
#define GDK_KEY_ISO_Last_Group_Lock 0xfe0f
#define GDK_KEY_ISO_Level5_Shift 0xfe11
#define GDK_KEY_ISO_Level5_Latch 0xfe12
#define GDK_KEY_ISO_Level5_Lock 0xfe13
#define GDK_KEY_ISO_Left_Tab 0xfe20
#define GDK_KEY_ISO_Move_Line_Up 0xfe21
#define GDK_KEY_ISO_Move_Line_Down 0xfe22
#define GDK_KEY_ISO_Partial_Line_Up 0xfe23
#define GDK_KEY_ISO_Partial_Line_Down 0xfe24
#define GDK_KEY_ISO_Partial_Space_Left 0xfe25
#define GDK_KEY_ISO_Partial_Space_Right 0xfe26
#define GDK_KEY_ISO_Set_Margin_Left 0xfe27
#define GDK_KEY_ISO_Set_Margin_Right 0xfe28
#define GDK_KEY_ISO_Release_Margin_Left 0xfe29
#define GDK_KEY_ISO_Release_Margin_Right 0xfe2a

```



```

#define GDK_KEY_ISO_Release_Both_Margins 0xfe2b
#define GDK_KEY_ISO_Fast_Cursor_Left 0xfe2c
#define GDK_KEY_ISO_Fast_Cursor_Right 0xfe2d
#define GDK_KEY_ISO_Fast_Cursor_Up 0xfe2e
#define GDK_KEY_ISO_Fast_Cursor_Down 0xfe2f
#define GDK_KEY_ISO_Continuous_Underline 0xfe30
#define GDK_KEY_ISO_Discontinuous_Underline 0xfe31
#define GDK_KEY_ISO_Emphasize 0xfe32
#define GDK_KEY_ISO_Center_Object 0xfe33
#define GDK_KEY_ISO_Enter 0xfe34
#define GDK_KEY_dead_grave 0xfe50
#define GDK_KEY_dead_acute 0xfe51
#define GDK_KEY_dead_circumflex 0xfe52
#define GDK_KEY_dead_perispomeni 0xfe53
#define GDK_KEY_dead_tilde 0xfe53
#define GDK_KEY_dead_macron 0xfe54
#define GDK_KEY_dead_breve 0xfe55
#define GDK_KEY_dead_abovedot 0xfe56
#define GDK_KEY_dead_diaeresis 0xfe57
#define GDK_KEY_dead_aoovering 0xfe58
#define GDK_KEY_dead_doubleacute 0xfe59
#define GDK_KEY_dead_caron 0xfe5a
#define GDK_KEY_dead_cedilla 0xfe5b
#define GDK_KEY_dead_ogonek 0xfe5c
#define GDK_KEY_dead_iota 0xfe5d
#define GDK_KEY_dead_voiced_sound 0xfe5e
#define GDK_KEY_dead_semivoiced_sound 0xfe5f
#define GDK_KEY_dead_belowdot 0xfe60
#define GDK_KEY_dead_hook 0xfe61
#define GDK_KEY_dead_horn 0xfe62
#define GDK_KEY_dead_stroke 0xfe63
#define GDK_KEY_dead_abovecomma 0xfe64
#define GDK_KEY_dead_psili 0xfe64
#define GDK_KEY_dead_aooverereversedcomma 0xfe65
#define GDK_KEY_dead_dasia 0xfe65
#define GDK_KEY_dead_doublegrave 0xfe66
#define GDK_KEY_dead_belowring 0xfe67
#define GDK_KEY_dead_belowmacron 0xfe68
#define GDK_KEY_dead_belowcircumflex 0xfe69
#define GDK_KEY_dead_belowtilde 0xfe6a
#define GDK_KEY_dead_belowbreve 0xfe6b
#define GDK_KEY_dead_belowdiaeresis 0xfe6c
#define GDK_KEY_dead_invertedbreve 0xfe6d
#define GDK_KEY_dead_belowcomma 0xfe6e
#define GDK_KEY_dead_currency 0xfe6f
#define GDK_KEY_AccessX_Enable 0xfe70
#define GDK_KEY_AccessX_Feedback_Enable 0xfe71
#define GDK_KEY_RepeatKeys_Enable 0xfe72
#define GDK_KEY_SlowKeys_Enable 0xfe73
#define GDK_KEY_BounceKeys_Enable 0xfe74
#define GDK_KEY_StickyKeys_Enable 0xfe75
#define GDK_KEY_MouseKeys_Enable 0xfe76
#define GDK_KEY_MouseKeys_Accel_Enable 0xfe77
#define GDK_KEY_Overlay1_Enable 0xfe78
#define GDK_KEY_Overlay2_Enable 0xfe79
#define GDK_KEY_AudibleBell_Enable 0xfe7a
#define GDK_KEY_dead_a 0xfe80
#define GDK_KEY_dead_A 0xfe81
#define GDK_KEY_dead_e 0xfe82
#define GDK_KEY_dead_E 0xfe83
#define GDK_KEY_dead_i 0xfe84
#define GDK_KEY_dead_I 0xfe85
#define GDK_KEY_dead_o 0xfe86
#define GDK_KEY_dead_O 0xfe87
#define GDK_KEY_dead_u 0xfe88
#define GDK_KEY_dead_U 0xfe89

```



```

#define GDK_KEY_dead_small_schwa      0xfe8a
#define GDK_KEY_dead_capital_schwa    0xfe8b
#define GDK_KEY_dead_greek            0xfe8c
#define GDK_KEY_ch                     0xfea0
#define GDK_KEY_Ch                     0xfea1
#define GDK_KEY_CH                     0xfea2
#define GDK_KEY_c_h                    0xfea3
#define GDK_KEY_C_h                    0xfea4
#define GDK_KEY_C_H                    0xfea5
#define GDK_KEY_First_Virtual_Screen   0xfed0
#define GDK_KEY_Prev_Virtual_Screen    0xfed1
#define GDK_KEY_Next_Virtual_Screen    0xfed2
#define GDK_KEY_Last_Virtual_Screen    0xfed4
#define GDK_KEY_Terminate_Server       0xfed5
#define GDK_KEY_Pointer_Left           0fee0
#define GDK_KEY_Pointer_Right          0fee1
#define GDK_KEY_Pointer_Up             0fee2
#define GDK_KEY_Pointer_Down           0fee3
#define GDK_KEY_Pointer_UpLeft         0fee4
#define GDK_KEY_Pointer_UpRight        0fee5
#define GDK_KEY_Pointer_DownLeft       0fee6
#define GDK_KEY_Pointer_DownRight      0fee7
#define GDK_KEY_Pointer_Button_Dflt    0fee8
#define GDK_KEY_Pointer_Button1        0fee9
#define GDK_KEY_Pointer_Button2        0feea
#define GDK_KEY_Pointer_Button3        0feeb
#define GDK_KEY_Pointer_Button4        0feec
#define GDK_KEY_Pointer_Button5        0feed
#define GDK_KEY_Pointer_DblClick_Dflt  0feee
#define GDK_KEY_Pointer_DblClick1      0feef
#define GDK_KEY_Pointer_DblClick2      0fef0
#define GDK_KEY_Pointer_DblClick3      0fef1
#define GDK_KEY_Pointer_DblClick4      0fef2
#define GDK_KEY_Pointer_DblClick5      0fef3
#define GDK_KEY_Pointer_Drag_Dflt      0fef4
#define GDK_KEY_Pointer_Drag1          0fef5
#define GDK_KEY_Pointer_Drag2          0fef6
#define GDK_KEY_Pointer_Drag3          0fef7
#define GDK_KEY_Pointer_Drag4          0fef8
#define GDK_KEY_Pointer_EnableKeys     0fef9
#define GDK_KEY_Pointer_Accelerate     0fefa
#define GDK_KEY_Pointer_DfltBtnNext    0fefb
#define GDK_KEY_Pointer_DfltBtnPrev    0fefc
#define GDK_KEY_Pointer_Drag5          0xfefd
#define GDK_KEY_BackSpace              0xff08
#define GDK_KEY_Tab                    0xff09
#define GDK_KEY_Linefeed               0xff0a
#define GDK_KEY_Clear                   0xff0b
#define GDK_KEY_Return                 0xff0d
#define GDK_KEY_Pause                  0xff13
#define GDK_KEY_Scroll_Lock            0xff14
#define GDK_KEY_Sys_Req                0xff15
#define GDK_KEY_Escape                 0xff1b
#define GDK_KEY_Multi_key              0xff20
#define GDK_KEY_Kanji                  0xff21
#define GDK_KEY_Muhenkan                0xff22
#define GDK_KEY_Henkan                 0xff23
#define GDK_KEY_Henkan_Mode            0xff23
#define GDK_KEY_Romaji                 0xff24
#define GDK_KEY_Hiragana               0xff25
#define GDK_KEY_Katakana               0xff26
#define GDK_KEY_Hiragana_Katakana      0xff27
#define GDK_KEY_Zenkaku                0xff28
#define GDK_KEY_Hankaku                0xff29
#define GDK_KEY_Zenkaku_Hankaku        0xff2a
#define GDK_KEY_Touroku                0xff2b

```

```

#define GDK_KEY_Massyo 0xff2c
#define GDK_KEY_Kana_Lock 0xff2d
#define GDK_KEY_Kana_Shift 0xff2e
#define GDK_KEY_Eisu_Shift 0xff2f
#define GDK_KEY_Eisu_toggle 0xff30
#define GDK_KEY_Hangul 0xff31
#define GDK_KEY_Hangul_Start 0xff32
#define GDK_KEY_Hangul_End 0xff33
#define GDK_KEY_Hangul_Hanja 0xff34
#define GDK_KEY_Hangul_Jamo 0xff35
#define GDK_KEY_Hangul_Romaja 0xff36
#define GDK_KEY_Codeinput 0xff37
#define GDK_KEY_Hangul_Codeinput 0xff37
#define GDK_KEY_Kanji_Bangou 0xff37
#define GDK_KEY_Hangul_Jeonja 0xff38
#define GDK_KEY_Hangul_Banja 0xff39
#define GDK_KEY_Hangul_PreHanja 0xff3a
#define GDK_KEY_Hangul_PostHanja 0xff3b
#define GDK_KEY_Hangul_SingleCandidate 0xff3c
#define GDK_KEY_SingleCandidate 0xff3c
#define GDK_KEY_Hangul_MultipleCandidate 0xff3d
#define GDK_KEY_MultipleCandidate 0xff3d
#define GDK_KEY_Zen_Koho 0xff3d
#define GDK_KEY_Hangul_PreviousCandidate 0xff3e
#define GDK_KEY_Mae_Koho 0xff3e
#define GDK_KEY_PreviousCandidate 0xff3e
#define GDK_KEY_Hangul_Special 0xff3f
#define GDK_KEY_Home 0xff50
#define GDK_KEY_Left 0xff51
#define GDK_KEY_Up 0xff52
#define GDK_KEY_Right 0xff53
#define GDK_KEY_Down 0xff54
#define GDK_KEY_Page_Up 0xff55
#define GDK_KEY_Prior 0xff55
#define GDK_KEY_Next 0xff56
#define GDK_KEY_Page_Down 0xff56
#define GDK_KEY_End 0xff57
#define GDK_KEY_Begin 0xff58
#define GDK_KEY_Select 0xff60
#define GDK_KEY_Print 0xff61
#define GDK_KEY_Execute 0xff62
#define GDK_KEY_Insert 0xff63
#define GDK_KEY_Undo 0xff65
#define GDK_KEY_Redo 0xff66
#define GDK_KEY_Menu 0xff67
#define GDK_KEY_Find 0xff68
#define GDK_KEY_Cancel 0xff69
#define GDK_KEY_Help 0xff6a
#define GDK_KEY_Break 0xff6b
#define GDK_KEY_Arabic_switch 0xff7e
#define GDK_KEY_Greek_switch 0xff7e
#define GDK_KEY_Hangul_switch 0xff7e
#define GDK_KEY_Hebrew_switch 0xff7e
#define GDK_KEY_ISO_Group_Shift 0xff7e
#define GDK_KEY_Mode_switch 0xff7e
#define GDK_KEY_kana_switch 0xff7e
#define GDK_KEY_script_switch 0xff7e
#define GDK_KEY_Num_Lock 0xff7f
#define GDK_KEY_KP_Space 0xff80
#define GDK_KEY_KP_Tab 0xff89
#define GDK_KEY_KP_Enter 0xff8d
#define GDK_KEY_KP_F1 0xff91
#define GDK_KEY_KP_F2 0xff92
#define GDK_KEY_KP_F3 0xff93
#define GDK_KEY_KP_F4 0xff94
#define GDK_KEY_KP_Home 0xff95

```

```

#define GDK_KEY_KP_Left 0xff96
#define GDK_KEY_KP_Up 0xff97
#define GDK_KEY_KP_Right 0xff98
#define GDK_KEY_KP_Down 0xff99
#define GDK_KEY_KP_Page_Up 0xff9a
#define GDK_KEY_KP_Prior 0xff9a
#define GDK_KEY_KP_Next 0xff9b
#define GDK_KEY_KP_Page_Down 0xff9b
#define GDK_KEY_KP_End 0xff9c
#define GDK_KEY_KP_Begin 0xff9d
#define GDK_KEY_KP_Insert 0xff9e
#define GDK_KEY_KP_Delete 0xff9f
#define GDK_KEY_KP_Multiply 0xffaa
#define GDK_KEY_KP_Add 0xffab
#define GDK_KEY_KP_Separator 0xffac
#define GDK_KEY_KP_Subtract 0xffad
#define GDK_KEY_KP_Decimal 0xffae
#define GDK_KEY_KP_Divide 0xffaf
#define GDK_KEY_KP_0 0xffb0
#define GDK_KEY_KP_1 0xffb1
#define GDK_KEY_KP_2 0xffb2
#define GDK_KEY_KP_3 0xffb3
#define GDK_KEY_KP_4 0xffb4
#define GDK_KEY_KP_5 0xffb5
#define GDK_KEY_KP_6 0xffb6
#define GDK_KEY_KP_7 0xffb7
#define GDK_KEY_KP_8 0xffb8
#define GDK_KEY_KP_9 0xffb9
#define GDK_KEY_KP_Equal 0xffbd
#define GDK_KEY_F1 0xffbe
#define GDK_KEY_F2 0xffbf
#define GDK_KEY_F3 0xffc0
#define GDK_KEY_F4 0xffc1
#define GDK_KEY_F5 0xffc2
#define GDK_KEY_F6 0xffc3
#define GDK_KEY_F7 0xffc4
#define GDK_KEY_F8 0xffc5
#define GDK_KEY_F9 0xffc6
#define GDK_KEY_F10 0xffc7
#define GDK_KEY_F11 0xffc8
#define GDK_KEY_L1 0xffc8
#define GDK_KEY_F12 0xffc9
#define GDK_KEY_L2 0xffc9
#define GDK_KEY_F13 0xffca
#define GDK_KEY_L3 0xffca
#define GDK_KEY_F14 0xffcb
#define GDK_KEY_L4 0xffcb
#define GDK_KEY_F15 0xffcc
#define GDK_KEY_L5 0xffcc
#define GDK_KEY_F16 0xffcd
#define GDK_KEY_L6 0xffcd
#define GDK_KEY_F17 0xffce
#define GDK_KEY_L7 0xffce
#define GDK_KEY_F18 0xffcf
#define GDK_KEY_L8 0xffcf
#define GDK_KEY_F19 0xffd0
#define GDK_KEY_L9 0xffd0
#define GDK_KEY_F20 0xffd1
#define GDK_KEY_L10 0xffd1
#define GDK_KEY_F21 0xffd2
#define GDK_KEY_R1 0xffd2
#define GDK_KEY_F22 0xffd3
#define GDK_KEY_R2 0xffd3
#define GDK_KEY_F23 0xffd4
#define GDK_KEY_R3 0xffd4
#define GDK_KEY_F24 0xffd5

```

```

#define GDK_KEY_R4      0xffd5
#define GDK_KEY_F25     0xffd6
#define GDK_KEY_R5      0xffd6
#define GDK_KEY_F26     0xffd7
#define GDK_KEY_R6      0xffd7
#define GDK_KEY_F27     0xffd8
#define GDK_KEY_R7      0xffd8
#define GDK_KEY_F28     0xffd9
#define GDK_KEY_R8      0xffd9
#define GDK_KEY_F29     0xffda
#define GDK_KEY_R9      0xffda
#define GDK_KEY_F30     0xffdb
#define GDK_KEY_R10     0xffdb
#define GDK_KEY_F31     0xffdc
#define GDK_KEY_R11     0xffdc
#define GDK_KEY_F32     0xffdd
#define GDK_KEY_R12     0xffdd
#define GDK_KEY_F33     0xffde
#define GDK_KEY_R13     0xffde
#define GDK_KEY_F34     0xffdf
#define GDK_KEY_R14     0xffdf
#define GDK_KEY_F35     0xffe0
#define GDK_KEY_R15     0xffe0
#define GDK_KEY_Shift_L 0xffe1
#define GDK_KEY_Shift_R 0xffe2
#define GDK_KEY_Control_L 0xffe3
#define GDK_KEY_Control_R 0xffe4
#define GDK_KEY_Caps_Lock 0xffe5
#define GDK_KEY_Shift_Lock 0xffe6
#define GDK_KEY_Meta_L 0xffe7
#define GDK_KEY_Meta_R 0xffe8
#define GDK_KEY_Alt_L 0xffe9
#define GDK_KEY_Alt_R 0xffea
#define GDK_KEY_Super_L 0xffeb
#define GDK_KEY_Super_R 0xffec
#define GDK_KEY_Hyper_L 0xffed
#define GDK_KEY_Hyper_R 0xffee
#define GDK_KEY_braille_dot_1 0xffff1
#define GDK_KEY_braille_dot_2 0xffff2
#define GDK_KEY_braille_dot_3 0xffff3
#define GDK_KEY_braille_dot_4 0xffff4
#define GDK_KEY_braille_dot_5 0xffff5
#define GDK_KEY_braille_dot_6 0xffff6
#define GDK_KEY_braille_dot_7 0xffff7
#define GDK_KEY_braille_dot_8 0xffff8
#define GDK_KEY_braille_dot_9 0xffff9
#define GDK_KEY_braille_dot_10 0xffffa
#define GDK_KEY_Delete 0xfffff
#define GDK_KEY_VoidSymbol 0xfffffff

```

7.3.5 gtk-3.0/gdk/gdkx.h

```

#define __GDKX_H_INSIDE__
#define __GDK_X11_APP_LAUNCH_CONTEXT_H__
#define __GDK_X11_CURSOR_H__
#define __GDK_X11_DEVICE_CORE_H__
#define __GDK_X11_DEVICE_H__
#define __GDK_X11_DEVICE_MANAGER_CORE_H__
#define __GDK_X11_DEVICE_MANAGER_H__
#define __GDK_X11_DEVICE_MANAGER_XI2_H__
#define __GDK_X11_DEVICE_XI2_H__
#define __GDK_X11_DISPLAY_H__
#define __GDK_X11_DISPLAY_MANAGER_H__
#define __GDK_X11_DND_H__
#define __GDK_X11_KEYS_H__

```

```

#define __GDK_X11_PROPERTY_H__
#define __GDK_X11_SCREEN_H__
#define __GDK_X11_SELECTION_H__
#define __GDK_X11_UTILS_H__
#define __GDK_X11_VISUAL_H__
#define __GDK_X11_WINDOW_H__
#define __GDK_X_H__
#define GDK_TYPE_X11_APP_LAUNCH_CONTEXT
(gdk_x11_app_launch_context_get_type ())
#define GDK_TYPE_X11_CURSOR (gdk_x11_cursor_get_type ())
#define GDK_TYPE_X11_DEVICE_CORE
(gdk_x11_device_core_get_type ())
#define GDK_TYPE_X11_DEVICE_MANAGER_CORE
(gdk_x11_device_manager_core_get_type ())
#define GDK_TYPE_X11_DEVICE_MANAGER_XI2
(gdk_x11_device_manager_xi2_get_type ())
#define GDK_TYPE_X11_DEVICE_XI2 (gdk_x11_device_xi2_get_type ())
#define GDK_TYPE_X11_DISPLAY (gdk_x11_display_get_type ())
#define GDK_TYPE_X11_DISPLAY_MANAGER
(gdk_x11_display_manager_get_type ())
#define GDK_TYPE_X11_DRAG_CONTEXT
(gdk_x11_drag_context_get_type ())
#define GDK_TYPE_X11_KEYMAP (gdk_x11_keymap_get_type ())
#define GDK_TYPE_X11_SCREEN (gdk_x11_screen_get_type ())
#define GDK_TYPE_X11_VISUAL (gdk_x11_visual_get_type ())
#define GDK_TYPE_X11_WINDOW (gdk_x11_window_get_type ())
#define GDK_X11_DEVICE_CORE_CLASS(c) (G_TYPE_CHECK_CLASS_CAST
((c), GDK_TYPE_X11_DEVICE_CORE, GdkX11DeviceCoreClass))
#define GDK_X11_DEVICE_MANAGER_CORE_CLASS(c)
(G_TYPE_CHECK_CLASS_CAST ((c), GDK_TYPE_X11_DEVICE_MANAGER_CORE,
GdkX11DeviceManagerCoreClass))
#define GDK_X11_DEVICE_MANAGER_XI2_CLASS(c)
(G_TYPE_CHECK_CLASS_CAST ((c), GDK_TYPE_X11_DEVICE_MANAGER_XI2,
GdkX11DeviceManagerXI2Class))
#define GDK_X11_DEVICE_XI2_CLASS(c) (G_TYPE_CHECK_CLASS_CAST
((c), GDK_TYPE_X11_DEVICE_XI2, GdkX11DeviceXI2Class))
#define GDK_X11_APP_LAUNCH_CONTEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_X11_APP_LAUNCH_CONTEXT,
GdkX11AppLaunchContextClass))
#define GDK_X11_CURSOR_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_CURSOR, GdkX11CursorClass))
#define GDK_X11_DISPLAY_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_DISPLAY, GdkX11DisplayClass))
#define GDK_X11_DISPLAY_MANAGER_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_X11_DISPLAY_MANAGER,
GdkX11DisplayManagerClass))
#define GDK_X11_DRAG_CONTEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GDK_TYPE_X11_DRAG_CONTEXT,
GdkX11DragContextClass))
#define GDK_X11_KEYMAP_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_KEYMAP, GdkX11KeymapClass))
#define GDK_X11_SCREEN_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_SCREEN, GdkX11ScreenClass))
#define GDK_X11_VISUAL_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_VISUAL, GdkX11VisualClass))
#define GDK_X11_WINDOW_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GDK_TYPE_X11_WINDOW, GdkX11WindowClass))
#define GDK_IS_X11_DEVICE_CORE_CLASS(c) (G_TYPE_CHECK_CLASS_TYPE
((c), GDK_TYPE_X11_DEVICE_CORE))
#define GDK_IS_X11_DEVICE_MANAGER_CORE_CLASS(c)
(G_TYPE_CHECK_CLASS_TYPE ((c), GDK_TYPE_X11_DEVICE_MANAGER_CORE))
#define GDK_IS_X11_DEVICE_MANAGER_XI2_CLASS(c)
(G_TYPE_CHECK_CLASS_TYPE ((c), GDK_TYPE_X11_DEVICE_MANAGER_XI2))
#define GDK_IS_X11_DEVICE_XI2_CLASS(c) (G_TYPE_CHECK_CLASS_TYPE
((c), GDK_TYPE_X11_DEVICE_XI2))

```

```

#define GDK_IS_X11_APP_LAUNCH_CONTEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass),
GDK_TYPE_X11_APP_LAUNCH_CONTEXT))
#define GDK_IS_X11_CURSOR_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_CURSOR))
#define GDK_IS_X11_DISPLAY_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_DISPLAY))
#define GDK_IS_X11_DISPLAY_MANAGER_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GDK_TYPE_X11_DISPLAY_MANAGER))
#define GDK_IS_X11_DRAG_CONTEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GDK_TYPE_X11_DRAG_CONTEXT))
#define GDK_IS_X11_KEYMAP_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_KEYMAP))
#define GDK_IS_X11_SCREEN_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_SCREEN))
#define GDK_IS_X11_VISUAL_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_VISUAL))
#define GDK_IS_X11_WINDOW_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GDK_TYPE_X11_WINDOW))
#define GDK_X11_DEVICE_CORE(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GDK_TYPE_X11_DEVICE_CORE, GdkX11DeviceCore))
#define GDK_X11_DEVICE_MANAGER_CORE(o)
(G_TYPE_CHECK_INSTANCE_CAST ((o), GDK_TYPE_X11_DEVICE_MANAGER_CORE,
GdkX11DeviceManagerCore))
#define GDK_X11_DEVICE_MANAGER_XI2(o)
(G_TYPE_CHECK_INSTANCE_CAST ((o), GDK_TYPE_X11_DEVICE_MANAGER_XI2,
GdkX11DeviceManagerXI2))
#define GDK_X11_DEVICE_XI2(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GDK_TYPE_X11_DEVICE_XI2, GdkX11DeviceXI2))
#define GDK_X11_APP_LAUNCH_CONTEXT(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object),
GDK_TYPE_X11_APP_LAUNCH_CONTEXT, GdkX11AppLaunchContext))
#define GDK_X11_CURSOR(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_CURSOR, GdkX11Cursor))
#define GDK_X11_DISPLAY(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_DISPLAY, GdkX11Display))
#define GDK_X11_DISPLAY_MANAGER(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object),
GDK_TYPE_X11_DISPLAY_MANAGER, GdkX11DisplayManager))
#define GDK_X11_DRAG_CONTEXT(object)
(G_TYPE_CHECK_INSTANCE_CAST ((object), GDK_TYPE_X11_DRAG_CONTEXT,
GdkX11DragContext))
#define GDK_X11_KEYMAP(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_KEYMAP, GdkX11Keymap))
#define GDK_X11_SCREEN(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_SCREEN, GdkX11Screen))
#define GDK_X11_VISUAL(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_VISUAL, GdkX11Visual))
#define GDK_X11_WINDOW(object) (G_TYPE_CHECK_INSTANCE_CAST
((object), GDK_TYPE_X11_WINDOW, GdkX11Window))
#define GDK_IS_X11_DEVICE_CORE(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GDK_TYPE_X11_DEVICE_CORE))
#define GDK_IS_X11_DEVICE_MANAGER_CORE(o)
(G_TYPE_CHECK_INSTANCE_TYPE ((o),
GDK_TYPE_X11_DEVICE_MANAGER_CORE))
#define GDK_IS_X11_DEVICE_MANAGER_XI2(o)
(G_TYPE_CHECK_INSTANCE_TYPE ((o), GDK_TYPE_X11_DEVICE_MANAGER_XI2))
#define GDK_IS_X11_DEVICE_XI2(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GDK_TYPE_X11_DEVICE_XI2))
#define GDK_IS_X11_APP_LAUNCH_CONTEXT(object)
(G_TYPE_CHECK_INSTANCE_TYPE ((object),
GDK_TYPE_X11_APP_LAUNCH_CONTEXT))
#define GDK_IS_X11_CURSOR(object) (G_TYPE_CHECK_INSTANCE_TYPE
((object), GDK_TYPE_X11_CURSOR))
#define GDK_IS_X11_DISPLAY(object) (G_TYPE_CHECK_INSTANCE_TYPE
((object), GDK_TYPE_X11_DISPLAY))

```



```

#define GDK_IS_X11_DISPLAY_MANAGER(object) \
(G_TYPE_CHECK_INSTANCE_TYPE ((object), \
GDK_TYPE_X11_DISPLAY_MANAGER))
#define GDK_IS_X11_DRAG_CONTEXT(object) \
(G_TYPE_CHECK_INSTANCE_TYPE ((object), GDK_TYPE_X11_DRAG_CONTEXT))
#define GDK_IS_X11_KEYMAP(object) \
((object), GDK_TYPE_X11_KEYMAP))
#define GDK_IS_X11_SCREEN(object) \
((object), GDK_TYPE_X11_SCREEN))
#define GDK_IS_X11_VISUAL(object) \
((object), GDK_TYPE_X11_VISUAL))
#define GDK_IS_X11_WINDOW(object) \
((object), GDK_TYPE_X11_WINDOW))
#define GDK_X11_DEVICE_CORE_GET_CLASS(o) \
(G_TYPE_INSTANCE_GET_CLASS ((o), \
GdkX11DeviceCoreClass))
#define GDK_X11_DEVICE_MANAGER_CORE_GET_CLASS(o) \
(G_TYPE_INSTANCE_GET_CLASS ((o), \
GdkX11DeviceManagerCoreClass))
#define GDK_X11_DEVICE_MANAGER_XI2_GET_CLASS(o) \
(G_TYPE_INSTANCE_GET_CLASS ((o), \
GdkX11DeviceManagerXI2Class))
#define GDK_X11_DEVICE_XI2_GET_CLASS(o) \
(G_TYPE_INSTANCE_GET_CLASS ((o), \
GdkX11DeviceXI2Class))
#define GDK_X11_APP_LAUNCH_CONTEXT_GET_CLASS(obj) \
(G_TYPE_INSTANCE_GET_CLASS ((obj), \
GdkX11AppLaunchContextClass))
#define GDK_X11_CURSOR_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_CURSOR, \
GdkX11CursorClass))
#define GDK_X11_DISPLAY_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_DISPLAY, \
GdkX11DisplayClass))
#define GDK_X11_DISPLAY_MANAGER_GET_CLASS(obj) \
(G_TYPE_INSTANCE_GET_CLASS ((obj), \
GdkX11DisplayManagerClass))
#define GDK_X11_DRAG_CONTEXT_GET_CLASS(obj) \
(G_TYPE_INSTANCE_GET_CLASS ((obj), \
GdkX11DragContextClass))
#define GDK_X11_KEYMAP_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_KEYMAP, \
GdkX11KeymapClass))
#define GDK_X11_SCREEN_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_SCREEN, \
GdkX11ScreenClass))
#define GDK_X11_VISUAL_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_VISUAL, \
GdkX11VisualClass))
#define GDK_X11_WINDOW_GET_CLASS(obj) \
((obj), GDK_TYPE_X11_WINDOW, \
GdkX11WindowClass))
#define GDK_POINTER_TO_XID(pointer) \
GPOINTER_TO_UINT(pointer)
#define GDK_XID_TO_POINTER(pointer) \
GUINT_TO_POINTER(pointer)

extern GType gdk_x11_app_launch_context_get_type(void);
extern Atom gdk_x11_atom_to_xatom(GdkAtom atom);
extern Atom gdk_x11_atom_to_xatom_for_display(GdkDisplay * display,
GdkAtom);
extern GType gdk_x11_cursor_get_type(void);
extern Cursor gdk_x11_cursor_get_xcursor(GdkCursor * cursor);
extern Display *gdk_x11_cursor_get_xdisplay(GdkCursor * cursor);
extern GType gdk_x11_device_core_get_type(void);
extern gint gdk_x11_device_get_id(GdkDevice * device);
extern GType gdk_x11_device_manager_core_get_type(void);
extern GdkDevice *gdk_x11_device_manager_lookup(GdkDeviceManager *
device_manager, gint);
extern GType gdk_x11_device_manager_xi2_get_type(void);
extern GType gdk_x11_device_xi2_get_type(void);
extern void gdk_x11_display_broadcast_startup_message(GdkDisplay *
display,
const char *, ...);
extern gint gdk_x11_display_error_trap_pop(GdkDisplay * display);

```

```

extern void gdk_x11_display_error_trap_pop_ignored(GdkDisplay *
display);
extern void gdk_x11_display_error_trap_push(GdkDisplay * display);
extern          const          char
*gdk_x11_display_get_startup_notification_id(GdkDisplay *
display);

extern GType gdk_x11_display_get_type(void);
extern guint32 gdk_x11_display_get_user_time(GdkDisplay * display);
extern Display *gdk_x11_display_get_xdisplay(GdkDisplay * display);
extern void gdk_x11_display_grab(GdkDisplay * display);
extern GType gdk_x11_display_manager_get_type(void);
extern void gdk_x11_display_set_cursor_theme(GdkDisplay * display,
const char *, gint);
extern void gdk_x11_display_set_startup_notification_id(GdkDisplay
*
display,
const char *);
extern gint gdk_x11_display_string_to_compound_text(GdkDisplay *
display,
const char *,
GdkAtom *, gint *,
guchar *, gint *);
extern gint gdk_x11_display_text_property_to_text_list(GdkDisplay
*
display, GdkAtom,
gint,
const unsigned char
*, gint,
gchar * **);
extern void gdk_x11_display_ungrab(GdkDisplay * display);
extern gboolean gdk_x11_display_utf8_to_compound_text(GdkDisplay *
display,
const char *,
GdkAtom *, gint *,
guchar * *, gint *);

extern GType gdk_x11_drag_context_get_type(void);
extern void gdk_x11_free_compound_text(guchar * ctext);
extern void gdk_x11_free_text_list(gchar * *list);
extern Window gdk_x11_get_default_root_xwindow(void);
extern gint gdk_x11_get_default_screen(void);
extern Display *gdk_x11_get_default_xdisplay(void);
extern guint32 gdk_x11_get_server_time(GdkWindow * window);
extern Atom gdk_x11_get_xatom_by_name(const char *atom_name);
extern Atom gdk_x11_get_xatom_by_name_for_display(GdkDisplay *
display,
const char *);
extern const char *gdk_x11_get_xatom_name(Atom xatom);
extern const char *gdk_x11_get_xatom_name_for_display(GdkDisplay *
display,
Atom);
extern void gdk_x11_grab_server(void);
extern gint gdk_x11_keymap_get_group_for_state(GdkKeymap * keymap,
gint);
extern GType gdk_x11_keymap_get_type(void);
extern gboolean gdk_x11_keymap_key_is_modifier(GdkKeymap * keymap,
gint);
extern GdkDisplay *gdk_x11_lookup_xdisplay(Display * xdisplay);
extern void gdk_x11_register_standard_event_type(GdkDisplay *
display,
gint, gint);
extern XID gdk_x11_screen_get_monitor_output(GdkScreen * screen,
gint);
extern int gdk_x11_screen_get_screen_number(GdkScreen * screen);
extern GType gdk_x11_screen_get_type(void);
extern          const          char
*gdk_x11_screen_get_window_manager_name(GdkScreen *

```



```

                                screen);
extern Screen *gdk_x11_screen_get_xscreen(GdkScreen * screen);
extern GdkVisual *gdk_x11_screen_lookup_visual(GdkScreen * screen,
                                              VisualID);
extern gboolean gdk_x11_screen_supports_net_wm_hint(GdkScreen *
screen,
                                              GdkAtom);
extern void gdk_x11_set_sm_client_id(const char *sm_client_id);
extern void gdk_x11_ungrab_server(void);
extern GType gdk_x11_visual_get_type(void);
extern Visual *gdk_x11_visual_get_xvisual(GdkVisual * visual);
extern GdkWindow
*gdk_x11_window_foreign_new_for_display(GdkDisplay *
display, Window);
extern GType gdk_x11_window_get_type(void);
extern Window gdk_x11_window_get_xid(GdkWindow * window);
extern GdkWindow *gdk_x11_window_lookup_for_display(GdkDisplay *
display,
                                              Window);
extern void gdk_x11_window_move_to_current_desktop(GdkWindow *
window);
extern void
gdk_x11_window_set_hide_titlebar_when_maximized(GdkWindow *
window,
gboolean);
extern void gdk_x11_window_set_theme_variant(GdkWindow * window,
char *);
extern void gdk_x11_window_set_user_time(GdkWindow * window,
guint32);
extern void gdk_x11_window_set_utf8_property(GdkWindow * window,
const char *, const char *);
extern GdkAtom gdk_x11_xatom_to_atom(Atom xatom);
extern GdkAtom gdk_x11_xatom_to_atom_for_display(GdkDisplay *
display,
                                              Atom);

```

7.4 Interfaces for libgtk-3

Table 7-3 defines the library name and shared object name for the libgtk-3 library

Table 7-3 libgtk-3 Definition

Library:	libgtk-3
SONAME:	libgtk-3.so.0

The behavior of the interfaces in this library is specified by the following specifications:

[Gtk3 3.6] Gtk 3.6.4 Reference Manual

7.4.1 libgtk-3 interfaces

7.4.1.1 Interfaces for libgtk-3 interfaces

An LSB conforming implementation shall provide the generic functions for libgtk-3 interfaces specified in Table 7-4, with the full mandatory functionality as described in the referenced underlying specification.

Table 7-4 libgtk-3 - libgtk-3 interfaces Function Interfaces

gtk_about_dialog_add_credit_section [Gtk3 3.6]	gtk_about_dialog_get_artists [Gtk3 3.6]
gtk_about_dialog_get_authors [Gtk3 3.6]	gtk_about_dialog_get_comments [Gtk3 3.6]
gtk_about_dialog_get_copyright [Gtk3 3.6]	gtk_about_dialog_get_documenters [Gtk3 3.6]
gtk_about_dialog_get_license [Gtk3 3.6]	gtk_about_dialog_get_license_type [Gtk3 3.6]
gtk_about_dialog_get_logo [Gtk3 3.6]	gtk_about_dialog_get_logo_icon_name [Gtk3 3.6]
gtk_about_dialog_get_program_name [Gtk3 3.6]	gtk_about_dialog_get_translator_credits [Gtk3 3.6]
gtk_about_dialog_get_version [Gtk3 3.6]	gtk_about_dialog_get_website [Gtk3 3.6]
gtk_about_dialog_get_website_label [Gtk3 3.6]	gtk_about_dialog_get_wrap_license [Gtk3 3.6]
gtk_about_dialog_new [Gtk3 3.6]	gtk_about_dialog_set_artists [Gtk3 3.6]
gtk_about_dialog_set_authors [Gtk3 3.6]	gtk_about_dialog_set_comments [Gtk3 3.6]
gtk_about_dialog_set_copyright [Gtk3 3.6]	gtk_about_dialog_set_documenters [Gtk3 3.6]
gtk_about_dialog_set_license [Gtk3 3.6]	gtk_about_dialog_set_license_type [Gtk3 3.6]
gtk_about_dialog_set_logo [Gtk3 3.6]	gtk_about_dialog_set_logo_icon_name [Gtk3 3.6]
gtk_about_dialog_set_program_name [Gtk3 3.6]	gtk_about_dialog_set_translator_credits [Gtk3 3.6]
gtk_about_dialog_set_version [Gtk3 3.6]	gtk_about_dialog_set_website [Gtk3 3.6]
gtk_about_dialog_set_website_label [Gtk3 3.6]	gtk_about_dialog_set_wrap_license [Gtk3 3.6]
gtk_accel_group_activate [Gtk3 3.6]	gtk_accel_group_connect [Gtk3 3.6]
gtk_accel_group_connect_by_path [Gtk3 3.6]	gtk_accel_group_disconnect [Gtk3 3.6]
gtk_accel_group_disconnect_key [Gtk3 3.6]	gtk_accel_group_find [Gtk3 3.6]
gtk_accel_group_from_accel_closure [Gtk3 3.6]	gtk_accel_group_get_is_locked [Gtk3 3.6]

gtk_accel_group_get_modifier_mask [Gtk3 3.6]	gtk_accel_group_lock [Gtk3 3.6]
gtk_accel_group_new [Gtk3 3.6]	gtk_accel_group_unlock [Gtk3 3.6]
gtk_accel_groups_activate [Gtk3 3.6]	gtk_accel_groups_from_object [Gtk3 3.6]
gtk_accel_label_get_accel_widget [Gtk3 3.6]	gtk_accel_label_get_accel_width [Gtk3 3.6]
gtk_accel_label_new [Gtk3 3.6]	gtk_accel_label_refetch [Gtk3 3.6]
gtk_accel_label_set_accel [Gtk3 3.6]	gtk_accel_label_set_accel_closure [Gtk3 3.6]
gtk_accel_label_set_accel_widget [Gtk3 3.6]	gtk_accel_map_add_entry [Gtk3 3.6]
gtk_accel_map_add_filter [Gtk3 3.6]	gtk_accel_map_change_entry [Gtk3 3.6]
gtk_accel_map_foreach [Gtk3 3.6]	gtk_accel_map_foreach_unfiltered [Gtk3 3.6]
gtk_accel_map_get [Gtk3 3.6]	gtk_accel_map_load [Gtk3 3.6]
gtk_accel_map_load_fd [Gtk3 3.6]	gtk_accel_map_load_scanner [Gtk3 3.6]
gtk_accel_map_lock_path [Gtk3 3.6]	gtk_accel_map_lookup_entry [Gtk3 3.6]
gtk_accel_map_save [Gtk3 3.6]	gtk_accel_map_save_fd [Gtk3 3.6]
gtk_accel_map_unlock_path [Gtk3 3.6]	gtk_accelerator_get_default_mod_mask [Gtk3 3.6]
gtk_accelerator_get_label [Gtk3 3.6]	gtk_accelerator_get_label_with_keycode [Gtk3 3.6]
gtk_accelerator_name [Gtk3 3.6]	gtk_accelerator_name_with_keycode [Gtk3 3.6]
gtk_accelerator_parse [Gtk3 3.6]	gtk_accelerator_parse_with_keycode [Gtk3 3.6]
gtk_accelerator_set_default_mod_mask [Gtk3 3.6]	gtk_accelerator_valid [Gtk3 3.6]
gtk_accessible_connect_widget_destroyed [Gtk3 3.6]	gtk_accessible_get_widget [Gtk3 3.6]
gtk_accessible_set_widget [Gtk3 3.6]	gtk_action_activate [Gtk3 3.6]
gtk_action_block_activate [Gtk3 3.6]	gtk_action_connect_accelerator [Gtk3 3.6]
gtk_action_create_icon [Gtk3 3.6]	gtk_action_create_menu [Gtk3 3.6]
gtk_action_create_menu_item [Gtk3 3.6]	gtk_action_create_tool_item [Gtk3 3.6]

gtk_action_disconnect_accelerator [Gtk3 3.6]	gtk_action_get_accel_closure [Gtk3 3.6]
gtk_action_get_accel_path [Gtk3 3.6]	gtk_action_get_always_show_image [Gtk3 3.6]
gtk_action_get_gicon [Gtk3 3.6]	gtk_action_get_icon_name [Gtk3 3.6]
gtk_action_get_is_important [Gtk3 3.6]	gtk_action_get_label [Gtk3 3.6]
gtk_action_get_name [Gtk3 3.6]	gtk_action_get_proxies [Gtk3 3.6]
gtk_action_get_sensitive [Gtk3 3.6]	gtk_action_get_short_label [Gtk3 3.6]
gtk_action_get_stock_id [Gtk3 3.6]	gtk_action_get_tooltip [Gtk3 3.6]
gtk_action_get_visible [Gtk3 3.6]	gtk_action_get_visible_horizontal [Gtk3 3.6]
gtk_action_get_visible_vertical [Gtk3 3.6]	gtk_action_group_add_action [Gtk3 3.6]
gtk_action_group_add_action_with_accel [Gtk3 3.6]	gtk_action_group_add_actions [Gtk3 3.6]
gtk_action_group_add_actions_full [Gtk3 3.6]	gtk_action_group_add_radio_actions [Gtk3 3.6]
gtk_action_group_add_radio_actions_full [Gtk3 3.6]	gtk_action_group_add_toggle_actions [Gtk3 3.6]
gtk_action_group_add_toggle_actions_full [Gtk3 3.6]	gtk_action_group_get_accel_group [Gtk3 3.6]
gtk_action_group_get_action [Gtk3 3.6]	gtk_action_group_get_name [Gtk3 3.6]
gtk_action_group_get_sensitive [Gtk3 3.6]	gtk_action_group_get_visible [Gtk3 3.6]
gtk_action_group_list_actions [Gtk3 3.6]	gtk_action_group_new [Gtk3 3.6]
gtk_action_group_remove_action [Gtk3 3.6]	gtk_action_group_set_accel_group [Gtk3 3.6]
gtk_action_group_set_sensitive [Gtk3 3.6]	gtk_action_group_set_translate_func [Gtk3 3.6]
gtk_action_group_set_translation_domain [Gtk3 3.6]	gtk_action_group_set_visible [Gtk3 3.6]
gtk_action_group_translate_string [Gtk3 3.6]	gtk_action_is_sensitive [Gtk3 3.6]
gtk_action_is_visible [Gtk3 3.6]	gtk_action_new [Gtk3 3.6]
gtk_action_set_accel_group [Gtk3 3.6]	gtk_action_set_accel_path [Gtk3 3.6]
gtk_action_set_always_show_image [Gtk3 3.6]	gtk_action_set_gicon [Gtk3 3.6]

gtk_action_set_icon_name [Gtk3 3.6]	gtk_action_set_is_important [Gtk3 3.6]
gtk_action_set_label [Gtk3 3.6]	gtk_action_set_sensitive [Gtk3 3.6]
gtk_action_set_short_label [Gtk3 3.6]	gtk_action_set_stock_id [Gtk3 3.6]
gtk_action_set_tooltip [Gtk3 3.6]	gtk_action_set_visible [Gtk3 3.6]
gtk_action_set_visible_horizontal [Gtk3 3.6]	gtk_action_set_visible_vertical [Gtk3 3.6]
gtk_action_unblock_activate [Gtk3 3.6]	gtk_actionable_get_action_name [Gtk3 3.6]
gtk_actionable_get_action_target_value [Gtk3 3.6]	gtk_actionable_set_action_name [Gtk3 3.6]
gtk_actionable_set_action_target [Gtk3 3.6]	gtk_actionable_set_action_target_value [Gtk3 3.6]
gtk_actionable_set_detailed_action_name [Gtk3 3.6]	gtk_activatable_do_set_related_action [Gtk3 3.6]
gtk_activatable_get_related_action [Gtk3 3.6]	gtk_activatable_get_use_action_appearance [Gtk3 3.6]
gtk_activatable_set_related_action [Gtk3 3.6]	gtk_activatable_set_use_action_appearance [Gtk3 3.6]
gtk_activatable_sync_action_properties [Gtk3 3.6]	gtk_adjustment_changed [Gtk3 3.6]
gtk_adjustment_clamp_page [Gtk3 3.6]	gtk_adjustment_configure [Gtk3 3.6]
gtk_adjustment_get_lower [Gtk3 3.6]	gtk_adjustment_get_minimum_increment [Gtk3 3.6]
gtk_adjustment_get_page_increment [Gtk3 3.6]	gtk_adjustment_get_page_size [Gtk3 3.6]
gtk_adjustment_get_step_increment [Gtk3 3.6]	gtk_adjustment_get_upper [Gtk3 3.6]
gtk_adjustment_get_value [Gtk3 3.6]	gtk_adjustment_new [Gtk3 3.6]
gtk_adjustment_set_lower [Gtk3 3.6]	gtk_adjustment_set_page_increment [Gtk3 3.6]
gtk_adjustment_set_page_size [Gtk3 3.6]	gtk_adjustment_set_step_increment [Gtk3 3.6]
gtk_adjustment_set_upper [Gtk3 3.6]	gtk_adjustment_set_value [Gtk3 3.6]
gtk_adjustment_value_changed [Gtk3 3.6]	gtk_alignment_get_padding [Gtk3 3.6]
gtk_alignment_new [Gtk3 3.6]	gtk_alignment_set [Gtk3 3.6]
gtk_alignment_set_padding [Gtk3 3.6]	gtk_alternative_dialog_button_order [Gtk3 3.6]

gtk_app_chooser_button_append_custom_item [Gtk3 3.6]	gtk_app_chooser_button_append_separator [Gtk3 3.6]
gtk_app_chooser_button_get_heading [Gtk3 3.6]	gtk_app_chooser_button_get_show_default_item [Gtk3 3.6]
gtk_app_chooser_button_get_show_dialog_item [Gtk3 3.6]	gtk_app_chooser_button_new [Gtk3 3.6]
gtk_app_chooser_button_set_active_custom_item [Gtk3 3.6]	gtk_app_chooser_button_set_heading [Gtk3 3.6]
gtk_app_chooser_button_set_show_default_item [Gtk3 3.6]	gtk_app_chooser_button_set_show_dialog_item [Gtk3 3.6]
gtk_app_chooser_dialog_get_heading [Gtk3 3.6]	gtk_app_chooser_dialog_get_widget [Gtk3 3.6]
gtk_app_chooser_dialog_new [Gtk3 3.6]	gtk_app_chooser_dialog_new_for_content_type [Gtk3 3.6]
gtk_app_chooser_dialog_set_heading [Gtk3 3.6]	gtk_app_chooser_get_app_info [Gtk3 3.6]
gtk_app_chooser_get_content_type [Gtk3 3.6]	gtk_app_chooser_refresh [Gtk3 3.6]
gtk_app_chooser_widget_get_default_text [Gtk3 3.6]	gtk_app_chooser_widget_get_show_all [Gtk3 3.6]
gtk_app_chooser_widget_get_show_default [Gtk3 3.6]	gtk_app_chooser_widget_get_show_fallback [Gtk3 3.6]
gtk_app_chooser_widget_get_show_other [Gtk3 3.6]	gtk_app_chooser_widget_get_show_recommended [Gtk3 3.6]
gtk_app_chooser_widget_new [Gtk3 3.6]	gtk_app_chooser_widget_set_default_text [Gtk3 3.6]
gtk_app_chooser_widget_set_show_all [Gtk3 3.6]	gtk_app_chooser_widget_set_show_default [Gtk3 3.6]
gtk_app_chooser_widget_set_show_fallback [Gtk3 3.6]	gtk_app_chooser_widget_set_show_other [Gtk3 3.6]
gtk_app_chooser_widget_set_show_recommended [Gtk3 3.6]	gtk_application_add_accelerator [Gtk3 3.6]
gtk_application_add_window [Gtk3 3.6]	gtk_application_get_active_window [Gtk3 3.6]
gtk_application_get_app_menu [Gtk3 3.6]	gtk_application_get_menubar [Gtk3 3.6]
gtk_application_get_window_by_id [Gtk3 3.6]	gtk_application_get_windows [Gtk3 3.6]
gtk_application_inhibit [Gtk3 3.6]	gtk_application_is_inhibited [Gtk3 3.6]
gtk_application_new [Gtk3 3.6]	gtk_application_remove_accelerator [Gtk3 3.6]

gtk_application_remove_window [Gtk3 3.6]	gtk_application_set_app_menu [Gtk3 3.6]
gtk_application_set_menubar [Gtk3 3.6]	gtk_application_uninhibit [Gtk3 3.6]
gtk_application_window_get_id [Gtk3 3.6]	gtk_application_window_get_show_menubar [Gtk3 3.6]
gtk_application_window_new [Gtk3 3.6]	gtk_application_window_set_show_menubar [Gtk3 3.6]
gtk_arrow_new [Gtk3 3.6]	gtk_arrow_set [Gtk3 3.6]
gtk_aspect_frame_new [Gtk3 3.6]	gtk_aspect_frame_set [Gtk3 3.6]
gtk_assistant_add_action_widget [Gtk3 3.6]	gtk_assistant_append_page [Gtk3 3.6]
gtk_assistant_commit [Gtk3 3.6]	gtk_assistant_get_current_page [Gtk3 3.6]
gtk_assistant_get_n_pages [Gtk3 3.6]	gtk_assistant_get_nth_page [Gtk3 3.6]
gtk_assistant_get_page_complete [Gtk3 3.6]	gtk_assistant_get_page_header_image [Gtk3 3.6]
gtk_assistant_get_page_side_image [Gtk3 3.6]	gtk_assistant_get_page_title [Gtk3 3.6]
gtk_assistant_get_page_type [Gtk3 3.6]	gtk_assistant_insert_page [Gtk3 3.6]
gtk_assistant_new [Gtk3 3.6]	gtk_assistant_next_page [Gtk3 3.6]
gtk_assistant_prepend_page [Gtk3 3.6]	gtk_assistant_previous_page [Gtk3 3.6]
gtk_assistant_remove_action_widget [Gtk3 3.6]	gtk_assistant_remove_page [Gtk3 3.6]
gtk_assistant_set_current_page [Gtk3 3.6]	gtk_assistant_set_forward_page_function [Gtk3 3.6]
gtk_assistant_set_page_complete [Gtk3 3.6]	gtk_assistant_set_page_header_image [Gtk3 3.6]
gtk_assistant_set_page_side_image [Gtk3 3.6]	gtk_assistant_set_page_title [Gtk3 3.6]
gtk_assistant_set_page_type [Gtk3 3.6]	gtk_assistant_update_buttons_state [Gtk3 3.6]
gtk_bin_get_child [Gtk3 3.6]	gtk_binding_entry_add_signal [Gtk3 3.6]
gtk_binding_entry_add_signal_from_string [Gtk3 3.6]	gtk_binding_entry_add_signal1 [Gtk3 3.6]
gtk_binding_entry_remove [Gtk3 3.6]	gtk_binding_entry_skip [Gtk3 3.6]
gtk_binding_set_activate [Gtk3 3.6]	gtk_binding_set_add_path [Gtk3 3.6]

gtk_binding_set_by_class [Gtk3 3.6]	gtk_binding_set_find [Gtk3 3.6]
gtk_binding_set_new [Gtk3 3.6]	gtk_bindings_activate [Gtk3 3.6]
gtk_bindings_activate_event [Gtk3 3.6]	gtk_border_copy [Gtk3 3.6]
gtk_border_free [Gtk3 3.6]	gtk_border_new [Gtk3 3.6]
gtk_box_get_homogeneous [Gtk3 3.6]	gtk_box_get_spacing [Gtk3 3.6]
gtk_box_new [Gtk3 3.6]	gtk_box_pack_end [Gtk3 3.6]
gtk_box_pack_start [Gtk3 3.6]	gtk_box_query_child_packing [Gtk3 3.6]
gtk_box_reorder_child [Gtk3 3.6]	gtk_box_set_child_packing [Gtk3 3.6]
gtk_box_set_homogeneous [Gtk3 3.6]	gtk_box_set_spacing [Gtk3 3.6]
gtk_buildable_add_child [Gtk3 3.6]	gtk_buildable_construct_child [Gtk3 3.6]
gtk_buildable_custom_finished [Gtk3 3.6]	gtk_buildable_custom_tag_end [Gtk3 3.6]
gtk_buildable_custom_tag_start [Gtk3 3.6]	gtk_buildable_get_internal_child [Gtk3 3.6]
gtk_buildable_get_name [Gtk3 3.6]	gtk_buildable_parser_finished [Gtk3 3.6]
gtk_buildable_set_buildable_property [Gtk3 3.6]	gtk_buildable_set_name [Gtk3 3.6]
gtk_builder_add_from_file [Gtk3 3.6]	gtk_builder_add_from_resource [Gtk3 3.6]
gtk_builder_add_from_string [Gtk3 3.6]	gtk_builder_add_objects_from_file [Gtk3 3.6]
gtk_builder_add_objects_from_resource [Gtk3 3.6]	gtk_builder_add_objects_from_string [Gtk3 3.6]
gtk_builder_connect_signals [Gtk3 3.6]	gtk_builder_connect_signals_full [Gtk3 3.6]
gtk_builder_get_object [Gtk3 3.6]	gtk_builder_get_objects [Gtk3 3.6]
gtk_builder_get_translation_domain [Gtk3 3.6]	gtk_builder_get_type_from_name [Gtk3 3.6]
gtk_builder_new [Gtk3 3.6]	gtk_builder_set_translation_domain [Gtk3 3.6]
gtk_builder_value_from_string [Gtk3 3.6]	gtk_builder_value_from_string_type [Gtk3 3.6]
gtk_button_box_get_child_non_homogeneous [Gtk3 3.6]	gtk_button_box_get_child_secondary [Gtk3 3.6]
gtk_button_box_get_layout [Gtk3 3.6]	gtk_button_box_new [Gtk3 3.6]

gtk_button_box_set_child_non_homogeneous [Gtk3 3.6]	gtk_button_box_set_child_secondary [Gtk3 3.6]
gtk_button_box_set_layout [Gtk3 3.6]	gtk_button_clicked [Gtk3 3.6]
gtk_button_enter [Gtk3 3.6]	gtk_button_get_alignment [Gtk3 3.6]
gtk_button_get_always_show_image [Gtk3 3.6]	gtk_button_get_event_window [Gtk3 3.6]
gtk_button_get_focus_on_click [Gtk3 3.6]	gtk_button_get_image [Gtk3 3.6]
gtk_button_get_image_position [Gtk3 3.6]	gtk_button_get_label [Gtk3 3.6]
gtk_button_get_relief [Gtk3 3.6]	gtk_button_get_use_stock [Gtk3 3.6]
gtk_button_get_use_underline [Gtk3 3.6]	gtk_button_leave [Gtk3 3.6]
gtk_button_new [Gtk3 3.6]	gtk_button_new_from_stock [Gtk3 3.6]
gtk_button_new_with_label [Gtk3 3.6]	gtk_button_new_with_mnemonic [Gtk3 3.6]
gtk_button_pressed [Gtk3 3.6]	gtk_button_released [Gtk3 3.6]
gtk_button_set_alignment [Gtk3 3.6]	gtk_button_set_always_show_image [Gtk3 3.6]
gtk_button_set_focus_on_click [Gtk3 3.6]	gtk_button_set_image [Gtk3 3.6]
gtk_button_set_image_position [Gtk3 3.6]	gtk_button_set_label [Gtk3 3.6]
gtk_button_set_relief [Gtk3 3.6]	gtk_button_set_use_stock [Gtk3 3.6]
gtk_button_set_use_underline [Gtk3 3.6]	gtk_cairo_should_draw_window [Gtk3 3.6]
gtk_cairo_transform_to_window [Gtk3 3.6]	gtk_calendar_clear_marks [Gtk3 3.6]
gtk_calendar_get_date [Gtk3 3.6]	gtk_calendar_get_day_is_marked [Gtk3 3.6]
gtk_calendar_get_detail_height_rows [Gtk3 3.6]	gtk_calendar_get_detail_width_chars [Gtk3 3.6]
gtk_calendar_get_display_options [Gtk3 3.6]	gtk_calendar_mark_day [Gtk3 3.6]
gtk_calendar_new [Gtk3 3.6]	gtk_calendar_select_day [Gtk3 3.6]
gtk_calendar_select_month [Gtk3 3.6]	gtk_calendar_set_detail_func [Gtk3 3.6]
gtk_calendar_set_detail_height_rows [Gtk3 3.6]	gtk_calendar_set_detail_width_chars [Gtk3 3.6]

gtk_calendar_set_display_options [Gtk3 3.6]	gtk_calendar_unmark_day [Gtk3 3.6]
gtk_cell_area_activate [Gtk3 3.6]	gtk_cell_area_activate_cell [Gtk3 3.6]
gtk_cell_area_add [Gtk3 3.6]	gtk_cell_area_add_focus_sibling [Gtk3 3.6]
gtk_cell_area_add_with_properties [Gtk3 3.6]	gtk_cell_area_apply_attributes [Gtk3 3.6]
gtk_cell_area_attribute_connect [Gtk3 3.6]	gtk_cell_area_attribute_disconnect [Gtk3 3.6]
gtk_cell_area_box_get_spacing [Gtk3 3.6]	gtk_cell_area_box_new [Gtk3 3.6]
gtk_cell_area_box_pack_end [Gtk3 3.6]	gtk_cell_area_box_pack_start [Gtk3 3.6]
gtk_cell_area_box_set_spacing [Gtk3 3.6]	gtk_cell_area_cell_get [Gtk3 3.6]
gtk_cell_area_cell_get_property [Gtk3 3.6]	gtk_cell_area_cell_get_valist [Gtk3 3.6]
gtk_cell_area_cell_set [Gtk3 3.6]	gtk_cell_area_cell_set_property [Gtk3 3.6]
gtk_cell_area_cell_set_valist [Gtk3 3.6]	gtk_cell_area_class_find_cell_property [Gtk3 3.6]
gtk_cell_area_class_install_cell_property [Gtk3 3.6]	gtk_cell_area_class_list_cell_properties [Gtk3 3.6]
gtk_cell_area_context_allocate [Gtk3 3.6]	gtk_cell_area_context_get_allocation [Gtk3 3.6]
gtk_cell_area_context_get_area [Gtk3 3.6]	gtk_cell_area_context_get_preferred_height [Gtk3 3.6]
gtk_cell_area_context_get_preferred_height_for_width [Gtk3 3.6]	gtk_cell_area_context_get_preferred_width [Gtk3 3.6]
gtk_cell_area_context_get_preferred_width_for_height [Gtk3 3.6]	gtk_cell_area_context_push_preferred_height [Gtk3 3.6]
gtk_cell_area_context_push_preferred_width [Gtk3 3.6]	gtk_cell_area_context_reset [Gtk3 3.6]
gtk_cell_area_copy_context [Gtk3 3.6]	gtk_cell_area_create_context [Gtk3 3.6]
gtk_cell_area_event [Gtk3 3.6]	gtk_cell_area_focus [Gtk3 3.6]
gtk_cell_area_foreach [Gtk3 3.6]	gtk_cell_area_foreach_alloc [Gtk3 3.6]
gtk_cell_area_get_cell_allocation [Gtk3 3.6]	gtk_cell_area_get_cell_at_position [Gtk3 3.6]
gtk_cell_area_get_current_path_string [Gtk3 3.6]	gtk_cell_area_get_edit_widget [Gtk3 3.6]

gtk_cell_area_get_edited_cell [Gtk3 3.6]	gtk_cell_area_get_focus_cell [Gtk3 3.6]
gtk_cell_area_get_focus_from_sibling [Gtk3 3.6]	gtk_cell_area_get_focus_siblings [Gtk3 3.6]
gtk_cell_area_get_preferred_height [Gtk3 3.6]	gtk_cell_area_get_preferred_height_for_width [Gtk3 3.6]
gtk_cell_area_get_preferred_width [Gtk3 3.6]	gtk_cell_area_get_preferred_width_for_height [Gtk3 3.6]
gtk_cell_area_get_request_mode [Gtk3 3.6]	gtk_cell_area_has_renderer [Gtk3 3.6]
gtk_cell_area_inner_cell_area [Gtk3 3.6]	gtk_cell_area_is_activatable [Gtk3 3.6]
gtk_cell_area_is_focus_sibling [Gtk3 3.6]	gtk_cell_area_remove [Gtk3 3.6]
gtk_cell_area_remove_focus_sibling [Gtk3 3.6]	gtk_cell_area_render [Gtk3 3.6]
gtk_cell_area_request_renderer [Gtk3 3.6]	gtk_cell_area_set_focus_cell [Gtk3 3.6]
gtk_cell_area_stop_editing [Gtk3 3.6]	gtk_cell_editable_editing_done [Gtk3 3.6]
gtk_cell_editable_remove_widget [Gtk3 3.6]	gtk_cell_editable_start_editing [Gtk3 3.6]
gtk_cell_layout_add_attribute [Gtk3 3.6]	gtk_cell_layout_clear [Gtk3 3.6]
gtk_cell_layout_clear_attributes [Gtk3 3.6]	gtk_cell_layout_get_area [Gtk3 3.6]
gtk_cell_layout_get_cells [Gtk3 3.6]	gtk_cell_layout_pack_end [Gtk3 3.6]
gtk_cell_layout_pack_start [Gtk3 3.6]	gtk_cell_layout_reorder [Gtk3 3.6]
gtk_cell_layout_set_attributes [Gtk3 3.6]	gtk_cell_layout_set_cell_data_func [Gtk3 3.6]
gtk_cell_renderer_accel_new [Gtk3 3.6]	gtk_cell_renderer_activate [Gtk3 3.6]
gtk_cell_renderer_combo_new [Gtk3 3.6]	gtk_cell_renderer_get_aligned_area [Gtk3 3.6]
gtk_cell_renderer_get_alignment [Gtk3 3.6]	gtk_cell_renderer_get_fixed_size [Gtk3 3.6]
gtk_cell_renderer_get_padding [Gtk3 3.6]	gtk_cell_renderer_get_preferred_height [Gtk3 3.6]
gtk_cell_renderer_get_preferred_height_for_width [Gtk3 3.6]	gtk_cell_renderer_get_preferred_size [Gtk3 3.6]
gtk_cell_renderer_get_preferred_width [Gtk3 3.6]	gtk_cell_renderer_get_preferred_width_for_height [Gtk3 3.6]

gtk_cell_renderer_get_request_mode [Gtk3 3.6]	gtk_cell_renderer_get_sensitive [Gtk3 3.6]
gtk_cell_renderer_get_size [Gtk3 3.6]	gtk_cell_renderer_get_state [Gtk3 3.6]
gtk_cell_renderer_get_visible [Gtk3 3.6]	gtk_cell_renderer_is_activatable [Gtk3 3.6]
gtk_cell_renderer_pixbuf_new [Gtk3 3.6]	gtk_cell_renderer_progress_new [Gtk3 3.6]
gtk_cell_renderer_render [Gtk3 3.6]	gtk_cell_renderer_set_alignment [Gtk3 3.6]
gtk_cell_renderer_set_fixed_size [Gtk3 3.6]	gtk_cell_renderer_set_padding [Gtk3 3.6]
gtk_cell_renderer_set_sensitive [Gtk3 3.6]	gtk_cell_renderer_set_visible [Gtk3 3.6]
gtk_cell_renderer_spin_new [Gtk3 3.6]	gtk_cell_renderer_spinner_new [Gtk3 3.6]
gtk_cell_renderer_start_editing [Gtk3 3.6]	gtk_cell_renderer_stop_editing [Gtk3 3.6]
gtk_cell_renderer_text_new [Gtk3 3.6]	gtk_cell_renderer_text_set_fixed_height_from_font [Gtk3 3.6]
gtk_cell_renderer_toggle_get_activatable [Gtk3 3.6]	gtk_cell_renderer_toggle_get_active [Gtk3 3.6]
gtk_cell_renderer_toggle_get_radio [Gtk3 3.6]	gtk_cell_renderer_toggle_new [Gtk3 3.6]
gtk_cell_renderer_toggle_set_activatable [Gtk3 3.6]	gtk_cell_renderer_toggle_set_active [Gtk3 3.6]
gtk_cell_renderer_toggle_set_radio [Gtk3 3.6]	gtk_cell_view_get_displayed_row [Gtk3 3.6]
gtk_cell_view_get_draw_sensitive [Gtk3 3.6]	gtk_cell_view_get_fit_model [Gtk3 3.6]
gtk_cell_view_get_model [Gtk3 3.6]	gtk_cell_view_get_size_of_row [Gtk3 3.6]
gtk_cell_view_new [Gtk3 3.6]	gtk_cell_view_new_with_context [Gtk3 3.6]
gtk_cell_view_new_with_markup [Gtk3 3.6]	gtk_cell_view_new_with_pixbuf [Gtk3 3.6]
gtk_cell_view_new_with_text [Gtk3 3.6]	gtk_cell_view_set_background_color [Gtk3 3.6]
gtk_cell_view_set_background_rgba [Gtk3 3.6]	gtk_cell_view_set_displayed_row [Gtk3 3.6]
gtk_cell_view_set_draw_sensitive [Gtk3 3.6]	gtk_cell_view_set_fit_model [Gtk3 3.6]

gtk_cell_view_set_model [Gtk3 3.6]	gtk_check_button_new [Gtk3 3.6]
gtk_check_button_new_with_label [Gtk3 3.6]	gtk_check_button_new_with_mnemonic [Gtk3 3.6]
gtk_check_menu_item_get_active [Gtk3 3.6]	gtk_check_menu_item_get_draw_as_radio [Gtk3 3.6]
gtk_check_menu_item_get_inconsistent [Gtk3 3.6]	gtk_check_menu_item_new [Gtk3 3.6]
gtk_check_menu_item_new_with_label [Gtk3 3.6]	gtk_check_menu_item_new_with_mnemonic [Gtk3 3.6]
gtk_check_menu_item_set_active [Gtk3 3.6]	gtk_check_menu_item_set_draw_as_radio [Gtk3 3.6]
gtk_check_menu_item_set_inconsistent [Gtk3 3.6]	gtk_check_menu_item_toggled [Gtk3 3.6]
gtk_check_version [Gtk3 3.6]	gtk_clipboard_clear [Gtk3 3.6]
gtk_clipboard_get [Gtk3 3.6]	gtk_clipboard_get_display [Gtk3 3.6]
gtk_clipboard_get_for_display [Gtk3 3.6]	gtk_clipboard_get_owner [Gtk3 3.6]
gtk_clipboard_request_contents [Gtk3 3.6]	gtk_clipboard_request_image [Gtk3 3.6]
gtk_clipboard_request_rich_text [Gtk3 3.6]	gtk_clipboard_request_targets [Gtk3 3.6]
gtk_clipboard_request_text [Gtk3 3.6]	gtk_clipboard_request_uris [Gtk3 3.6]
gtk_clipboard_set_can_store [Gtk3 3.6]	gtk_clipboard_set_image [Gtk3 3.6]
gtk_clipboard_set_text [Gtk3 3.6]	gtk_clipboard_set_with_data [Gtk3 3.6]
gtk_clipboard_set_with_owner [Gtk3 3.6]	gtk_clipboard_store [Gtk3 3.6]
gtk_clipboard_wait_for_contents [Gtk3 3.6]	gtk_clipboard_wait_for_image [Gtk3 3.6]
gtk_clipboard_wait_for_rich_text [Gtk3 3.6]	gtk_clipboard_wait_for_targets [Gtk3 3.6]
gtk_clipboard_wait_for_text [Gtk3 3.6]	gtk_clipboard_wait_for_uris [Gtk3 3.6]
gtk_clipboard_wait_is_image_available [Gtk3 3.6]	gtk_clipboard_wait_is_rich_text_available [Gtk3 3.6]
gtk_clipboard_wait_is_target_available [Gtk3 3.6]	gtk_clipboard_wait_is_text_available [Gtk3 3.6]
gtk_clipboard_wait_is_uris_available [Gtk3 3.6]	gtk_color_button_get_alpha [Gtk3 3.6]

gtk_color_button_get_color [Gtk3 3.6]	gtk_color_button_get_rgba [Gtk3 3.6]
gtk_color_button_get_title [Gtk3 3.6]	gtk_color_button_get_use_alpha [Gtk3 3.6]
gtk_color_button_new [Gtk3 3.6]	gtk_color_button_new_with_color [Gtk3 3.6]
gtk_color_button_new_with_rgba [Gtk3 3.6]	gtk_color_button_set_alpha [Gtk3 3.6]
gtk_color_button_set_color [Gtk3 3.6]	gtk_color_button_set_rgba [Gtk3 3.6]
gtk_color_button_set_title [Gtk3 3.6]	gtk_color_button_set_use_alpha [Gtk3 3.6]
gtk_color_chooser_add_palette [Gtk3 3.6]	gtk_color_chooser_dialog_new [Gtk3 3.6]
gtk_color_chooser_get_rgba [Gtk3 3.6]	gtk_color_chooser_get_use_alpha [Gtk3 3.6]
gtk_color_chooser_set_rgba [Gtk3 3.6]	gtk_color_chooser_set_use_alpha [Gtk3 3.6]
gtk_color_chooser_widget_new [Gtk3 3.6]	gtk_combo_box_get_active [Gtk3 3.6]
gtk_combo_box_get_active_id [Gtk3 3.6]	gtk_combo_box_get_active_iter [Gtk3 3.6]
gtk_combo_box_get_add_tearoffs [Gtk3 3.6]	gtk_combo_box_get_button_sensitivity [Gtk3 3.6]
gtk_combo_box_get_column_span_column [Gtk3 3.6]	gtk_combo_box_get_entry_text_column [Gtk3 3.6]
gtk_combo_box_get_focus_on_click [Gtk3 3.6]	gtk_combo_box_get_has_entry [Gtk3 3.6]
gtk_combo_box_get_id_column [Gtk3 3.6]	gtk_combo_box_get_model [Gtk3 3.6]
gtk_combo_box_get_popup_accessible [Gtk3 3.6]	gtk_combo_box_get_popup_fixed_width [Gtk3 3.6]
gtk_combo_box_get_row_separator_func [Gtk3 3.6]	gtk_combo_box_get_row_span_column [Gtk3 3.6]
gtk_combo_box_get_title [Gtk3 3.6]	gtk_combo_box_get_wrap_width [Gtk3 3.6]
gtk_combo_box_new [Gtk3 3.6]	gtk_combo_box_new_with_area [Gtk3 3.6]
gtk_combo_box_new_with_area_and_entry [Gtk3 3.6]	gtk_combo_box_new_with_entry [Gtk3 3.6]
gtk_combo_box_new_with_model [Gtk3 3.6]	gtk_combo_box_new_with_model_and_entry [Gtk3 3.6]
gtk_combo_box_popdown [Gtk3 3.6]	gtk_combo_box_popup [Gtk3 3.6]

gtk_combo_box_popup_for_device [Gtk3 3.6]	gtk_combo_box_set_active [Gtk3 3.6]
gtk_combo_box_set_active_id [Gtk3 3.6]	gtk_combo_box_set_active_iter [Gtk3 3.6]
gtk_combo_box_set_add_tearoffs [Gtk3 3.6]	gtk_combo_box_set_button_sensitivity [Gtk3 3.6]
gtk_combo_box_set_column_span_column [Gtk3 3.6]	gtk_combo_box_set_entry_text_column [Gtk3 3.6]
gtk_combo_box_set_focus_on_click [Gtk3 3.6]	gtk_combo_box_set_id_column [Gtk3 3.6]
gtk_combo_box_set_model [Gtk3 3.6]	gtk_combo_box_set_popup_fixed_width [Gtk3 3.6]
gtk_combo_box_set_row_separator_func [Gtk3 3.6]	gtk_combo_box_set_row_span_column [Gtk3 3.6]
gtk_combo_box_set_title [Gtk3 3.6]	gtk_combo_box_set_wrap_width [Gtk3 3.6]
gtk_combo_box_text_append [Gtk3 3.6]	gtk_combo_box_text_append_text [Gtk3 3.6]
gtk_combo_box_text_get_active_text [Gtk3 3.6]	gtk_combo_box_text_insert [Gtk3 3.6]
gtk_combo_box_text_insert_text [Gtk3 3.6]	gtk_combo_box_text_new [Gtk3 3.6]
gtk_combo_box_text_new_with_entry [Gtk3 3.6]	gtk_combo_box_text_prepend [Gtk3 3.6]
gtk_combo_box_text_prepend_text [Gtk3 3.6]	gtk_combo_box_text_remove [Gtk3 3.6]
gtk_combo_box_text_remove_all [Gtk3 3.6]	gtk_container_add [Gtk3 3.6]
gtk_container_add_with_properties [Gtk3 3.6]	gtk_container_check_resize [Gtk3 3.6]
gtk_container_child_get [Gtk3 3.6]	gtk_container_child_get_property [Gtk3 3.6]
gtk_container_child_get_valist [Gtk3 3.6]	gtk_container_child_notify [Gtk3 3.6]
gtk_container_child_set [Gtk3 3.6]	gtk_container_child_set_property [Gtk3 3.6]
gtk_container_child_set_valist [Gtk3 3.6]	gtk_container_child_type [Gtk3 3.6]
gtk_container_class_find_child_property [Gtk3 3.6]	gtk_container_class_handle_border_width [Gtk3 3.6]
gtk_container_class_install_child_property [Gtk3 3.6]	gtk_container_class_list_child_properties [Gtk3 3.6]

gtk_container_forall [Gtk3 3.6]	gtk_container_foreach [Gtk3 3.6]
gtk_container_get_border_width [Gtk3 3.6]	gtk_container_get_children [Gtk3 3.6]
gtk_container_get_focus_chain [Gtk3 3.6]	gtk_container_get_focus_child [Gtk3 3.6]
gtk_container_get_focus_hadjustment [Gtk3 3.6]	gtk_container_get_focus_vadjustment [Gtk3 3.6]
gtk_container_get_path_for_child [Gtk3 3.6]	gtk_container_get_resize_mode [Gtk3 3.6]
gtk_container_propagate_draw [Gtk3 3.6]	gtk_container_remove [Gtk3 3.6]
gtk_container_resize_children [Gtk3 3.6]	gtk_container_set_border_width [Gtk3 3.6]
gtk_container_set_focus_chain [Gtk3 3.6]	gtk_container_set_focus_child [Gtk3 3.6]
gtk_container_set_focus_hadjustment [Gtk3 3.6]	gtk_container_set_focus_vadjustment [Gtk3 3.6]
gtk_container_set_reallocate_redraws [Gtk3 3.6]	gtk_container_set_resize_mode [Gtk3 3.6]
gtk_container_unset_focus_chain [Gtk3 3.6]	gtk_css_provider_get_default [Gtk3 3.6]
gtk_css_provider_get_named [Gtk3 3.6]	gtk_css_provider_load_from_data [Gtk3 3.6]
gtk_css_provider_load_from_file [Gtk3 3.6]	gtk_css_provider_load_from_path [Gtk3 3.6]
gtk_css_provider_new [Gtk3 3.6]	gtk_css_provider_to_string [Gtk3 3.6]
gtk_css_section_get_end_line [Gtk3 3.6]	gtk_css_section_get_end_position [Gtk3 3.6]
gtk_css_section_get_file [Gtk3 3.6]	gtk_css_section_get_parent [Gtk3 3.6]
gtk_css_section_get_section_type [Gtk3 3.6]	gtk_css_section_get_start_line [Gtk3 3.6]
gtk_css_section_get_start_position [Gtk3 3.6]	gtk_css_section_ref [Gtk3 3.6]
gtk_css_section_unref [Gtk3 3.6]	gtk_device_grab_add [Gtk3 3.6]
gtk_device_grab_remove [Gtk3 3.6]	gtk_dialog_add_action_widget [Gtk3 3.6]
gtk_dialog_add_button [Gtk3 3.6]	gtk_dialog_add_buttons [Gtk3 3.6]
gtk_dialog_get_action_area [Gtk3 3.6]	gtk_dialog_get_content_area [Gtk3 3.6]
gtk_dialog_get_response_for_widget [Gtk3 3.6]	gtk_dialog_get_widget_for_response [Gtk3 3.6]

gtk_dialog_new [Gtk3 3.6]	gtk_dialog_new_with_buttons [Gtk3 3.6]
gtk_dialog_response [Gtk3 3.6]	gtk_dialog_run [Gtk3 3.6]
gtk_dialog_set_alternative_button_order [Gtk3 3.6]	gtk_dialog_set_alternative_button_order_from_array [Gtk3 3.6]
gtk_dialog_set_default_response [Gtk3 3.6]	gtk_dialog_set_response_sensitive [Gtk3 3.6]
gtk_disable_setlocale [Gtk3 3.6]	gtk_distribute_natural_allocation [Gtk3 3.6]
gtk_drag_begin [Gtk3 3.6]	gtk_drag_check_threshold [Gtk3 3.6]
gtk_drag_dest_add_image_targets [Gtk3 3.6]	gtk_drag_dest_add_text_targets [Gtk3 3.6]
gtk_drag_dest_add_uri_targets [Gtk3 3.6]	gtk_drag_dest_find_target [Gtk3 3.6]
gtk_drag_dest_get_target_list [Gtk3 3.6]	gtk_drag_dest_get_track_motion [Gtk3 3.6]
gtk_drag_dest_set [Gtk3 3.6]	gtk_drag_dest_set_proxy [Gtk3 3.6]
gtk_drag_dest_set_target_list [Gtk3 3.6]	gtk_drag_dest_set_track_motion [Gtk3 3.6]
gtk_drag_dest_unset [Gtk3 3.6]	gtk_drag_finish [Gtk3 3.6]
gtk_drag_get_data [Gtk3 3.6]	gtk_drag_get_source_widget [Gtk3 3.6]
gtk_drag_highlight [Gtk3 3.6]	gtk_drag_set_icon_default [Gtk3 3.6]
gtk_drag_set_icon_gicon [Gtk3 3.6]	gtk_drag_set_icon_name [Gtk3 3.6]
gtk_drag_set_icon_pixbuf [Gtk3 3.6]	gtk_drag_set_icon_stock [Gtk3 3.6]
gtk_drag_set_icon_surface [Gtk3 3.6]	gtk_drag_set_icon_widget [Gtk3 3.6]
gtk_drag_source_add_image_targets [Gtk3 3.6]	gtk_drag_source_add_text_targets [Gtk3 3.6]
gtk_drag_source_add_uri_targets [Gtk3 3.6]	gtk_drag_source_get_target_list [Gtk3 3.6]
gtk_drag_source_set [Gtk3 3.6]	gtk_drag_source_set_icon_gicon [Gtk3 3.6]
gtk_drag_source_set_icon_name [Gtk3 3.6]	gtk_drag_source_set_icon_pixbuf [Gtk3 3.6]
gtk_drag_source_set_icon_stock [Gtk3 3.6]	gtk_drag_source_set_target_list [Gtk3 3.6]
gtk_drag_source_unset [Gtk3 3.6]	gtk_drag_unhighlight [Gtk3 3.6]
gtk_draw_insertion_cursor [Gtk3 3.6]	gtk_drawing_area_new [Gtk3 3.6]
gtk_editable_copy_clipboard [Gtk3 3.6]	gtk_editable_cut_clipboard [Gtk3 3.6]

gtk_editable_delete_selection [Gtk3 3.6]	gtk_editable_delete_text [Gtk3 3.6]
gtk_editable_get_chars [Gtk3 3.6]	gtk_editable_get_editable [Gtk3 3.6]
gtk_editable_get_position [Gtk3 3.6]	gtk_editable_get_selection_bounds [Gtk3 3.6]
gtk_editable_insert_text [Gtk3 3.6]	gtk_editable_paste_clipboard [Gtk3 3.6]
gtk_editable_select_region [Gtk3 3.6]	gtk_editable_set_editable [Gtk3 3.6]
gtk_editable_set_position [Gtk3 3.6]	gtk_entry_buffer_delete_text [Gtk3 3.6]
gtk_entry_buffer_emit_deleted_text [Gtk3 3.6]	gtk_entry_buffer_emit_inserted_text [Gtk3 3.6]
gtk_entry_buffer_get_bytes [Gtk3 3.6]	gtk_entry_buffer_get_length [Gtk3 3.6]
gtk_entry_buffer_get_max_length [Gtk3 3.6]	gtk_entry_buffer_get_text [Gtk3 3.6]
gtk_entry_buffer_insert_text [Gtk3 3.6]	gtk_entry_buffer_new [Gtk3 3.6]
gtk_entry_buffer_set_max_length [Gtk3 3.6]	gtk_entry_buffer_set_text [Gtk3 3.6]
gtk_entry_completion_complete [Gtk3 3.6]	gtk_entry_completion_compute_prefix [Gtk3 3.6]
gtk_entry_completion_delete_action [Gtk3 3.6]	gtk_entry_completion_get_completion_prefix [Gtk3 3.6]
gtk_entry_completion_get_entry [Gtk3 3.6]	gtk_entry_completion_get_inline_completion [Gtk3 3.6]
gtk_entry_completion_get_inline_selection [Gtk3 3.6]	gtk_entry_completion_get_minimum_key_length [Gtk3 3.6]
gtk_entry_completion_get_model [Gtk3 3.6]	gtk_entry_completion_get_popup_completion [Gtk3 3.6]
gtk_entry_completion_get_popup_set_width [Gtk3 3.6]	gtk_entry_completion_get_popup_single_match [Gtk3 3.6]
gtk_entry_completion_get_text_column [Gtk3 3.6]	gtk_entry_completion_insert_action_markup [Gtk3 3.6]
gtk_entry_completion_insert_action_text [Gtk3 3.6]	gtk_entry_completion_insert_prefix [Gtk3 3.6]
gtk_entry_completion_new [Gtk3 3.6]	gtk_entry_completion_new_with_area [Gtk3 3.6]
gtk_entry_completion_set_inline_completion [Gtk3 3.6]	gtk_entry_completion_set_inline_selection [Gtk3 3.6]
gtk_entry_completion_set_match_function [Gtk3 3.6]	gtk_entry_completion_set_minimum_key_length [Gtk3 3.6]

gtk_entry_completion_set_model [Gtk3 3.6]	gtk_entry_completion_set_popup_completion [Gtk3 3.6]
gtk_entry_completion_set_popup_set_width [Gtk3 3.6]	gtk_entry_completion_set_popup_single_match [Gtk3 3.6]
gtk_entry_completion_set_text_column [Gtk3 3.6]	gtk_entry_get_activates_default [Gtk3 3.6]
gtk_entry_get_alignment [Gtk3 3.6]	gtk_entry_get_attributes [Gtk3 3.6]
gtk_entry_get_buffer [Gtk3 3.6]	gtk_entry_get_completion [Gtk3 3.6]
gtk_entry_get_current_icon_drag_source [Gtk3 3.6]	gtk_entry_get_cursor_hadjustment [Gtk3 3.6]
gtk_entry_get_has_frame [Gtk3 3.6]	gtk_entry_get_icon_activatable [Gtk3 3.6]
gtk_entry_get_icon_area [Gtk3 3.6]	gtk_entry_get_icon_at_pos [Gtk3 3.6]
gtk_entry_get_icon_gicon [Gtk3 3.6]	gtk_entry_get_icon_name [Gtk3 3.6]
gtk_entry_get_icon_pixbuf [Gtk3 3.6]	gtk_entry_get_icon_sensitive [Gtk3 3.6]
gtk_entry_get_icon_stock [Gtk3 3.6]	gtk_entry_get_icon_storage_type [Gtk3 3.6]
gtk_entry_get_icon_tooltip_markup [Gtk3 3.6]	gtk_entry_get_icon_tooltip_text [Gtk3 3.6]
gtk_entry_get_inner_border [Gtk3 3.6]	gtk_entry_get_input_hints [Gtk3 3.6]
gtk_entry_get_input_purpose [Gtk3 3.6]	gtk_entry_get_invisible_char [Gtk3 3.6]
gtk_entry_get_layout [Gtk3 3.6]	gtk_entry_get_layout_offsets [Gtk3 3.6]
gtk_entry_get_max_length [Gtk3 3.6]	gtk_entry_get_overwrite_mode [Gtk3 3.6]
gtk_entry_get_placeholder_text [Gtk3 3.6]	gtk_entry_get_progress_fraction [Gtk3 3.6]
gtk_entry_get_progress_pulse_step [Gtk3 3.6]	gtk_entry_get_text [Gtk3 3.6]
gtk_entry_get_text_area [Gtk3 3.6]	gtk_entry_get_text_length [Gtk3 3.6]
gtk_entry_get_visibility [Gtk3 3.6]	gtk_entry_get_width_chars [Gtk3 3.6]
gtk_entry_im_context_filter_keypress [Gtk3 3.6]	gtk_entry_layout_index_to_text_index [Gtk3 3.6]
gtk_entry_new [Gtk3 3.6]	gtk_entry_new_with_buffer [Gtk3 3.6]
gtk_entry_progress_pulse [Gtk3 3.6]	gtk_entry_reset_im_context [Gtk3 3.6]

gtk_entry_set_activates_default [Gtk3 3.6]	gtk_entry_set_alignment [Gtk3 3.6]
gtk_entry_set_attributes [Gtk3 3.6]	gtk_entry_set_buffer [Gtk3 3.6]
gtk_entry_set_completion [Gtk3 3.6]	gtk_entry_set_cursor_hadjustment [Gtk3 3.6]
gtk_entry_set_has_frame [Gtk3 3.6]	gtk_entry_set_icon_activatable [Gtk3 3.6]
gtk_entry_set_icon_drag_source [Gtk3 3.6]	gtk_entry_set_icon_from_gicon [Gtk3 3.6]
gtk_entry_set_icon_from_icon_name [Gtk3 3.6]	gtk_entry_set_icon_from_pixbuf [Gtk3 3.6]
gtk_entry_set_icon_from_stock [Gtk3 3.6]	gtk_entry_set_icon_sensitive [Gtk3 3.6]
gtk_entry_set_icon_tooltip_markup [Gtk3 3.6]	gtk_entry_set_icon_tooltip_text [Gtk3 3.6]
gtk_entry_set_inner_border [Gtk3 3.6]	gtk_entry_set_input_hints [Gtk3 3.6]
gtk_entry_set_input_purpose [Gtk3 3.6]	gtk_entry_set_invisible_char [Gtk3 3.6]
gtk_entry_set_max_length [Gtk3 3.6]	gtk_entry_set_overwrite_mode [Gtk3 3.6]
gtk_entry_set_placeholder_text [Gtk3 3.6]	gtk_entry_set_progress_fraction [Gtk3 3.6]
gtk_entry_set_progress_pulse_step [Gtk3 3.6]	gtk_entry_set_text [Gtk3 3.6]
gtk_entry_set_visibility [Gtk3 3.6]	gtk_entry_set_width_chars [Gtk3 3.6]
gtk_entry_text_index_to_layout_index [Gtk3 3.6]	gtk_entry_unset_invisible_char [Gtk3 3.6]
gtk_enumerate_printers [Gtk3 3.6]	gtk_event_box_get_above_child [Gtk3 3.6]
gtk_event_box_get_visible_window [Gtk3 3.6]	gtk_event_box_new [Gtk3 3.6]
gtk_event_box_set_above_child [Gtk3 3.6]	gtk_event_box_set_visible_window [Gtk3 3.6]
gtk_events_pending [Gtk3 3.6]	gtk_expander_get_expanded [Gtk3 3.6]
gtk_expander_get_label [Gtk3 3.6]	gtk_expander_get_label_fill [Gtk3 3.6]
gtk_expander_get_label_widget [Gtk3 3.6]	gtk_expander_get_resize_toplevel [Gtk3 3.6]
gtk_expander_get_spacing [Gtk3 3.6]	gtk_expander_get_use_markup [Gtk3 3.6]

gtk_expander_get_use_underline [Gtk3 3.6]	gtk_expander_new [Gtk3 3.6]
gtk_expander_new_with_mnemonic [Gtk3 3.6]	gtk_expander_set_expanded [Gtk3 3.6]
gtk_expander_set_label [Gtk3 3.6]	gtk_expander_set_label_fill [Gtk3 3.6]
gtk_expander_set_label_widget [Gtk3 3.6]	gtk_expander_set_resize_toplevel [Gtk3 3.6]
gtk_expander_set_spacing [Gtk3 3.6]	gtk_expander_set_use_markup [Gtk3 3.6]
gtk_expander_set_use_underline [Gtk3 3.6]	gtk_false [Gtk3 3.6]
gtk_file_chooser_add_filter [Gtk3 3.6]	gtk_file_chooser_add_shortcut_folder [Gtk3 3.6]
gtk_file_chooser_add_shortcut_folder_uri [Gtk3 3.6]	gtk_file_chooser_button_get_focus_on_click [Gtk3 3.6]
gtk_file_chooser_button_get_title [Gtk3 3.6]	gtk_file_chooser_button_get_width_chars [Gtk3 3.6]
gtk_file_chooser_button_new [Gtk3 3.6]	gtk_file_chooser_button_new_with_dialog [Gtk3 3.6]
gtk_file_chooser_button_set_focus_on_click [Gtk3 3.6]	gtk_file_chooser_button_set_title [Gtk3 3.6]
gtk_file_chooser_button_set_width_chars [Gtk3 3.6]	gtk_file_chooser_dialog_new [Gtk3 3.6]
gtk_file_chooser_get_action [Gtk3 3.6]	gtk_file_chooser_get_create_folders [Gtk3 3.6]
gtk_file_chooser_get_current_folder [Gtk3 3.6]	gtk_file_chooser_get_current_folder_file [Gtk3 3.6]
gtk_file_chooser_get_current_folder_uri [Gtk3 3.6]	gtk_file_chooser_get_do_overwrite_confirmation [Gtk3 3.6]
gtk_file_chooser_get_extra_widget [Gtk3 3.6]	gtk_file_chooser_get_file [Gtk3 3.6]
gtk_file_chooser_get_filename [Gtk3 3.6]	gtk_file_chooser_get_filenames [Gtk3 3.6]
gtk_file_chooser_get_files [Gtk3 3.6]	gtk_file_chooser_get_filter [Gtk3 3.6]
gtk_file_chooser_get_local_only [Gtk3 3.6]	gtk_file_chooser_get_preview_file [Gtk3 3.6]
gtk_file_chooser_get_preview_filename [Gtk3 3.6]	gtk_file_chooser_get_preview_uri [Gtk3 3.6]
gtk_file_chooser_get_preview_widget [Gtk3 3.6]	gtk_file_chooser_get_preview_widget_active [Gtk3 3.6]

gtk_file_chooser_get_select_multiple [Gtk3 3.6]	gtk_file_chooser_get_show_hidden [Gtk3 3.6]
gtk_file_chooser_get_uri [Gtk3 3.6]	gtk_file_chooser_get_uris [Gtk3 3.6]
gtk_file_chooser_get_use_preview_label [Gtk3 3.6]	gtk_file_chooser_list_filters [Gtk3 3.6]
gtk_file_chooser_list_shortcut_folders [Gtk3 3.6]	gtk_file_chooser_list_shortcut_folders [Gtk3 3.6]
gtk_file_chooser_remove_filter [Gtk3 3.6]	gtk_file_chooser_remove_shortcut_folder [Gtk3 3.6]
gtk_file_chooser_remove_shortcut_folder_uri [Gtk3 3.6]	gtk_file_chooser_select_all [Gtk3 3.6]
gtk_file_chooser_select_file [Gtk3 3.6]	gtk_file_chooser_select_filename [Gtk3 3.6]
gtk_file_chooser_select_uri [Gtk3 3.6]	gtk_file_chooser_set_action [Gtk3 3.6]
gtk_file_chooser_set_create_folders [Gtk3 3.6]	gtk_file_chooser_set_current_folder [Gtk3 3.6]
gtk_file_chooser_set_current_folder_file [Gtk3 3.6]	gtk_file_chooser_set_current_folder_uri [Gtk3 3.6]
gtk_file_chooser_set_current_name [Gtk3 3.6]	gtk_file_chooser_set_do_overwrite_confirmation [Gtk3 3.6]
gtk_file_chooser_set_extra_widget [Gtk3 3.6]	gtk_file_chooser_set_file [Gtk3 3.6]
gtk_file_chooser_set_filename [Gtk3 3.6]	gtk_file_chooser_set_filter [Gtk3 3.6]
gtk_file_chooser_set_local_only [Gtk3 3.6]	gtk_file_chooser_set_preview_widget [Gtk3 3.6]
gtk_file_chooser_set_preview_widget_active [Gtk3 3.6]	gtk_file_chooser_set_select_multiple [Gtk3 3.6]
gtk_file_chooser_set_show_hidden [Gtk3 3.6]	gtk_file_chooser_set_uri [Gtk3 3.6]
gtk_file_chooser_set_use_preview_label [Gtk3 3.6]	gtk_file_chooser_unselect_all [Gtk3 3.6]
gtk_file_chooser_unselect_file [Gtk3 3.6]	gtk_file_chooser_unselect_filename [Gtk3 3.6]
gtk_file_chooser_unselect_uri [Gtk3 3.6]	gtk_file_chooser_widget_new [Gtk3 3.6]
gtk_file_filter_add_custom [Gtk3 3.6]	gtk_file_filter_add_mime_type [Gtk3 3.6]
gtk_file_filter_add_pattern [Gtk3 3.6]	gtk_file_filter_add_pixbuf_formats [Gtk3 3.6]
gtk_file_filter_filter [Gtk3 3.6]	gtk_file_filter_get_name [Gtk3 3.6]

gtk_file_filter_get_needed [Gtk3 3.6]	gtk_file_filter_new [Gtk3 3.6]
gtk_file_filter_set_name [Gtk3 3.6]	gtk_fixed_move [Gtk3 3.6]
gtk_fixed_new [Gtk3 3.6]	gtk_fixed_put [Gtk3 3.6]
gtk_font_button_get_font_name [Gtk3 3.6]	gtk_font_button_get_show_size [Gtk3 3.6]
gtk_font_button_get_show_style [Gtk3 3.6]	gtk_font_button_get_title [Gtk3 3.6]
gtk_font_button_get_use_font [Gtk3 3.6]	gtk_font_button_get_use_size [Gtk3 3.6]
gtk_font_button_new [Gtk3 3.6]	gtk_font_button_new_with_font [Gtk3 3.6]
gtk_font_button_set_font_name [Gtk3 3.6]	gtk_font_button_set_show_size [Gtk3 3.6]
gtk_font_button_set_show_style [Gtk3 3.6]	gtk_font_button_set_title [Gtk3 3.6]
gtk_font_button_set_use_font [Gtk3 3.6]	gtk_font_button_set_use_size [Gtk3 3.6]
gtk_font_chooser_dialog_new [Gtk3 3.6]	gtk_font_chooser_get_font [Gtk3 3.6]
gtk_font_chooser_get_font_desc [Gtk3 3.6]	gtk_font_chooser_get_font_face [Gtk3 3.6]
gtk_font_chooser_get_font_family [Gtk3 3.6]	gtk_font_chooser_get_font_size [Gtk3 3.6]
gtk_font_chooser_get_preview_text [Gtk3 3.6]	gtk_font_chooser_get_show_preview_entry [Gtk3 3.6]
gtk_font_chooser_set_filter_func [Gtk3 3.6]	gtk_font_chooser_set_font [Gtk3 3.6]
gtk_font_chooser_set_font_desc [Gtk3 3.6]	gtk_font_chooser_set_preview_text [Gtk3 3.6]
gtk_font_chooser_set_show_preview_entry [Gtk3 3.6]	gtk_font_chooser_widget_new [Gtk3 3.6]
gtk_frame_get_label [Gtk3 3.6]	gtk_frame_get_label_align [Gtk3 3.6]
gtk_frame_get_label_widget [Gtk3 3.6]	gtk_frame_get_shadow_type [Gtk3 3.6]
gtk_frame_new [Gtk3 3.6]	gtk_frame_set_label [Gtk3 3.6]
gtk_frame_set_label_align [Gtk3 3.6]	gtk_frame_set_label_widget [Gtk3 3.6]
gtk_frame_set_shadow_type [Gtk3 3.6]	gtk_get_binary_age [Gtk3 3.6]
gtk_get_current_event [Gtk3 3.6]	gtk_get_current_event_device [Gtk3 3.6]

gtk_get_current_event_state [Gtk3 3.6]	gtk_get_current_event_time [Gtk3 3.6]
gtk_get_default_language [Gtk3 3.6]	gtk_get_event_widget [Gtk3 3.6]
gtk_get_interface_age [Gtk3 3.6]	gtk_get_major_version [Gtk3 3.6]
gtk_get_micro_version [Gtk3 3.6]	gtk_get_minor_version [Gtk3 3.6]
gtk_get_option_group [Gtk3 3.6]	gtk_grab_add [Gtk3 3.6]
gtk_grab_get_current [Gtk3 3.6]	gtk_grab_remove [Gtk3 3.6]
gtk_gradient_add_color_stop [Gtk3 3.6]	gtk_gradient_new_linear [Gtk3 3.6]
gtk_gradient_new_radial [Gtk3 3.6]	gtk_gradient_ref [Gtk3 3.6]
gtk_gradient_resolve [Gtk3 3.6]	gtk_gradient_resolve_for_context [Gtk3 3.6]
gtk_gradient_to_string [Gtk3 3.6]	gtk_gradient_unref [Gtk3 3.6]
gtk_grid_attach [Gtk3 3.6]	gtk_grid_attach_next_to [Gtk3 3.6]
gtk_grid_get_child_at [Gtk3 3.6]	gtk_grid_get_column_homogeneous [Gtk3 3.6]
gtk_grid_get_column_spacing [Gtk3 3.6]	gtk_grid_get_row_homogeneous [Gtk3 3.6]
gtk_grid_get_row_spacing [Gtk3 3.6]	gtk_grid_insert_column [Gtk3 3.6]
gtk_grid_insert_next_to [Gtk3 3.6]	gtk_grid_insert_row [Gtk3 3.6]
gtk_grid_new [Gtk3 3.6]	gtk_grid_set_column_homogeneous [Gtk3 3.6]
gtk_grid_set_column_spacing [Gtk3 3.6]	gtk_grid_set_row_homogeneous [Gtk3 3.6]
gtk_grid_set_row_spacing [Gtk3 3.6]	gtk_hsv_to_rgb [Gtk3 3.6]
gtk_icon_factory_add [Gtk3 3.6]	gtk_icon_factory_add_default [Gtk3 3.6]
gtk_icon_factory_lookup [Gtk3 3.6]	gtk_icon_factory_lookup_default [Gtk3 3.6]
gtk_icon_factory_new [Gtk3 3.6]	gtk_icon_factory_remove_default [Gtk3 3.6]
gtk_icon_info_copy [Gtk3 3.6]	gtk_icon_info_free [Gtk3 3.6]
gtk_icon_info_get_attach_points [Gtk3 3.6]	gtk_icon_info_get_base_size [Gtk3 3.6]
gtk_icon_info_get_builtin_pixbuf [Gtk3 3.6]	gtk_icon_info_get_display_name [Gtk3 3.6]
gtk_icon_info_get_embedded_rect [Gtk3 3.6]	gtk_icon_info_get_filename [Gtk3 3.6]

gtk_icon_info_load_icon [Gtk3 3.6]	gtk_icon_info_load_symbolic [Gtk3 3.6]
gtk_icon_info_load_symbolic_for_context [Gtk3 3.6]	gtk_icon_info_load_symbolic_for_style [Gtk3 3.6]
gtk_icon_info_new_for_pixbuf [Gtk3 3.6]	gtk_icon_info_set_raw_coordinates [Gtk3 3.6]
gtk_icon_set_add_source [Gtk3 3.6]	gtk_icon_set_copy [Gtk3 3.6]
gtk_icon_set_get_sizes [Gtk3 3.6]	gtk_icon_set_new [Gtk3 3.6]
gtk_icon_set_new_from_pixbuf [Gtk3 3.6]	gtk_icon_set_ref [Gtk3 3.6]
gtk_icon_set_render_icon [Gtk3 3.6]	gtk_icon_set_render_icon_pixbuf [Gtk3 3.6]
gtk_icon_set_unref [Gtk3 3.6]	gtk_icon_size_from_name [Gtk3 3.6]
gtk_icon_size_get_name [Gtk3 3.6]	gtk_icon_size_lookup [Gtk3 3.6]
gtk_icon_size_lookup_for_settings [Gtk3 3.6]	gtk_icon_size_register [Gtk3 3.6]
gtk_icon_size_register_alias [Gtk3 3.6]	gtk_icon_source_copy [Gtk3 3.6]
gtk_icon_source_free [Gtk3 3.6]	gtk_icon_source_get_direction [Gtk3 3.6]
gtk_icon_source_get_direction_wildcarded [Gtk3 3.6]	gtk_icon_source_get_filename [Gtk3 3.6]
gtk_icon_source_get_icon_name [Gtk3 3.6]	gtk_icon_source_get_pixbuf [Gtk3 3.6]
gtk_icon_source_get_size [Gtk3 3.6]	gtk_icon_source_get_size_wildcarded [Gtk3 3.6]
gtk_icon_source_get_state [Gtk3 3.6]	gtk_icon_source_get_state_wildcarded [Gtk3 3.6]
gtk_icon_source_new [Gtk3 3.6]	gtk_icon_source_set_direction [Gtk3 3.6]
gtk_icon_source_set_direction_wildcarded [Gtk3 3.6]	gtk_icon_source_set_filename [Gtk3 3.6]
gtk_icon_source_set_icon_name [Gtk3 3.6]	gtk_icon_source_set_pixbuf [Gtk3 3.6]
gtk_icon_source_set_size [Gtk3 3.6]	gtk_icon_source_set_size_wildcarded [Gtk3 3.6]
gtk_icon_source_set_state [Gtk3 3.6]	gtk_icon_source_set_state_wildcarded [Gtk3 3.6]
gtk_icon_theme_add_builtin_icon [Gtk3 3.6]	gtk_icon_theme_append_search_path [Gtk3 3.6]

gtk_icon_theme_choose_icon [Gtk3 3.6]	gtk_icon_theme_get_default [Gtk3 3.6]
gtk_icon_theme_get_example_icon_name [Gtk3 3.6]	gtk_icon_theme_get_for_screen [Gtk3 3.6]
gtk_icon_theme_get_icon_sizes [Gtk3 3.6]	gtk_icon_theme_get_search_path [Gtk3 3.6]
gtk_icon_theme_has_icon [Gtk3 3.6]	gtk_icon_theme_list_contexts [Gtk3 3.6]
gtk_icon_theme_list_icons [Gtk3 3.6]	gtk_icon_theme_load_icon [Gtk3 3.6]
gtk_icon_theme_lookup_by_gicon [Gtk3 3.6]	gtk_icon_theme_lookup_icon [Gtk3 3.6]
gtk_icon_theme_new [Gtk3 3.6]	gtk_icon_theme_prepend_search_path [Gtk3 3.6]
gtk_icon_theme_rescan_if_needed [Gtk3 3.6]	gtk_icon_theme_set_custom_theme [Gtk3 3.6]
gtk_icon_theme_set_screen [Gtk3 3.6]	gtk_icon_theme_set_search_path [Gtk3 3.6]
gtk_icon_view_convert_widget_to_bin_window_coords [Gtk3 3.6]	gtk_icon_view_create_drag_icon [Gtk3 3.6]
gtk_icon_view_enable_model_drag_dest [Gtk3 3.6]	gtk_icon_view_enable_model_drag_source [Gtk3 3.6]
gtk_icon_view_get_cell_rect [Gtk3 3.6]	gtk_icon_view_get_column_spacing [Gtk3 3.6]
gtk_icon_view_get_columns [Gtk3 3.6]	gtk_icon_view_get_cursor [Gtk3 3.6]
gtk_icon_view_get_dest_item_at_pos [Gtk3 3.6]	gtk_icon_view_get_drag_dest_item [Gtk3 3.6]
gtk_icon_view_get_item_at_pos [Gtk3 3.6]	gtk_icon_view_get_item_column [Gtk3 3.6]
gtk_icon_view_get_item_orientation [Gtk3 3.6]	gtk_icon_view_get_item_padding [Gtk3 3.6]
gtk_icon_view_get_item_row [Gtk3 3.6]	gtk_icon_view_get_item_width [Gtk3 3.6]
gtk_icon_view_get_margin [Gtk3 3.6]	gtk_icon_view_get_markup_column [Gtk3 3.6]
gtk_icon_view_get_model [Gtk3 3.6]	gtk_icon_view_get_path_at_pos [Gtk3 3.6]
gtk_icon_view_get_pixbuf_column [Gtk3 3.6]	gtk_icon_view_get_reorderable [Gtk3 3.6]
gtk_icon_view_get_row_spacing [Gtk3 3.6]	gtk_icon_view_get_selected_items [Gtk3 3.6]

gtk_icon_view_get_selection_mode [Gtk3 3.6]	gtk_icon_view_get_spacing [Gtk3 3.6]
gtk_icon_view_get_text_column [Gtk3 3.6]	gtk_icon_view_get_tooltip_column [Gtk3 3.6]
gtk_icon_view_get_tooltip_context [Gtk3 3.6]	gtk_icon_view_get_visible_range [Gtk3 3.6]
gtk_icon_view_item_activated [Gtk3 3.6]	gtk_icon_view_new [Gtk3 3.6]
gtk_icon_view_new_with_area [Gtk3 3.6]	gtk_icon_view_new_with_model [Gtk3 3.6]
gtk_icon_view_path_is_selected [Gtk3 3.6]	gtk_icon_view_scroll_to_path [Gtk3 3.6]
gtk_icon_view_select_all [Gtk3 3.6]	gtk_icon_view_select_path [Gtk3 3.6]
gtk_icon_view_selected_foreach [Gtk3 3.6]	gtk_icon_view_set_column_spacing [Gtk3 3.6]
gtk_icon_view_set_columns [Gtk3 3.6]	gtk_icon_view_set_cursor [Gtk3 3.6]
gtk_icon_view_set_drag_dest_item [Gtk3 3.6]	gtk_icon_view_set_item_orientation [Gtk3 3.6]
gtk_icon_view_set_item_padding [Gtk3 3.6]	gtk_icon_view_set_item_width [Gtk3 3.6]
gtk_icon_view_set_margin [Gtk3 3.6]	gtk_icon_view_set_markup_column [Gtk3 3.6]
gtk_icon_view_set_model [Gtk3 3.6]	gtk_icon_view_set_pixbuf_column [Gtk3 3.6]
gtk_icon_view_set_reorderable [Gtk3 3.6]	gtk_icon_view_set_row_spacing [Gtk3 3.6]
gtk_icon_view_set_selection_mode [Gtk3 3.6]	gtk_icon_view_set_spacing [Gtk3 3.6]
gtk_icon_view_set_text_column [Gtk3 3.6]	gtk_icon_view_set_tooltip_cell [Gtk3 3.6]
gtk_icon_view_set_tooltip_column [Gtk3 3.6]	gtk_icon_view_set_tooltip_item [Gtk3 3.6]
gtk_icon_view_unselect_all [Gtk3 3.6]	gtk_icon_view_unselect_path [Gtk3 3.6]
gtk_icon_view_unset_model_drag_dest [Gtk3 3.6]	gtk_icon_view_unset_model_drag_source [Gtk3 3.6]
gtk_im_context_delete_surrounding [Gtk3 3.6]	gtk_im_context_filter_keypress [Gtk3 3.6]
gtk_im_context_focus_in [Gtk3 3.6]	gtk_im_context_focus_out [Gtk3 3.6]
gtk_im_context_get_preedit_string [Gtk3 3.6]	gtk_im_context_get_surrounding [Gtk3 3.6]

gtk_im_context_reset [Gtk3 3.6]	gtk_im_context_set_client_window [Gtk3 3.6]
gtk_im_context_set_cursor_location [Gtk3 3.6]	gtk_im_context_set_surrounding [Gtk3 3.6]
gtk_im_context_set_use_preedit [Gtk3 3.6]	gtk_im_context_simple_add_table [Gtk3 3.6]
gtk_im_context_simple_new [Gtk3 3.6]	gtk_im_multicontext_append_menus [Gtk3 3.6]
gtk_im_multicontext_get_context_id [Gtk3 3.6]	gtk_im_multicontext_new [Gtk3 3.6]
gtk_im_multicontext_set_context_id [Gtk3 3.6]	gtk_image_clear [Gtk3 3.6]
gtk_image_get_animation [Gtk3 3.6]	gtk_image_get_gicon [Gtk3 3.6]
gtk_image_get_icon_name [Gtk3 3.6]	gtk_image_get_icon_set [Gtk3 3.6]
gtk_image_get_pixbuf [Gtk3 3.6]	gtk_image_get_pixel_size [Gtk3 3.6]
gtk_image_get_stock [Gtk3 3.6]	gtk_image_get_storage_type [Gtk3 3.6]
gtk_image_menu_item_get_always_show_image [Gtk3 3.6]	gtk_image_menu_item_get_image [Gtk3 3.6]
gtk_image_menu_item_get_use_stock [Gtk3 3.6]	gtk_image_menu_item_new [Gtk3 3.6]
gtk_image_menu_item_new_from_stock [Gtk3 3.6]	gtk_image_menu_item_new_with_label [Gtk3 3.6]
gtk_image_menu_item_new_with_mnemonic [Gtk3 3.6]	gtk_image_menu_item_set_accel_group [Gtk3 3.6]
gtk_image_menu_item_set_always_show_image [Gtk3 3.6]	gtk_image_menu_item_set_image [Gtk3 3.6]
gtk_image_menu_item_set_use_stock [Gtk3 3.6]	gtk_image_new [Gtk3 3.6]
gtk_image_new_from_animation [Gtk3 3.6]	gtk_image_new_from_file [Gtk3 3.6]
gtk_image_new_from_gicon [Gtk3 3.6]	gtk_image_new_from_icon_name [Gtk3 3.6]
gtk_image_new_from_icon_set [Gtk3 3.6]	gtk_image_new_from_pixbuf [Gtk3 3.6]
gtk_image_new_from_resource [Gtk3 3.6]	gtk_image_new_from_stock [Gtk3 3.6]
gtk_image_set_from_animation [Gtk3 3.6]	gtk_image_set_from_file [Gtk3 3.6]
gtk_image_set_from_gicon [Gtk3 3.6]	gtk_image_set_from_icon_name [Gtk3 3.6]

gtk_image_set_from_icon_set [Gtk3 3.6]	gtk_image_set_from_pixbuf [Gtk3 3.6]
gtk_image_set_from_resource [Gtk3 3.6]	gtk_image_set_from_stock [Gtk3 3.6]
gtk_image_set_pixel_size [Gtk3 3.6]	gtk_info_bar_add_action_widget [Gtk3 3.6]
gtk_info_bar_add_button [Gtk3 3.6]	gtk_info_bar_add_buttons [Gtk3 3.6]
gtk_info_bar_get_action_area [Gtk3 3.6]	gtk_info_bar_get_content_area [Gtk3 3.6]
gtk_info_bar_get_message_type [Gtk3 3.6]	gtk_info_bar_new [Gtk3 3.6]
gtk_info_bar_new_with_buttons [Gtk3 3.6]	gtk_info_bar_response [Gtk3 3.6]
gtk_info_bar_set_default_response [Gtk3 3.6]	gtk_info_bar_set_message_type [Gtk3 3.6]
gtk_info_bar_set_response_sensitive [Gtk3 3.6]	gtk_init [Gtk3 3.6]
gtk_init_check [Gtk3 3.6]	gtk_init_with_args [Gtk3 3.6]
gtk_invisible_get_screen [Gtk3 3.6]	gtk_invisible_new [Gtk3 3.6]
gtk_invisible_new_for_screen [Gtk3 3.6]	gtk_invisible_set_screen [Gtk3 3.6]
gtk_key_snooper_install [Gtk3 3.6]	gtk_key_snooper_remove [Gtk3 3.6]
gtk_label_get_angle [Gtk3 3.6]	gtk_label_get_attributes [Gtk3 3.6]
gtk_label_get_current_uri [Gtk3 3.6]	gtk_label_get_ellipsize [Gtk3 3.6]
gtk_label_get_justify [Gtk3 3.6]	gtk_label_get_label [Gtk3 3.6]
gtk_label_get_layout [Gtk3 3.6]	gtk_label_get_layout_offsets [Gtk3 3.6]
gtk_label_get_line_wrap [Gtk3 3.6]	gtk_label_get_line_wrap_mode [Gtk3 3.6]
gtk_label_get_max_width_chars [Gtk3 3.6]	gtk_label_get_mnemonic_keyval [Gtk3 3.6]
gtk_label_get_mnemonic_widget [Gtk3 3.6]	gtk_label_get_selectable [Gtk3 3.6]
gtk_label_get_selection_bounds [Gtk3 3.6]	gtk_label_get_single_line_mode [Gtk3 3.6]
gtk_label_get_text [Gtk3 3.6]	gtk_label_get_track_visited_links [Gtk3 3.6]
gtk_label_get_use_markup [Gtk3 3.6]	gtk_label_get_use_underline [Gtk3 3.6]
gtk_label_get_width_chars [Gtk3 3.6]	gtk_label_new [Gtk3 3.6]

gtk_label_new_with_mnemonic [Gtk3 3.6]	gtk_label_select_region [Gtk3 3.6]
gtk_label_set_angle [Gtk3 3.6]	gtk_label_set_attributes [Gtk3 3.6]
gtk_label_set_ellipsize [Gtk3 3.6]	gtk_label_set_justify [Gtk3 3.6]
gtk_label_set_label [Gtk3 3.6]	gtk_label_set_line_wrap [Gtk3 3.6]
gtk_label_set_line_wrap_mode [Gtk3 3.6]	gtk_label_set_markup [Gtk3 3.6]
gtk_label_set_markup_with_mnemonic [Gtk3 3.6]	gtk_label_set_max_width_chars [Gtk3 3.6]
gtk_label_set_mnemonic_widget [Gtk3 3.6]	gtk_label_set_pattern [Gtk3 3.6]
gtk_label_set_selectable [Gtk3 3.6]	gtk_label_set_single_line_mode [Gtk3 3.6]
gtk_label_set_text [Gtk3 3.6]	gtk_label_set_text_with_mnemonic [Gtk3 3.6]
gtk_label_set_track_visited_links [Gtk3 3.6]	gtk_label_set_use_markup [Gtk3 3.6]
gtk_label_set_use_underline [Gtk3 3.6]	gtk_label_set_width_chars [Gtk3 3.6]
gtk_layout_get_bin_window [Gtk3 3.6]	gtk_layout_get_hadjustment [Gtk3 3.6]
gtk_layout_get_size [Gtk3 3.6]	gtk_layout_get_vadjustment [Gtk3 3.6]
gtk_layout_move [Gtk3 3.6]	gtk_layout_new [Gtk3 3.6]
gtk_layout_put [Gtk3 3.6]	gtk_layout_set_hadjustment [Gtk3 3.6]
gtk_layout_set_size [Gtk3 3.6]	gtk_layout_set_vadjustment [Gtk3 3.6]
gtk_level_bar_add_offset_value [Gtk3 3.6]	gtk_level_bar_get_max_value [Gtk3 3.6]
gtk_level_bar_get_min_value [Gtk3 3.6]	gtk_level_bar_get_mode [Gtk3 3.6]
gtk_level_bar_get_offset_value [Gtk3 3.6]	gtk_level_bar_get_value [Gtk3 3.6]
gtk_level_bar_new [Gtk3 3.6]	gtk_level_bar_new_for_interval [Gtk3 3.6]
gtk_level_bar_remove_offset_value [Gtk3 3.6]	gtk_level_bar_set_max_value [Gtk3 3.6]
gtk_level_bar_set_min_value [Gtk3 3.6]	gtk_level_bar_set_mode [Gtk3 3.6]
gtk_level_bar_set_value [Gtk3 3.6]	gtk_link_button_get_uri [Gtk3 3.6]

gtk_link_button_get_visited [Gtk3 3.6]	gtk_link_button_new [Gtk3 3.6]
gtk_link_button_new_with_label [Gtk3 3.6]	gtk_link_button_set_uri [Gtk3 3.6]
gtk_link_button_set_visited [Gtk3 3.6]	gtk_list_store_append [Gtk3 3.6]
gtk_list_store_clear [Gtk3 3.6]	gtk_list_store_insert [Gtk3 3.6]
gtk_list_store_insert_after [Gtk3 3.6]	gtk_list_store_insert_before [Gtk3 3.6]
gtk_list_store_insert_with_values [Gtk3 3.6]	gtk_list_store_insert_with_valuesv [Gtk3 3.6]
gtk_list_store_iter_is_valid [Gtk3 3.6]	gtk_list_store_move_after [Gtk3 3.6]
gtk_list_store_move_before [Gtk3 3.6]	gtk_list_store_new [Gtk3 3.6]
gtk_list_store_newv [Gtk3 3.6]	gtk_list_store_prepend [Gtk3 3.6]
gtk_list_store_remove [Gtk3 3.6]	gtk_list_store_reorder [Gtk3 3.6]
gtk_list_store_set [Gtk3 3.6]	gtk_list_store_set_column_types [Gtk3 3.6]
gtk_list_store_set_valist [Gtk3 3.6]	gtk_list_store_set_value [Gtk3 3.6]
gtk_list_store_set_valuesv [Gtk3 3.6]	gtk_list_store_swap [Gtk3 3.6]
gtk_lock_button_get_permission [Gtk3 3.6]	gtk_lock_button_new [Gtk3 3.6]
gtk_lock_button_set_permission [Gtk3 3.6]	gtk_main [Gtk3 3.6]
gtk_main_do_event [Gtk3 3.6]	gtk_main_iteration [Gtk3 3.6]
gtk_main_iteration_do [Gtk3 3.6]	gtk_main_level [Gtk3 3.6]
gtk_main_quit [Gtk3 3.6]	gtk_menu_attach [Gtk3 3.6]
gtk_menu_attach_to_widget [Gtk3 3.6]	gtk_menu_bar_get_child_pack_direction [Gtk3 3.6]
gtk_menu_bar_get_pack_direction [Gtk3 3.6]	gtk_menu_bar_new [Gtk3 3.6]
gtk_menu_bar_new_from_model [Gtk3 3.6]	gtk_menu_bar_set_child_pack_direction [Gtk3 3.6]
gtk_menu_bar_set_pack_direction [Gtk3 3.6]	gtk_menu_button_get_align_widget [Gtk3 3.6]
gtk_menu_button_get_direction [Gtk3 3.6]	gtk_menu_button_get_menu_model [Gtk3 3.6]
gtk_menu_button_get_popup [Gtk3 3.6]	gtk_menu_button_new [Gtk3 3.6]

gtk_menu_button_set_align_widget [Gtk3 3.6]	gtk_menu_button_set_direction [Gtk3 3.6]
gtk_menu_button_set_menu_model [Gtk3 3.6]	gtk_menu_button_set_popup [Gtk3 3.6]
gtk_menu_detach [Gtk3 3.6]	gtk_menu_get_accel_group [Gtk3 3.6]
gtk_menu_get_accel_path [Gtk3 3.6]	gtk_menu_get_active [Gtk3 3.6]
gtk_menu_get_attach_widget [Gtk3 3.6]	gtk_menu_get_for_attach_widget [Gtk3 3.6]
gtk_menu_get_monitor [Gtk3 3.6]	gtk_menu_get_reserve_toggle_size [Gtk3 3.6]
gtk_menu_get_tearoff_state [Gtk3 3.6]	gtk_menu_get_title [Gtk3 3.6]
gtk_menu_item_activate [Gtk3 3.6]	gtk_menu_item_deselect [Gtk3 3.6]
gtk_menu_item_get_accel_path [Gtk3 3.6]	gtk_menu_item_get_label [Gtk3 3.6]
gtk_menu_item_get_reserve_indicator [Gtk3 3.6]	gtk_menu_item_get_right_justified [Gtk3 3.6]
gtk_menu_item_get_submenu [Gtk3 3.6]	gtk_menu_item_get_use_underline [Gtk3 3.6]
gtk_menu_item_new [Gtk3 3.6]	gtk_menu_item_new_with_label [Gtk3 3.6]
gtk_menu_item_new_with_mnemonic [Gtk3 3.6]	gtk_menu_item_select [Gtk3 3.6]
gtk_menu_item_set_accel_path [Gtk3 3.6]	gtk_menu_item_set_label [Gtk3 3.6]
gtk_menu_item_set_reserve_indicator [Gtk3 3.6]	gtk_menu_item_set_right_justified [Gtk3 3.6]
gtk_menu_item_set_submenu [Gtk3 3.6]	gtk_menu_item_set_use_underline [Gtk3 3.6]
gtk_menu_item_toggle_size_allocate [Gtk3 3.6]	gtk_menu_item_toggle_size_request [Gtk3 3.6]
gtk_menu_new [Gtk3 3.6]	gtk_menu_new_from_model [Gtk3 3.6]
gtk_menu_popdown [Gtk3 3.6]	gtk_menu_popup [Gtk3 3.6]
gtk_menu_popup_for_device [Gtk3 3.6]	gtk_menu_reorder_child [Gtk3 3.6]
gtk_menu_reposition [Gtk3 3.6]	gtk_menu_set_accel_group [Gtk3 3.6]
gtk_menu_set_accel_path [Gtk3 3.6]	gtk_menu_set_active [Gtk3 3.6]
gtk_menu_set_monitor [Gtk3 3.6]	gtk_menu_set_reserve_toggle_size [Gtk3 3.6]

gtk_menu_set_screen [Gtk3 3.6]	gtk_menu_set_tearoff_state [Gtk3 3.6]
gtk_menu_set_title [Gtk3 3.6]	gtk_menu_shell_activate_item [Gtk3 3.6]
gtk_menu_shell_append [Gtk3 3.6]	gtk_menu_shell_bind_model [Gtk3 3.6]
gtk_menu_shell_cancel [Gtk3 3.6]	gtk_menu_shell_deactivate [Gtk3 3.6]
gtk_menu_shell_deselect [Gtk3 3.6]	gtk_menu_shell_get_parent_shell [Gtk3 3.6]
gtk_menu_shell_get_selected_item [Gtk3 3.6]	gtk_menu_shell_get_take_focus [Gtk3 3.6]
gtk_menu_shell_insert [Gtk3 3.6]	gtk_menu_shell_prepend [Gtk3 3.6]
gtk_menu_shell_select_first [Gtk3 3.6]	gtk_menu_shell_select_item [Gtk3 3.6]
gtk_menu_shell_set_take_focus [Gtk3 3.6]	gtk_menu_tool_button_get_menu [Gtk3 3.6]
gtk_menu_tool_button_new [Gtk3 3.6]	gtk_menu_tool_button_new_from_stock [Gtk3 3.6]
gtk_menu_tool_button_set_arrow_tooltip_markup [Gtk3 3.6]	gtk_menu_tool_button_set_arrow_tooltip_text [Gtk3 3.6]
gtk_menu_tool_button_set_menu [Gtk3 3.6]	gtk_message_dialog_format_secondary_markup [Gtk3 3.6]
gtk_message_dialog_format_secondary_text [Gtk3 3.6]	gtk_message_dialog_get_image [Gtk3 3.6]
gtk_message_dialog_get_message_area [Gtk3 3.6]	gtk_message_dialog_new [Gtk3 3.6]
gtk_message_dialog_new_with_markup [Gtk3 3.6]	gtk_message_dialog_set_image [Gtk3 3.6]
gtk_message_dialog_set_markup [Gtk3 3.6]	gtk_misc_get_alignment [Gtk3 3.6]
gtk_misc_get_padding [Gtk3 3.6]	gtk_misc_set_alignment [Gtk3 3.6]
gtk_misc_set_padding [Gtk3 3.6]	gtk_mount_operation_get_parent [Gtk3 3.6]
gtk_mount_operation_get_screen [Gtk3 3.6]	gtk_mount_operation_is_showing [Gtk3 3.6]
gtk_mount_operation_new [Gtk3 3.6]	gtk_mount_operation_set_parent [Gtk3 3.6]
gtk_mount_operation_set_screen [Gtk3 3.6]	gtk_notebook_append_page [Gtk3 3.6]
gtk_notebook_append_page_menu [Gtk3 3.6]	gtk_notebook_get_action_widget [Gtk3 3.6]

gtk_notebook_get_current_page [Gtk3 3.6]	gtk_notebook_get_group_name [Gtk3 3.6]
gtk_notebook_get_menu_label [Gtk3 3.6]	gtk_notebook_get_menu_label_text [Gtk3 3.6]
gtk_notebook_get_n_pages [Gtk3 3.6]	gtk_notebook_get_nth_page [Gtk3 3.6]
gtk_notebook_get_scrollable [Gtk3 3.6]	gtk_notebook_get_show_border [Gtk3 3.6]
gtk_notebook_get_show_tabs [Gtk3 3.6]	gtk_notebook_get_tab_detachable [Gtk3 3.6]
gtk_notebook_get_tab_hborder [Gtk3 3.6]	gtk_notebook_get_tab_label [Gtk3 3.6]
gtk_notebook_get_tab_label_text [Gtk3 3.6]	gtk_notebook_get_tab_pos [Gtk3 3.6]
gtk_notebook_get_tab_reorderable [Gtk3 3.6]	gtk_notebook_get_tab_vborder [Gtk3 3.6]
gtk_notebook_insert_page [Gtk3 3.6]	gtk_notebook_insert_page_menu [Gtk3 3.6]
gtk_notebook_new [Gtk3 3.6]	gtk_notebook_next_page [Gtk3 3.6]
gtk_notebook_page_num [Gtk3 3.6]	gtk_notebook_popup_disable [Gtk3 3.6]
gtk_notebook_popup_enable [Gtk3 3.6]	gtk_notebook_prepend_page [Gtk3 3.6]
gtk_notebook_prepend_page_menu [Gtk3 3.6]	gtk_notebook_prev_page [Gtk3 3.6]
gtk_notebook_remove_page [Gtk3 3.6]	gtk_notebook_reorder_child [Gtk3 3.6]
gtk_notebook_set_action_widget [Gtk3 3.6]	gtk_notebook_set_current_page [Gtk3 3.6]
gtk_notebook_set_group_name [Gtk3 3.6]	gtk_notebook_set_menu_label [Gtk3 3.6]
gtk_notebook_set_menu_label_text [Gtk3 3.6]	gtk_notebook_set_scrollable [Gtk3 3.6]
gtk_notebook_set_show_border [Gtk3 3.6]	gtk_notebook_set_show_tabs [Gtk3 3.6]
gtk_notebook_set_tab_detachable [Gtk3 3.6]	gtk_notebook_set_tab_label [Gtk3 3.6]
gtk_notebook_set_tab_label_text [Gtk3 3.6]	gtk_notebook_set_tab_pos [Gtk3 3.6]
gtk_notebook_set_tab_reorderable [Gtk3 3.6]	gtk_numerable_icon_get_backgroun d_gicon [Gtk3 3.6]

gtk_numerable_icon_get_background _icon_name [Gtk3 3.6]	gtk_numerable_icon_get_count [Gtk3 3.6]
gtk_numerable_icon_get_label [Gtk3 3.6]	gtk_numerable_icon_get_style_conte xt [Gtk3 3.6]
gtk_numerable_icon_new [Gtk3 3.6]	gtk_numerable_icon_new_with_style _context [Gtk3 3.6]
gtk_numerable_icon_set_background _icon [Gtk3 3.6]	gtk_numerable_icon_set_background _icon_name [Gtk3 3.6]
gtk_numerable_icon_set_count [Gtk3 3.6]	gtk_numerable_icon_set_label [Gtk3 3.6]
gtk_numerable_icon_set_style_conte xt [Gtk3 3.6]	gtk_offscreen_window_get_pixbuf [Gtk3 3.6]
gtk_offscreen_window_get_surface [Gtk3 3.6]	gtk_offscreen_window_new [Gtk3 3.6]
gtk_orientable_get_orientation [Gtk3 3.6]	gtk_orientable_set_orientation [Gtk3 3.6]
gtk_overlay_add_overlay [Gtk3 3.6]	gtk_overlay_new [Gtk3 3.6]
gtk_page_setup_copy [Gtk3 3.6]	gtk_page_setup_get_bottom_margin [Gtk3 3.6]
gtk_page_setup_get_left_margin [Gtk3 3.6]	gtk_page_setup_get_orientation [Gtk3 3.6]
gtk_page_setup_get_page_height [Gtk3 3.6]	gtk_page_setup_get_page_width [Gtk3 3.6]
gtk_page_setup_get_paper_height [Gtk3 3.6]	gtk_page_setup_get_paper_size [Gtk3 3.6]
gtk_page_setup_get_paper_width [Gtk3 3.6]	gtk_page_setup_get_right_margin [Gtk3 3.6]
gtk_page_setup_get_top_margin [Gtk3 3.6]	gtk_page_setup_load_file [Gtk3 3.6]
gtk_page_setup_load_key_file [Gtk3 3.6]	gtk_page_setup_new [Gtk3 3.6]
gtk_page_setup_new_from_file [Gtk3 3.6]	gtk_page_setup_new_from_key_file [Gtk3 3.6]
gtk_page_setup_set_bottom_margin [Gtk3 3.6]	gtk_page_setup_set_left_margin [Gtk3 3.6]
gtk_page_setup_set_orientation [Gtk3 3.6]	gtk_page_setup_set_paper_size [Gtk3 3.6]
gtk_page_setup_set_paper_size_and _default_margins [Gtk3 3.6]	gtk_page_setup_set_right_margin [Gtk3 3.6]
gtk_page_setup_set_top_margin [Gtk3 3.6]	gtk_page_setup_to_file [Gtk3 3.6]

gtk_page_setup_to_key_file [Gtk3 3.6]	gtk_page_setup_unix_dialog_get_page_setup [Gtk3 3.6]
gtk_page_setup_unix_dialog_get_print_settings [Gtk3 3.6]	gtk_page_setup_unix_dialog_new [Gtk3 3.6]
gtk_page_setup_unix_dialog_set_page_setup [Gtk3 3.6]	gtk_page_setup_unix_dialog_set_print_settings [Gtk3 3.6]
gtk_paned_add1 [Gtk3 3.6]	gtk_paned_add2 [Gtk3 3.6]
gtk_paned_get_child1 [Gtk3 3.6]	gtk_paned_get_child2 [Gtk3 3.6]
gtk_paned_get_handle_window [Gtk3 3.6]	gtk_paned_get_position [Gtk3 3.6]
gtk_paned_new [Gtk3 3.6]	gtk_paned_pack1 [Gtk3 3.6]
gtk_paned_pack2 [Gtk3 3.6]	gtk_paned_set_position [Gtk3 3.6]
gtk_paper_size_copy [Gtk3 3.6]	gtk_paper_size_free [Gtk3 3.6]
gtk_paper_size_get_default [Gtk3 3.6]	gtk_paper_size_get_default_bottom_margin [Gtk3 3.6]
gtk_paper_size_get_default_left_margin [Gtk3 3.6]	gtk_paper_size_get_default_right_margin [Gtk3 3.6]
gtk_paper_size_get_default_top_margin [Gtk3 3.6]	gtk_paper_size_get_display_name [Gtk3 3.6]
gtk_paper_size_get_height [Gtk3 3.6]	gtk_paper_size_get_name [Gtk3 3.6]
gtk_paper_size_get_paper_sizes [Gtk3 3.6]	gtk_paper_size_get_ppd_name [Gtk3 3.6]
gtk_paper_size_get_width [Gtk3 3.6]	gtk_paper_size_is_custom [Gtk3 3.6]
gtk_paper_size_is_equal [Gtk3 3.6]	gtk_paper_size_new [Gtk3 3.6]
gtk_paper_size_new_custom [Gtk3 3.6]	gtk_paper_size_new_from_key_file [Gtk3 3.6]
gtk_paper_size_new_from_ppd [Gtk3 3.6]	gtk_paper_size_set_size [Gtk3 3.6]
gtk_paper_size_to_key_file [Gtk3 3.6]	gtk_parse_args [Gtk3 3.6]
gtk_plugin_construct [Gtk3 3.6]	gtk_plugin_construct_for_display [Gtk3 3.6]
gtk_plugin_get_embedded [Gtk3 3.6]	gtk_plugin_get_id [Gtk3 3.6]
gtk_plugin_get_socket_window [Gtk3 3.6]	gtk_plugin_new [Gtk3 3.6]
gtk_plugin_new_for_display [Gtk3 3.6]	gtk_print_context_create_pango_context [Gtk3 3.6]
gtk_print_context_create_pango_layout [Gtk3 3.6]	gtk_print_context_get_cairo_context [Gtk3 3.6]
gtk_print_context_get_dpi_x [Gtk3 3.6]	gtk_print_context_get_dpi_y [Gtk3 3.6]

gtk_print_context_get_hard_margins [Gtk3 3.6]	gtk_print_context_get_height [Gtk3 3.6]
gtk_print_context_get_page_setup [Gtk3 3.6]	gtk_print_context_get_pango_fontmap [Gtk3 3.6]
gtk_print_context_get_width [Gtk3 3.6]	gtk_print_context_set_cairo_context [Gtk3 3.6]
gtk_print_operation_cancel [Gtk3 3.6]	gtk_print_operation_draw_page_finish [Gtk3 3.6]
gtk_print_operation_get_default_page_setup [Gtk3 3.6]	gtk_print_operation_get_embed_page_setup [Gtk3 3.6]
gtk_print_operation_get_error [Gtk3 3.6]	gtk_print_operation_get_has_selection [Gtk3 3.6]
gtk_print_operation_get_n_pages_to_print [Gtk3 3.6]	gtk_print_operation_get_print_settings [Gtk3 3.6]
gtk_print_operation_get_status [Gtk3 3.6]	gtk_print_operation_get_status_string [Gtk3 3.6]
gtk_print_operation_get_support_selection [Gtk3 3.6]	gtk_print_operation_is_finished [Gtk3 3.6]
gtk_print_operation_new [Gtk3 3.6]	gtk_print_operation_preview_end_preview [Gtk3 3.6]
gtk_print_operation_preview_is_selected [Gtk3 3.6]	gtk_print_operation_preview_render_page [Gtk3 3.6]
gtk_print_operation_run [Gtk3 3.6]	gtk_print_operation_set_allow_async [Gtk3 3.6]
gtk_print_operation_set_current_page [Gtk3 3.6]	gtk_print_operation_set_custom_table_label [Gtk3 3.6]
gtk_print_operation_set_default_page_setup [Gtk3 3.6]	gtk_print_operation_set_defer_drawing [Gtk3 3.6]
gtk_print_operation_set_embed_page_setup [Gtk3 3.6]	gtk_print_operation_set_export_filename [Gtk3 3.6]
gtk_print_operation_set_has_selection [Gtk3 3.6]	gtk_print_operation_set_job_name [Gtk3 3.6]
gtk_print_operation_set_n_pages [Gtk3 3.6]	gtk_print_operation_set_print_settings [Gtk3 3.6]
gtk_print_operation_set_show_progress [Gtk3 3.6]	gtk_print_operation_set_support_selection [Gtk3 3.6]
gtk_print_operation_set_track_print_status [Gtk3 3.6]	gtk_print_operation_set_unit [Gtk3 3.6]
gtk_print_operation_set_use_full_page [Gtk3 3.6]	gtk_print_run_page_setup_dialog [Gtk3 3.6]
gtk_print_run_page_setup_dialog_async [Gtk3 3.6]	gtk_print_settings_copy [Gtk3 3.6]

gtk_print_settings_foreach [Gtk3 3.6]	gtk_print_settings_get [Gtk3 3.6]
gtk_print_settings_get_bool [Gtk3 3.6]	gtk_print_settings_get_collate [Gtk3 3.6]
gtk_print_settings_get_default_source [Gtk3 3.6]	gtk_print_settings_get_dither [Gtk3 3.6]
gtk_print_settings_get_double [Gtk3 3.6]	gtk_print_settings_get_double_with_default [Gtk3 3.6]
gtk_print_settings_get_duplex [Gtk3 3.6]	gtk_print_settings_get_finishings [Gtk3 3.6]
gtk_print_settings_get_int [Gtk3 3.6]	gtk_print_settings_get_int_with_default [Gtk3 3.6]
gtk_print_settings_get_length [Gtk3 3.6]	gtk_print_settings_get_media_type [Gtk3 3.6]
gtk_print_settings_get_n_copies [Gtk3 3.6]	gtk_print_settings_get_number_up [Gtk3 3.6]
gtk_print_settings_get_number_up_layout [Gtk3 3.6]	gtk_print_settings_get_orientation [Gtk3 3.6]
gtk_print_settings_get_output_bin [Gtk3 3.6]	gtk_print_settings_get_page_ranges [Gtk3 3.6]
gtk_print_settings_get_page_set [Gtk3 3.6]	gtk_print_settings_get_paper_height [Gtk3 3.6]
gtk_print_settings_get_paper_size [Gtk3 3.6]	gtk_print_settings_get_paper_width [Gtk3 3.6]
gtk_print_settings_get_print_pages [Gtk3 3.6]	gtk_print_settings_get_printer [Gtk3 3.6]
gtk_print_settings_get_printer_lpi [Gtk3 3.6]	gtk_print_settings_get_quality [Gtk3 3.6]
gtk_print_settings_get_resolution [Gtk3 3.6]	gtk_print_settings_get_resolution_x [Gtk3 3.6]
gtk_print_settings_get_resolution_y [Gtk3 3.6]	gtk_print_settings_get_reverse [Gtk3 3.6]
gtk_print_settings_get_scale [Gtk3 3.6]	gtk_print_settings_get_use_color [Gtk3 3.6]
gtk_print_settings_has_key [Gtk3 3.6]	gtk_print_settings_load_file [Gtk3 3.6]
gtk_print_settings_load_key_file [Gtk3 3.6]	gtk_print_settings_new [Gtk3 3.6]
gtk_print_settings_new_from_file [Gtk3 3.6]	gtk_print_settings_new_from_key_file [Gtk3 3.6]
gtk_print_settings_set [Gtk3 3.6]	gtk_print_settings_set_bool [Gtk3 3.6]

gtk_print_settings_set_collate [Gtk3 3.6]	gtk_print_settings_set_default_source [Gtk3 3.6]
gtk_print_settings_set_dither [Gtk3 3.6]	gtk_print_settings_set_double [Gtk3 3.6]
gtk_print_settings_set_duplex [Gtk3 3.6]	gtk_print_settings_set_finishings [Gtk3 3.6]
gtk_print_settings_set_int [Gtk3 3.6]	gtk_print_settings_set_length [Gtk3 3.6]
gtk_print_settings_set_media_type [Gtk3 3.6]	gtk_print_settings_set_n_copies [Gtk3 3.6]
gtk_print_settings_set_number_up [Gtk3 3.6]	gtk_print_settings_set_number_up_layout [Gtk3 3.6]
gtk_print_settings_set_orientation [Gtk3 3.6]	gtk_print_settings_set_output_bin [Gtk3 3.6]
gtk_print_settings_set_page_ranges [Gtk3 3.6]	gtk_print_settings_set_page_set [Gtk3 3.6]
gtk_print_settings_set_paper_height [Gtk3 3.6]	gtk_print_settings_set_paper_size [Gtk3 3.6]
gtk_print_settings_set_paper_width [Gtk3 3.6]	gtk_print_settings_set_print_pages [Gtk3 3.6]
gtk_print_settings_set_printer [Gtk3 3.6]	gtk_print_settings_set_printer_lpi [Gtk3 3.6]
gtk_print_settings_set_quality [Gtk3 3.6]	gtk_print_settings_set_resolution [Gtk3 3.6]
gtk_print_settings_set_resolution_xy [Gtk3 3.6]	gtk_print_settings_set_reverse [Gtk3 3.6]
gtk_print_settings_set_scale [Gtk3 3.6]	gtk_print_settings_set_use_color [Gtk3 3.6]
gtk_print_settings_to_file [Gtk3 3.6]	gtk_print_settings_to_key_file [Gtk3 3.6]
gtk_print_settings_unset [Gtk3 3.6]	gtk_printer_accepts_pdf [Gtk3 3.6]
gtk_printer_accepts_ps [Gtk3 3.6]	gtk_printer_compare [Gtk3 3.6]
gtk_printer_get_backend [Gtk3 3.6]	gtk_printer_get_capabilities [Gtk3 3.6]
gtk_printer_get_default_page_size [Gtk3 3.6]	gtk_printer_get_description [Gtk3 3.6]
gtk_printer_get_hard_margins [Gtk3 3.6]	gtk_printer_get_icon_name [Gtk3 3.6]
gtk_printer_get_job_count [Gtk3 3.6]	gtk_printer_get_location [Gtk3 3.6]
gtk_printer_get_name [Gtk3 3.6]	gtk_printer_get_state_message [Gtk3 3.6]

gtk_printer_has_details [Gtk3 3.6]	gtk_printer_is_accepting_jobs [Gtk3 3.6]
gtk_printer_is_active [Gtk3 3.6]	gtk_printer_is_default [Gtk3 3.6]
gtk_printer_is_paused [Gtk3 3.6]	gtk_printer_is_virtual [Gtk3 3.6]
gtk_printer_list_papers [Gtk3 3.6]	gtk_printer_new [Gtk3 3.6]
gtk_printer_request_details [Gtk3 3.6]	gtk_progress_bar_get_ellipsize [Gtk3 3.6]
gtk_progress_bar_get_fraction [Gtk3 3.6]	gtk_progress_bar_get_inverted [Gtk3 3.6]
gtk_progress_bar_get_pulse_step [Gtk3 3.6]	gtk_progress_bar_get_show_text [Gtk3 3.6]
gtk_progress_bar_get_text [Gtk3 3.6]	gtk_progress_bar_new [Gtk3 3.6]
gtk_progress_bar_pulse [Gtk3 3.6]	gtk_progress_bar_set_ellipsize [Gtk3 3.6]
gtk_progress_bar_set_fraction [Gtk3 3.6]	gtk_progress_bar_set_inverted [Gtk3 3.6]
gtk_progress_bar_set_pulse_step [Gtk3 3.6]	gtk_progress_bar_set_show_text [Gtk3 3.6]
gtk_progress_bar_set_text [Gtk3 3.6]	gtk_propagate_event [Gtk3 3.6]
gtk_radio_action_get_current_value [Gtk3 3.6]	gtk_radio_action_get_group [Gtk3 3.6]
gtk_radio_action_join_group [Gtk3 3.6]	gtk_radio_action_new [Gtk3 3.6]
gtk_radio_action_set_current_value [Gtk3 3.6]	gtk_radio_action_set_group [Gtk3 3.6]
gtk_radio_button_get_group [Gtk3 3.6]	gtk_radio_button_join_group [Gtk3 3.6]
gtk_radio_button_new [Gtk3 3.6]	gtk_radio_button_new_from_widget [Gtk3 3.6]
gtk_radio_button_new_with_label [Gtk3 3.6]	gtk_radio_button_new_with_label_from_widget [Gtk3 3.6]
gtk_radio_button_new_with_mnemonic [Gtk3 3.6]	gtk_radio_button_new_with_mnemonic_from_widget [Gtk3 3.6]
gtk_radio_button_set_group [Gtk3 3.6]	gtk_radio_menu_item_get_group [Gtk3 3.6]
gtk_radio_menu_item_new [Gtk3 3.6]	gtk_radio_menu_item_new_from_widget [Gtk3 3.6]
gtk_radio_menu_item_new_with_label [Gtk3 3.6]	gtk_radio_menu_item_new_with_label_from_widget [Gtk3 3.6]
gtk_radio_menu_item_new_with_mnemonic [Gtk3 3.6]	gtk_radio_menu_item_new_with_mnemonic_from_widget [Gtk3 3.6]

gtk_radio_menu_item_set_group [Gtk3 3.6]	gtk_radio_tool_button_get_group [Gtk3 3.6]
gtk_radio_tool_button_new [Gtk3 3.6]	gtk_radio_tool_button_new_from_stock [Gtk3 3.6]
gtk_radio_tool_button_new_from_widget [Gtk3 3.6]	gtk_radio_tool_button_new_with_stock_from_widget [Gtk3 3.6]
gtk_radio_tool_button_set_group [Gtk3 3.6]	gtk_range_get_adjustment [Gtk3 3.6]
gtk_range_get_fill_level [Gtk3 3.6]	gtk_range_get_flippable [Gtk3 3.6]
gtk_range_get_inverted [Gtk3 3.6]	gtk_range_get_lower_stepper_sensitivity [Gtk3 3.6]
gtk_range_get_min_slider_size [Gtk3 3.6]	gtk_range_get_range_rect [Gtk3 3.6]
gtk_range_get_restrict_to_fill_level [Gtk3 3.6]	gtk_range_get_round_digits [Gtk3 3.6]
gtk_range_get_show_fill_level [Gtk3 3.6]	gtk_range_get_slider_range [Gtk3 3.6]
gtk_range_get_slider_size_fixed [Gtk3 3.6]	gtk_range_get_upper_stepper_sensitivity [Gtk3 3.6]
gtk_range_get_value [Gtk3 3.6]	gtk_range_set_adjustment [Gtk3 3.6]
gtk_range_set_fill_level [Gtk3 3.6]	gtk_range_set_flippable [Gtk3 3.6]
gtk_range_set_increments [Gtk3 3.6]	gtk_range_set_inverted [Gtk3 3.6]
gtk_range_set_lower_stepper_sensitivity [Gtk3 3.6]	gtk_range_set_min_slider_size [Gtk3 3.6]
gtk_range_set_range [Gtk3 3.6]	gtk_range_set_restrict_to_fill_level [Gtk3 3.6]
gtk_range_set_round_digits [Gtk3 3.6]	gtk_range_set_show_fill_level [Gtk3 3.6]
gtk_range_set_slider_size_fixed [Gtk3 3.6]	gtk_range_set_upper_stepper_sensitivity [Gtk3 3.6]
gtk_range_set_value [Gtk3 3.6]	gtk_rc_property_parse_border [Gtk3 3.6]
gtk_rc_property_parse_color [Gtk3 3.6]	gtk_rc_property_parse_enum [Gtk3 3.6]
gtk_rc_property_parse_flags [Gtk3 3.6]	gtk_rc_property_parse_requisition [Gtk3 3.6]
gtk_recent_action_get_show_numbers [Gtk3 3.6]	gtk_recent_action_new [Gtk3 3.6]
gtk_recent_action_new_for_manager [Gtk3 3.6]	gtk_recent_action_set_show_numbers [Gtk3 3.6]

gtk_recent_chooser_add_filter [Gtk3 3.6]	gtk_recent_chooser_dialog_new [Gtk3 3.6]
gtk_recent_chooser_dialog_new_for_manager [Gtk3 3.6]	gtk_recent_chooser_get_current_item [Gtk3 3.6]
gtk_recent_chooser_get_current_uri [Gtk3 3.6]	gtk_recent_chooser_get_filter [Gtk3 3.6]
gtk_recent_chooser_get_items [Gtk3 3.6]	gtk_recent_chooser_get_limit [Gtk3 3.6]
gtk_recent_chooser_get_local_only [Gtk3 3.6]	gtk_recent_chooser_get_select_multiple [Gtk3 3.6]
gtk_recent_chooser_get_show_icons [Gtk3 3.6]	gtk_recent_chooser_get_show_not_found [Gtk3 3.6]
gtk_recent_chooser_get_show_private [Gtk3 3.6]	gtk_recent_chooser_get_show_tips [Gtk3 3.6]
gtk_recent_chooser_get_sort_type [Gtk3 3.6]	gtk_recent_chooser_get_uris [Gtk3 3.6]
gtk_recent_chooser_list_filters [Gtk3 3.6]	gtk_recent_chooser_menu_get_show_numbers [Gtk3 3.6]
gtk_recent_chooser_menu_new [Gtk3 3.6]	gtk_recent_chooser_menu_new_for_manager [Gtk3 3.6]
gtk_recent_chooser_menu_set_show_numbers [Gtk3 3.6]	gtk_recent_chooser_remove_filter [Gtk3 3.6]
gtk_recent_chooser_select_all [Gtk3 3.6]	gtk_recent_chooser_select_uri [Gtk3 3.6]
gtk_recent_chooser_set_current_uri [Gtk3 3.6]	gtk_recent_chooser_set_filter [Gtk3 3.6]
gtk_recent_chooser_set_limit [Gtk3 3.6]	gtk_recent_chooser_set_local_only [Gtk3 3.6]
gtk_recent_chooser_set_select_multiple [Gtk3 3.6]	gtk_recent_chooser_set_show_icons [Gtk3 3.6]
gtk_recent_chooser_set_show_not_found [Gtk3 3.6]	gtk_recent_chooser_set_show_private [Gtk3 3.6]
gtk_recent_chooser_set_show_tips [Gtk3 3.6]	gtk_recent_chooser_set_sort_func [Gtk3 3.6]
gtk_recent_chooser_set_sort_type [Gtk3 3.6]	gtk_recent_chooser_unselect_all [Gtk3 3.6]
gtk_recent_chooser_unselect_uri [Gtk3 3.6]	gtk_recent_chooser_widget_new [Gtk3 3.6]
gtk_recent_chooser_widget_new_for_manager [Gtk3 3.6]	gtk_recent_filter_add_age [Gtk3 3.6]
gtk_recent_filter_add_application [Gtk3 3.6]	gtk_recent_filter_add_custom [Gtk3 3.6]

gtk_recent_filter_add_group [Gtk3 3.6]	gtk_recent_filter_add_mime_type [Gtk3 3.6]
gtk_recent_filter_add_pattern [Gtk3 3.6]	gtk_recent_filter_add_pixbuf_formats [Gtk3 3.6]
gtk_recent_filter_filter [Gtk3 3.6]	gtk_recent_filter_get_name [Gtk3 3.6]
gtk_recent_filter_get_needed [Gtk3 3.6]	gtk_recent_filter_new [Gtk3 3.6]
gtk_recent_filter_set_name [Gtk3 3.6]	gtk_recent_info_create_app_info [Gtk3 3.6]
gtk_recent_info_exists [Gtk3 3.6]	gtk_recent_info_get_added [Gtk3 3.6]
gtk_recent_info_get_age [Gtk3 3.6]	gtk_recent_info_get_application_info [Gtk3 3.6]
gtk_recent_info_get_applications [Gtk3 3.6]	gtk_recent_info_get_description [Gtk3 3.6]
gtk_recent_info_get_display_name [Gtk3 3.6]	gtk_recent_info_get_gicon [Gtk3 3.6]
gtk_recent_info_get_groups [Gtk3 3.6]	gtk_recent_info_get_icon [Gtk3 3.6]
gtk_recent_info_get_mime_type [Gtk3 3.6]	gtk_recent_info_get_modified [Gtk3 3.6]
gtk_recent_info_get_private_hint [Gtk3 3.6]	gtk_recent_info_get_short_name [Gtk3 3.6]
gtk_recent_info_get_uri [Gtk3 3.6]	gtk_recent_info_get_uri_display [Gtk3 3.6]
gtk_recent_info_get_visited [Gtk3 3.6]	gtk_recent_info_has_application [Gtk3 3.6]
gtk_recent_info_has_group [Gtk3 3.6]	gtk_recent_info_is_local [Gtk3 3.6]
gtk_recent_info_last_application [Gtk3 3.6]	gtk_recent_info_match [Gtk3 3.6]
gtk_recent_info_ref [Gtk3 3.6]	gtk_recent_info_unref [Gtk3 3.6]
gtk_recent_manager_add_full [Gtk3 3.6]	gtk_recent_manager_add_item [Gtk3 3.6]
gtk_recent_manager_get_default [Gtk3 3.6]	gtk_recent_manager_get_items [Gtk3 3.6]
gtk_recent_manager_has_item [Gtk3 3.6]	gtk_recent_manager_lookup_item [Gtk3 3.6]
gtk_recent_manager_move_item [Gtk3 3.6]	gtk_recent_manager_new [Gtk3 3.6]
gtk_recent_manager_purge_items [Gtk3 3.6]	gtk_recent_manager_remove_item [Gtk3 3.6]

gtk_render_activity [Gtk3 3.6]	gtk_render_arrow [Gtk3 3.6]
gtk_render_background [Gtk3 3.6]	gtk_render_check [Gtk3 3.6]
gtk_render_expander [Gtk3 3.6]	gtk_render_extension [Gtk3 3.6]
gtk_render_focus [Gtk3 3.6]	gtk_render_frame [Gtk3 3.6]
gtk_render_frame_gap [Gtk3 3.6]	gtk_render_handle [Gtk3 3.6]
gtk_render_icon [Gtk3 3.6]	gtk_render_icon_pixbuf [Gtk3 3.6]
gtk_render_insertion_cursor [Gtk3 3.6]	gtk_render_layout [Gtk3 3.6]
gtk_render_line [Gtk3 3.6]	gtk_render_option [Gtk3 3.6]
gtk_render_slider [Gtk3 3.6]	gtk_requisition_copy [Gtk3 3.6]
gtk_requisition_free [Gtk3 3.6]	gtk_requisition_new [Gtk3 3.6]
gtk_rgb_to_hsv [Gtk3 3.6]	gtk_scale_add_mark [Gtk3 3.6]
gtk_scale_button_get_adjustment [Gtk3 3.6]	gtk_scale_button_get_minus_button [Gtk3 3.6]
gtk_scale_button_get_plus_button [Gtk3 3.6]	gtk_scale_button_get_popup [Gtk3 3.6]
gtk_scale_button_get_value [Gtk3 3.6]	gtk_scale_button_new [Gtk3 3.6]
gtk_scale_button_set_adjustment [Gtk3 3.6]	gtk_scale_button_set_icons [Gtk3 3.6]
gtk_scale_button_set_value [Gtk3 3.6]	gtk_scale_clear_marks [Gtk3 3.6]
gtk_scale_get_digits [Gtk3 3.6]	gtk_scale_get_draw_value [Gtk3 3.6]
gtk_scale_get_has_origin [Gtk3 3.6]	gtk_scale_get_layout [Gtk3 3.6]
gtk_scale_get_layout_offsets [Gtk3 3.6]	gtk_scale_get_value_pos [Gtk3 3.6]
gtk_scale_new [Gtk3 3.6]	gtk_scale_new_with_range [Gtk3 3.6]
gtk_scale_set_digits [Gtk3 3.6]	gtk_scale_set_draw_value [Gtk3 3.6]
gtk_scale_set_has_origin [Gtk3 3.6]	gtk_scale_set_value_pos [Gtk3 3.6]
gtk_scrollable_get_hadjustment [Gtk3 3.6]	gtk_scrollable_get_hscroll_policy [Gtk3 3.6]
gtk_scrollable_get_vadjustment [Gtk3 3.6]	gtk_scrollable_get_vscroll_policy [Gtk3 3.6]
gtk_scrollable_set_hadjustment [Gtk3 3.6]	gtk_scrollable_set_hscroll_policy [Gtk3 3.6]
gtk_scrollable_set_vadjustment [Gtk3 3.6]	gtk_scrollable_set_vscroll_policy [Gtk3 3.6]
gtk_scrollbar_new [Gtk3 3.6]	gtk_scrolled_window_add_with_viewport [Gtk3 3.6]

gtk_scrolled_window_get_capture_button_press [Gtk3 3.6]	gtk_scrolled_window_get_hadjustment [Gtk3 3.6]
gtk_scrolled_window_get_hscrollbar [Gtk3 3.6]	gtk_scrolled_window_get_kinetic_scrolling [Gtk3 3.6]
gtk_scrolled_window_get_min_content_height [Gtk3 3.6]	gtk_scrolled_window_get_min_content_width [Gtk3 3.6]
gtk_scrolled_window_get_placement [Gtk3 3.6]	gtk_scrolled_window_get_policy [Gtk3 3.6]
gtk_scrolled_window_get_shadow_type [Gtk3 3.6]	gtk_scrolled_window_get_vadjustment [Gtk3 3.6]
gtk_scrolled_window_get_vscrollbar [Gtk3 3.6]	gtk_scrolled_window_new [Gtk3 3.6]
gtk_scrolled_window_set_capture_button_press [Gtk3 3.6]	gtk_scrolled_window_set_hadjustment [Gtk3 3.6]
gtk_scrolled_window_set_kinetic_scrolling [Gtk3 3.6]	gtk_scrolled_window_set_min_content_height [Gtk3 3.6]
gtk_scrolled_window_set_min_content_width [Gtk3 3.6]	gtk_scrolled_window_set_placement [Gtk3 3.6]
gtk_scrolled_window_set_policy [Gtk3 3.6]	gtk_scrolled_window_set_shadow_type [Gtk3 3.6]
gtk_scrolled_window_set_vadjustment [Gtk3 3.6]	gtk_scrolled_window_unset_placement [Gtk3 3.6]
gtk_search_entry_new [Gtk3 3.6]	gtk_selection_add_target [Gtk3 3.6]
gtk_selection_add_targets [Gtk3 3.6]	gtk_selection_clear_targets [Gtk3 3.6]
gtk_selection_convert [Gtk3 3.6]	gtk_selection_data_copy [Gtk3 3.6]
gtk_selection_data_free [Gtk3 3.6]	gtk_selection_data_get_data [Gtk3 3.6]
gtk_selection_data_get_data_type [Gtk3 3.6]	gtk_selection_data_get_data_with_length [Gtk3 3.6]
gtk_selection_data_get_display [Gtk3 3.6]	gtk_selection_data_get_format [Gtk3 3.6]
gtk_selection_data_get_length [Gtk3 3.6]	gtk_selection_data_get_pixbuf [Gtk3 3.6]
gtk_selection_data_get_selection [Gtk3 3.6]	gtk_selection_data_get_target [Gtk3 3.6]
gtk_selection_data_get_targets [Gtk3 3.6]	gtk_selection_data_get_text [Gtk3 3.6]
gtk_selection_data_get_uris [Gtk3 3.6]	gtk_selection_data_set [Gtk3 3.6]
gtk_selection_data_set_pixbuf [Gtk3 3.6]	gtk_selection_data_set_text [Gtk3 3.6]

gtk_selection_data_set_uris [Gtk3 3.6]	gtk_selection_data_targets_include_image [Gtk3 3.6]
gtk_selection_data_targets_include_rich_text [Gtk3 3.6]	gtk_selection_data_targets_include_text [Gtk3 3.6]
gtk_selection_data_targets_include_uri [Gtk3 3.6]	gtk_selection_owner_set [Gtk3 3.6]
gtk_selection_owner_set_for_display [Gtk3 3.6]	gtk_selection_remove_all [Gtk3 3.6]
gtk_separator_menu_item_new [Gtk3 3.6]	gtk_separator_new [Gtk3 3.6]
gtk_separator_tool_item_get_draw [Gtk3 3.6]	gtk_separator_tool_item_new [Gtk3 3.6]
gtk_separator_tool_item_set_draw [Gtk3 3.6]	gtk_settings_get_default [Gtk3 3.6]
gtk_settings_get_for_screen [Gtk3 3.6]	gtk_settings_install_property [Gtk3 3.6]
gtk_settings_install_property_parser [Gtk3 3.6]	gtk_settings_set_double_property [Gtk3 3.6]
gtk_settings_set_long_property [Gtk3 3.6]	gtk_settings_set_property_value [Gtk3 3.6]
gtk_settings_set_string_property [Gtk3 3.6]	gtk_show_about_dialog [Gtk3 3.6]
gtk_show_uri [Gtk3 3.6]	gtk_size_group_add_widget [Gtk3 3.6]
gtk_size_group_get_ignore_hidden [Gtk3 3.6]	gtk_size_group_get_mode [Gtk3 3.6]
gtk_size_group_get_widgets [Gtk3 3.6]	gtk_size_group_new [Gtk3 3.6]
gtk_size_group_remove_widget [Gtk3 3.6]	gtk_size_group_set_ignore_hidden [Gtk3 3.6]
gtk_size_group_set_mode [Gtk3 3.6]	gtk_socket_add_id [Gtk3 3.6]
gtk_socket_get_id [Gtk3 3.6]	gtk_socket_get_plug_window [Gtk3 3.6]
gtk_socket_new [Gtk3 3.6]	gtk_spin_button_configure [Gtk3 3.6]
gtk_spin_button_get_adjustment [Gtk3 3.6]	gtk_spin_button_get_digits [Gtk3 3.6]
gtk_spin_button_get_increments [Gtk3 3.6]	gtk_spin_button_get_numeric [Gtk3 3.6]
gtk_spin_button_get_range [Gtk3 3.6]	gtk_spin_button_get_snap_to_ticks [Gtk3 3.6]
gtk_spin_button_get_update_policy [Gtk3 3.6]	gtk_spin_button_get_value [Gtk3 3.6]

gtk_spin_button_get_value_as_int [Gtk3 3.6]	gtk_spin_button_get_wrap [Gtk3 3.6]
gtk_spin_button_new [Gtk3 3.6]	gtk_spin_button_new_with_range [Gtk3 3.6]
gtk_spin_button_set_adjustment [Gtk3 3.6]	gtk_spin_button_set_digits [Gtk3 3.6]
gtk_spin_button_set_increments [Gtk3 3.6]	gtk_spin_button_set_numeric [Gtk3 3.6]
gtk_spin_button_set_range [Gtk3 3.6]	gtk_spin_button_set_snap_to_ticks [Gtk3 3.6]
gtk_spin_button_set_update_policy [Gtk3 3.6]	gtk_spin_button_set_value [Gtk3 3.6]
gtk_spin_button_set_wrap [Gtk3 3.6]	gtk_spin_button_spin [Gtk3 3.6]
gtk_spin_button_update [Gtk3 3.6]	gtk_spinner_new [Gtk3 3.6]
gtk_spinner_start [Gtk3 3.6]	gtk_spinner_stop [Gtk3 3.6]
gtk_status_icon_get_geometry [Gtk3 3.6]	gtk_status_icon_get_gicon [Gtk3 3.6]
gtk_status_icon_get_has_tooltip [Gtk3 3.6]	gtk_status_icon_get_icon_name [Gtk3 3.6]
gtk_status_icon_get_pixbuf [Gtk3 3.6]	gtk_status_icon_get_screen [Gtk3 3.6]
gtk_status_icon_get_size [Gtk3 3.6]	gtk_status_icon_get_stock [Gtk3 3.6]
gtk_status_icon_get_storage_type [Gtk3 3.6]	gtk_status_icon_get_title [Gtk3 3.6]
gtk_status_icon_get_tooltip_markup [Gtk3 3.6]	gtk_status_icon_get_tooltip_text [Gtk3 3.6]
gtk_status_icon_get_visible [Gtk3 3.6]	gtk_status_icon_get_x11_window_id [Gtk3 3.6]
gtk_status_icon_is_embedded [Gtk3 3.6]	gtk_status_icon_new [Gtk3 3.6]
gtk_status_icon_new_from_file [Gtk3 3.6]	gtk_status_icon_new_from_gicon [Gtk3 3.6]
gtk_status_icon_new_from_icon_name [Gtk3 3.6]	gtk_status_icon_new_from_pixbuf [Gtk3 3.6]
gtk_status_icon_new_from_stock [Gtk3 3.6]	gtk_status_icon_position_menu [Gtk3 3.6]
gtk_status_icon_set_from_file [Gtk3 3.6]	gtk_status_icon_set_from_gicon [Gtk3 3.6]
gtk_status_icon_set_from_icon_name [Gtk3 3.6]	gtk_status_icon_set_from_pixbuf [Gtk3 3.6]

gtk_status_icon_set_from_stock [Gtk3 3.6]	gtk_status_icon_set_has_tooltip [Gtk3 3.6]
gtk_status_icon_set_name [Gtk3 3.6]	gtk_status_icon_set_screen [Gtk3 3.6]
gtk_status_icon_set_title [Gtk3 3.6]	gtk_status_icon_set_tooltip_markup [Gtk3 3.6]
gtk_status_icon_set_tooltip_text [Gtk3 3.6]	gtk_status_icon_set_visible [Gtk3 3.6]
gtk_statusbar_get_context_id [Gtk3 3.6]	gtk_statusbar_get_message_area [Gtk3 3.6]
gtk_statusbar_new [Gtk3 3.6]	gtk_statusbar_pop [Gtk3 3.6]
gtk_statusbar_push [Gtk3 3.6]	gtk_statusbar_remove [Gtk3 3.6]
gtk_statusbar_remove_all [Gtk3 3.6]	gtk_stock_add [Gtk3 3.6]
gtk_stock_add_static [Gtk3 3.6]	gtk_stock_item_copy [Gtk3 3.6]
gtk_stock_item_free [Gtk3 3.6]	gtk_stock_list_ids [Gtk3 3.6]
gtk_stock_lookup [Gtk3 3.6]	gtk_stock_set_translate_func [Gtk3 3.6]
gtk_style_context_add_class [Gtk3 3.6]	gtk_style_context_add_provider [Gtk3 3.6]
gtk_style_context_add_provider_for_screen [Gtk3 3.6]	gtk_style_context_add_region [Gtk3 3.6]
gtk_style_context_cancel_animations [Gtk3 3.6]	gtk_style_context_get [Gtk3 3.6]
gtk_style_context_get_background_color [Gtk3 3.6]	gtk_style_context_get_border [Gtk3 3.6]
gtk_style_context_get_border_color [Gtk3 3.6]	gtk_style_context_get_color [Gtk3 3.6]
gtk_style_context_get_direction [Gtk3 3.6]	gtk_style_context_get_font [Gtk3 3.6]
gtk_style_context_get_junction_sides [Gtk3 3.6]	gtk_style_context_get_margin [Gtk3 3.6]
gtk_style_context_get_padding [Gtk3 3.6]	gtk_style_context_get_parent [Gtk3 3.6]
gtk_style_context_get_path [Gtk3 3.6]	gtk_style_context_get_property [Gtk3 3.6]
gtk_style_context_get_screen [Gtk3 3.6]	gtk_style_context_get_section [Gtk3 3.6]
gtk_style_context_get_state [Gtk3 3.6]	gtk_style_context_get_style [Gtk3 3.6]
gtk_style_context_get_style_property [Gtk3 3.6]	gtk_style_context_get_style_valist [Gtk3 3.6]

gtk_style_context_get_valist [Gtk3 3.6]	gtk_style_context_has_class [Gtk3 3.6]
gtk_style_context_has_region [Gtk3 3.6]	gtk_style_context_invalidate [Gtk3 3.6]
gtk_style_context_list_classes [Gtk3 3.6]	gtk_style_context_list_regions [Gtk3 3.6]
gtk_style_context_lookup_color [Gtk3 3.6]	gtk_style_context_lookup_icon_set [Gtk3 3.6]
gtk_style_context_new [Gtk3 3.6]	gtk_style_context_notify_state_change [Gtk3 3.6]
gtk_style_context_pop_animatable_region [Gtk3 3.6]	gtk_style_context_push_animatable_region [Gtk3 3.6]
gtk_style_context_remove_class [Gtk3 3.6]	gtk_style_context_remove_provider [Gtk3 3.6]
gtk_style_context_remove_provider_for_screen [Gtk3 3.6]	gtk_style_context_remove_region [Gtk3 3.6]
gtk_style_context_reset_widgets [Gtk3 3.6]	gtk_style_context_restore [Gtk3 3.6]
gtk_style_context_save [Gtk3 3.6]	gtk_style_context_scroll_animations [Gtk3 3.6]
gtk_style_context_set_background [Gtk3 3.6]	gtk_style_context_set_direction [Gtk3 3.6]
gtk_style_context_set_junction_sides [Gtk3 3.6]	gtk_style_context_set_parent [Gtk3 3.6]
gtk_style_context_set_path [Gtk3 3.6]	gtk_style_context_set_screen [Gtk3 3.6]
gtk_style_context_set_state [Gtk3 3.6]	gtk_style_context_state_is_running [Gtk3 3.6]
gtk_style_properties_clear [Gtk3 3.6]	gtk_style_properties_get [Gtk3 3.6]
gtk_style_properties_get_property [Gtk3 3.6]	gtk_style_properties_get_valist [Gtk3 3.6]
gtk_style_properties_lookup_color [Gtk3 3.6]	gtk_style_properties_lookup_property [Gtk3 3.6]
gtk_style_properties_map_color [Gtk3 3.6]	gtk_style_properties_merge [Gtk3 3.6]
gtk_style_properties_new [Gtk3 3.6]	gtk_style_properties_register_property [Gtk3 3.6]
gtk_style_properties_set [Gtk3 3.6]	gtk_style_properties_set_property [Gtk3 3.6]
gtk_style_properties_set_valist [Gtk3 3.6]	gtk_style_properties_unset_property [Gtk3 3.6]

gtk_style_provider_get_icon_factory [Gtk3 3.6]	gtk_style_provider_get_style [Gtk3 3.6]
gtk_style_provider_get_style_property [Gtk3 3.6]	gtk_switch_get_active [Gtk3 3.6]
gtk_switch_new [Gtk3 3.6]	gtk_switch_set_active [Gtk3 3.6]
gtk_symbolic_color_new_alpha [Gtk3 3.6]	gtk_symbolic_color_new_literal [Gtk3 3.6]
gtk_symbolic_color_new_mix [Gtk3 3.6]	gtk_symbolic_color_new_name [Gtk3 3.6]
gtk_symbolic_color_new_shade [Gtk3 3.6]	gtk_symbolic_color_new_win32 [Gtk3 3.6]
gtk_symbolic_color_ref [Gtk3 3.6]	gtk_symbolic_color_resolve [Gtk3 3.6]
gtk_symbolic_color_to_string [Gtk3 3.6]	gtk_symbolic_color_unref [Gtk3 3.6]
gtk_target_entry_copy [Gtk3 3.6]	gtk_target_entry_free [Gtk3 3.6]
gtk_target_entry_new [Gtk3 3.6]	gtk_target_list_add [Gtk3 3.6]
gtk_target_list_add_image_targets [Gtk3 3.6]	gtk_target_list_add_rich_text_targets [Gtk3 3.6]
gtk_target_list_add_table [Gtk3 3.6]	gtk_target_list_add_text_targets [Gtk3 3.6]
gtk_target_list_add_uri_targets [Gtk3 3.6]	gtk_target_list_find [Gtk3 3.6]
gtk_target_list_new [Gtk3 3.6]	gtk_target_list_ref [Gtk3 3.6]
gtk_target_list_remove [Gtk3 3.6]	gtk_target_list_unref [Gtk3 3.6]
gtk_target_table_free [Gtk3 3.6]	gtk_target_table_new_from_list [Gtk3 3.6]
gtk_targets_include_image [Gtk3 3.6]	gtk_targets_include_rich_text [Gtk3 3.6]
gtk_targets_include_text [Gtk3 3.6]	gtk_targets_include_uri [Gtk3 3.6]
gtk_test_create_simple_window [Gtk3 3.6]	gtk_test_create_widget [Gtk3 3.6]
gtk_test_display_button_window [Gtk3 3.6]	gtk_test_find_label [Gtk3 3.6]
gtk_test_find_sibling [Gtk3 3.6]	gtk_test_find_widget [Gtk3 3.6]
gtk_test_init [Gtk3 3.6]	gtk_test_list_all_types [Gtk3 3.6]
gtk_test_register_all_types [Gtk3 3.6]	gtk_test_slider_get_value [Gtk3 3.6]
gtk_test_slider_set_perc [Gtk3 3.6]	gtk_test_spin_button_click [Gtk3 3.6]
gtk_test_text_get [Gtk3 3.6]	gtk_test_text_set [Gtk3 3.6]

gtk_test_widget_click [Gtk3 3.6]	gtk_test_widget_send_key [Gtk3 3.6]
gtk_text_attributes_copy [Gtk3 3.6]	gtk_text_attributes_copy_values [Gtk3 3.6]
gtk_text_attributes_new [Gtk3 3.6]	gtk_text_attributes_ref [Gtk3 3.6]
gtk_text_attributes_unref [Gtk3 3.6]	gtk_text_buffer_add_mark [Gtk3 3.6]
gtk_text_buffer_add_selection_clipboard [Gtk3 3.6]	gtk_text_buffer_apply_tag [Gtk3 3.6]
gtk_text_buffer_apply_tag_by_name [Gtk3 3.6]	gtk_text_buffer_backspace [Gtk3 3.6]
gtk_text_buffer_begin_user_action [Gtk3 3.6]	gtk_text_buffer_copy_clipboard [Gtk3 3.6]
gtk_text_buffer_create_child_anchor [Gtk3 3.6]	gtk_text_buffer_create_mark [Gtk3 3.6]
gtk_text_buffer_create_tag [Gtk3 3.6]	gtk_text_buffer_cut_clipboard [Gtk3 3.6]
gtk_text_buffer_delete [Gtk3 3.6]	gtk_text_buffer_delete_interactive [Gtk3 3.6]
gtk_text_buffer_delete_mark [Gtk3 3.6]	gtk_text_buffer_delete_mark_by_name [Gtk3 3.6]
gtk_text_buffer_delete_selection [Gtk3 3.6]	gtk_text_buffer_deserialize [Gtk3 3.6]
gtk_text_buffer_deserialize_get_can_create_tags [Gtk3 3.6]	gtk_text_buffer_deserialize_set_can_create_tags [Gtk3 3.6]
gtk_text_buffer_end_user_action [Gtk3 3.6]	gtk_text_buffer_get_bounds [Gtk3 3.6]
gtk_text_buffer_get_char_count [Gtk3 3.6]	gtk_text_buffer_get_copy_target_list [Gtk3 3.6]
gtk_text_buffer_get_deserialize_formats [Gtk3 3.6]	gtk_text_buffer_get_end_iter [Gtk3 3.6]
gtk_text_buffer_get_has_selection [Gtk3 3.6]	gtk_text_buffer_get_insert [Gtk3 3.6]
gtk_text_buffer_get_iter_at_child_anchor [Gtk3 3.6]	gtk_text_buffer_get_iter_at_line [Gtk3 3.6]
gtk_text_buffer_get_iter_at_line_index [Gtk3 3.6]	gtk_text_buffer_get_iter_at_line_offset [Gtk3 3.6]
gtk_text_buffer_get_iter_at_mark [Gtk3 3.6]	gtk_text_buffer_get_iter_at_offset [Gtk3 3.6]
gtk_text_buffer_get_line_count [Gtk3 3.6]	gtk_text_buffer_get_mark [Gtk3 3.6]
gtk_text_buffer_get_modified [Gtk3 3.6]	gtk_text_buffer_get_paste_target_list [Gtk3 3.6]

gtk_text_buffer_get_selection_bound [Gtk3 3.6]	gtk_text_buffer_get_selection_bound s [Gtk3 3.6]
gtk_text_buffer_get_serialize_format s [Gtk3 3.6]	gtk_text_buffer_get_slice [Gtk3 3.6]
gtk_text_buffer_get_start_iter [Gtk3 3.6]	gtk_text_buffer_get_tag_table [Gtk3 3.6]
gtk_text_buffer_get_text [Gtk3 3.6]	gtk_text_buffer_insert [Gtk3 3.6]
gtk_text_buffer_insert_at_cursor [Gtk3 3.6]	gtk_text_buffer_insert_child_anchor [Gtk3 3.6]
gtk_text_buffer_insert_interactive [Gtk3 3.6]	gtk_text_buffer_insert_interactive_at _cursor [Gtk3 3.6]
gtk_text_buffer_insert_pixbuf [Gtk3 3.6]	gtk_text_buffer_insert_range [Gtk3 3.6]
gtk_text_buffer_insert_range_interact ive [Gtk3 3.6]	gtk_text_buffer_insert_with_tags [Gtk3 3.6]
gtk_text_buffer_insert_with_tags_by _name [Gtk3 3.6]	gtk_text_buffer_move_mark [Gtk3 3.6]
gtk_text_buffer_move_mark_by_na me [Gtk3 3.6]	gtk_text_buffer_new [Gtk3 3.6]
gtk_text_buffer_paste_clipboard [Gtk3 3.6]	gtk_text_buffer_place_cursor [Gtk3 3.6]
gtk_text_buffer_register_deserialize_ format [Gtk3 3.6]	gtk_text_buffer_register_deserialize_ tagset [Gtk3 3.6]
gtk_text_buffer_register_serialize_for mat [Gtk3 3.6]	gtk_text_buffer_register_serialize_tag set [Gtk3 3.6]
gtk_text_buffer_remove_all_tags [Gtk3 3.6]	gtk_text_buffer_remove_selection_cli pboard [Gtk3 3.6]
gtk_text_buffer_remove_tag [Gtk3 3.6]	gtk_text_buffer_remove_tag_by_nam e [Gtk3 3.6]
gtk_text_buffer_select_range [Gtk3 3.6]	gtk_text_buffer_serialize [Gtk3 3.6]
gtk_text_buffer_set_modified [Gtk3 3.6]	gtk_text_buffer_set_text [Gtk3 3.6]
gtk_text_buffer_unregister_deserializ e_format [Gtk3 3.6]	gtk_text_buffer_unregister_serialize_ format [Gtk3 3.6]
gtk_text_child_anchor_get_deleted [Gtk3 3.6]	gtk_text_child_anchor_get_widgets [Gtk3 3.6]
gtk_text_child_anchor_new [Gtk3 3.6]	gtk_text_iter_assign [Gtk3 3.6]
gtk_text_iter_backward_char [Gtk3 3.6]	gtk_text_iter_backward_chars [Gtk3 3.6]

gtk_text_iter_backward_cursor_position [Gtk3 3.6]	gtk_text_iter_backward_cursor_positions [Gtk3 3.6]
gtk_text_iter_backward_find_char [Gtk3 3.6]	gtk_text_iter_backward_line [Gtk3 3.6]
gtk_text_iter_backward_lines [Gtk3 3.6]	gtk_text_iter_backward_search [Gtk3 3.6]
gtk_text_iter_backward_sentence_start [Gtk3 3.6]	gtk_text_iter_backward_sentence_starts [Gtk3 3.6]
gtk_text_iter_backward_to_tag_toggle [Gtk3 3.6]	gtk_text_iter_backward_visible_cursor_position [Gtk3 3.6]
gtk_text_iter_backward_visible_cursor_positions [Gtk3 3.6]	gtk_text_iter_backward_visible_line [Gtk3 3.6]
gtk_text_iter_backward_visible_lines [Gtk3 3.6]	gtk_text_iter_backward_visible_word_start [Gtk3 3.6]
gtk_text_iter_backward_visible_word_starts [Gtk3 3.6]	gtk_text_iter_backward_word_start [Gtk3 3.6]
gtk_text_iter_backward_word_starts [Gtk3 3.6]	gtk_text_iter_begins_tag [Gtk3 3.6]
gtk_text_iter_can_insert [Gtk3 3.6]	gtk_text_iter_compare [Gtk3 3.6]
gtk_text_iter_copy [Gtk3 3.6]	gtk_text_iter_editable [Gtk3 3.6]
gtk_text_iter_ends_line [Gtk3 3.6]	gtk_text_iter_ends_sentence [Gtk3 3.6]
gtk_text_iter_ends_tag [Gtk3 3.6]	gtk_text_iter_ends_word [Gtk3 3.6]
gtk_text_iter_equal [Gtk3 3.6]	gtk_text_iter_forward_char [Gtk3 3.6]
gtk_text_iter_forward_chars [Gtk3 3.6]	gtk_text_iter_forward_cursor_position [Gtk3 3.6]
gtk_text_iter_forward_cursor_positions [Gtk3 3.6]	gtk_text_iter_forward_find_char [Gtk3 3.6]
gtk_text_iter_forward_line [Gtk3 3.6]	gtk_text_iter_forward_lines [Gtk3 3.6]
gtk_text_iter_forward_search [Gtk3 3.6]	gtk_text_iter_forward_sentence_end [Gtk3 3.6]
gtk_text_iter_forward_sentence_ends [Gtk3 3.6]	gtk_text_iter_forward_to_end [Gtk3 3.6]
gtk_text_iter_forward_to_line_end [Gtk3 3.6]	gtk_text_iter_forward_to_tag_toggle [Gtk3 3.6]
gtk_text_iter_forward_visible_cursor_position [Gtk3 3.6]	gtk_text_iter_forward_visible_cursor_positions [Gtk3 3.6]
gtk_text_iter_forward_visible_line [Gtk3 3.6]	gtk_text_iter_forward_visible_lines [Gtk3 3.6]

gtk_text_iter_forward_visible_word_end [Gtk3 3.6]	gtk_text_iter_forward_visible_word_ends [Gtk3 3.6]
gtk_text_iter_forward_word_end [Gtk3 3.6]	gtk_text_iter_forward_word_ends [Gtk3 3.6]
gtk_text_iter_free [Gtk3 3.6]	gtk_text_iter_get_attributes [Gtk3 3.6]
gtk_text_iter_get_buffer [Gtk3 3.6]	gtk_text_iter_get_bytes_in_line [Gtk3 3.6]
gtk_text_iter_get_char [Gtk3 3.6]	gtk_text_iter_get_chars_in_line [Gtk3 3.6]
gtk_text_iter_get_child_anchor [Gtk3 3.6]	gtk_text_iter_get_language [Gtk3 3.6]
gtk_text_iter_get_line [Gtk3 3.6]	gtk_text_iter_get_line_index [Gtk3 3.6]
gtk_text_iter_get_line_offset [Gtk3 3.6]	gtk_text_iter_get_marks [Gtk3 3.6]
gtk_text_iter_get_offset [Gtk3 3.6]	gtk_text_iter_get_pixbuf [Gtk3 3.6]
gtk_text_iter_get_slice [Gtk3 3.6]	gtk_text_iter_get_tags [Gtk3 3.6]
gtk_text_iter_get_text [Gtk3 3.6]	gtk_text_iter_get_toggled_tags [Gtk3 3.6]
gtk_text_iter_get_visible_line_index [Gtk3 3.6]	gtk_text_iter_get_visible_line_offset [Gtk3 3.6]
gtk_text_iter_get_visible_slice [Gtk3 3.6]	gtk_text_iter_get_visible_text [Gtk3 3.6]
gtk_text_iter_has_tag [Gtk3 3.6]	gtk_text_iter_in_range [Gtk3 3.6]
gtk_text_iter_inside_sentence [Gtk3 3.6]	gtk_text_iter_inside_word [Gtk3 3.6]
gtk_text_iter_is_cursor_position [Gtk3 3.6]	gtk_text_iter_is_end [Gtk3 3.6]
gtk_text_iter_is_start [Gtk3 3.6]	gtk_text_iter_order [Gtk3 3.6]
gtk_text_iter_set_line [Gtk3 3.6]	gtk_text_iter_set_line_index [Gtk3 3.6]
gtk_text_iter_set_line_offset [Gtk3 3.6]	gtk_text_iter_set_offset [Gtk3 3.6]
gtk_text_iter_set_visible_line_index [Gtk3 3.6]	gtk_text_iter_set_visible_line_offset [Gtk3 3.6]
gtk_text_iter_starts_line [Gtk3 3.6]	gtk_text_iter_starts_sentence [Gtk3 3.6]
gtk_text_iter_starts_word [Gtk3 3.6]	gtk_text_iter_toggles_tag [Gtk3 3.6]
gtk_text_mark_get_buffer [Gtk3 3.6]	gtk_text_mark_get_deleted [Gtk3 3.6]

gtk_text_mark_get_left_gravity [Gtk3 3.6]	gtk_text_mark_get_name [Gtk3 3.6]
gtk_text_mark_get_visible [Gtk3 3.6]	gtk_text_mark_new [Gtk3 3.6]
gtk_text_mark_set_visible [Gtk3 3.6]	gtk_text_tag_event [Gtk3 3.6]
gtk_text_tag_get_priority [Gtk3 3.6]	gtk_text_tag_new [Gtk3 3.6]
gtk_text_tag_set_priority [Gtk3 3.6]	gtk_text_tag_table_add [Gtk3 3.6]
gtk_text_tag_table_foreach [Gtk3 3.6]	gtk_text_tag_table_get_size [Gtk3 3.6]
gtk_text_tag_table_lookup [Gtk3 3.6]	gtk_text_tag_table_new [Gtk3 3.6]
gtk_text_tag_table_remove [Gtk3 3.6]	gtk_text_view_add_child_at_anchor [Gtk3 3.6]
gtk_text_view_add_child_in_window [Gtk3 3.6]	gtk_text_view_backward_display_line [Gtk3 3.6]
gtk_text_view_backward_display_line_start [Gtk3 3.6]	gtk_text_view_buffer_to_window_coords [Gtk3 3.6]
gtk_text_view_forward_display_line [Gtk3 3.6]	gtk_text_view_forward_display_line_end [Gtk3 3.6]
gtk_text_view_get_accepts_tab [Gtk3 3.6]	gtk_text_view_get_border_window_size [Gtk3 3.6]
gtk_text_view_get_buffer [Gtk3 3.6]	gtk_text_view_get_cursor_locations [Gtk3 3.6]
gtk_text_view_get_cursor_visible [Gtk3 3.6]	gtk_text_view_get_default_attributes [Gtk3 3.6]
gtk_text_view_get_editable [Gtk3 3.6]	gtk_text_view_get_hadjustment [Gtk3 3.6]
gtk_text_view_get_indent [Gtk3 3.6]	gtk_text_view_get_input_hints [Gtk3 3.6]
gtk_text_view_get_input_purpose [Gtk3 3.6]	gtk_text_view_get_iter_at_location [Gtk3 3.6]
gtk_text_view_get_iter_at_position [Gtk3 3.6]	gtk_text_view_get_iter_location [Gtk3 3.6]
gtk_text_view_get_justification [Gtk3 3.6]	gtk_text_view_get_left_margin [Gtk3 3.6]
gtk_text_view_get_line_at_y [Gtk3 3.6]	gtk_text_view_get_line_yrange [Gtk3 3.6]
gtk_text_view_get_overwrite [Gtk3 3.6]	gtk_text_view_get_pixels_above_lines [Gtk3 3.6]
gtk_text_view_get_pixels_below_lines [Gtk3 3.6]	gtk_text_view_get_pixels_inside_wrap [Gtk3 3.6]
gtk_text_view_get_right_margin [Gtk3 3.6]	gtk_text_view_get_tabs [Gtk3 3.6]

gtk_text_view_get_vadjustment [Gtk3 3.6]	gtk_text_view_get_visible_rect [Gtk3 3.6]
gtk_text_view_get_window [Gtk3 3.6]	gtk_text_view_get_window_type [Gtk3 3.6]
gtk_text_view_get_wrap_mode [Gtk3 3.6]	gtk_text_view_im_context_filter_key_press [Gtk3 3.6]
gtk_text_view_move_child [Gtk3 3.6]	gtk_text_view_move_mark_onscreen [Gtk3 3.6]
gtk_text_view_move_visually [Gtk3 3.6]	gtk_text_view_new [Gtk3 3.6]
gtk_text_view_new_with_buffer [Gtk3 3.6]	gtk_text_view_place_cursor_onscreen [Gtk3 3.6]
gtk_text_view_reset_im_context [Gtk3 3.6]	gtk_text_view_scroll_mark_onscreen [Gtk3 3.6]
gtk_text_view_scroll_to_iter [Gtk3 3.6]	gtk_text_view_scroll_to_mark [Gtk3 3.6]
gtk_text_view_set_accepts_tab [Gtk3 3.6]	gtk_text_view_set_border_window_size [Gtk3 3.6]
gtk_text_view_set_buffer [Gtk3 3.6]	gtk_text_view_set_cursor_visible [Gtk3 3.6]
gtk_text_view_set_editable [Gtk3 3.6]	gtk_text_view_set_indent [Gtk3 3.6]
gtk_text_view_set_input_hints [Gtk3 3.6]	gtk_text_view_set_input_purpose [Gtk3 3.6]
gtk_text_view_set_justification [Gtk3 3.6]	gtk_text_view_set_left_margin [Gtk3 3.6]
gtk_text_view_set_overwrite [Gtk3 3.6]	gtk_text_view_set_pixels_above_lines [Gtk3 3.6]
gtk_text_view_set_pixels_below_lines [Gtk3 3.6]	gtk_text_view_set_pixels_inside_wrap [Gtk3 3.6]
gtk_text_view_set_right_margin [Gtk3 3.6]	gtk_text_view_set_tabs [Gtk3 3.6]
gtk_text_view_set_wrap_mode [Gtk3 3.6]	gtk_text_view_starts_display_line [Gtk3 3.6]
gtk_text_view_window_to_buffer_coords [Gtk3 3.6]	gtk_theming_engine_get [Gtk3 3.6]
gtk_theming_engine_get_background_color [Gtk3 3.6]	gtk_theming_engine_get_border [Gtk3 3.6]
gtk_theming_engine_get_border_color [Gtk3 3.6]	gtk_theming_engine_get_color [Gtk3 3.6]
gtk_theming_engine_get_direction [Gtk3 3.6]	gtk_theming_engine_get_font [Gtk3 3.6]

gtk_theming_engine_get_junction_sides [Gtk3 3.6]	gtk_theming_engine_get_margin [Gtk3 3.6]
gtk_theming_engine_get_padding [Gtk3 3.6]	gtk_theming_engine_get_path [Gtk3 3.6]
gtk_theming_engine_get_property [Gtk3 3.6]	gtk_theming_engine_get_screen [Gtk3 3.6]
gtk_theming_engine_get_state [Gtk3 3.6]	gtk_theming_engine_get_style [Gtk3 3.6]
gtk_theming_engine_get_style_property [Gtk3 3.6]	gtk_theming_engine_get_style_valist [Gtk3 3.6]
gtk_theming_engine_get_valist [Gtk3 3.6]	gtk_theming_engine_has_class [Gtk3 3.6]
gtk_theming_engine_has_region [Gtk3 3.6]	gtk_theming_engine_load [Gtk3 3.6]
gtk_theming_engine_lookup_color [Gtk3 3.6]	gtk_theming_engine_register_property [Gtk3 3.6]
gtk_theming_engine_state_is_running [Gtk3 3.6]	gtk_toggle_action_get_active [Gtk3 3.6]
gtk_toggle_action_get_draw_as_radio [Gtk3 3.6]	gtk_toggle_action_new [Gtk3 3.6]
gtk_toggle_action_set_active [Gtk3 3.6]	gtk_toggle_action_set_draw_as_radio [Gtk3 3.6]
gtk_toggle_action_toggled [Gtk3 3.6]	gtk_toggle_button_get_active [Gtk3 3.6]
gtk_toggle_button_get_inconsistent [Gtk3 3.6]	gtk_toggle_button_get_mode [Gtk3 3.6]
gtk_toggle_button_new [Gtk3 3.6]	gtk_toggle_button_new_with_label [Gtk3 3.6]
gtk_toggle_button_new_with_mnemonic [Gtk3 3.6]	gtk_toggle_button_set_active [Gtk3 3.6]
gtk_toggle_button_set_inconsistent [Gtk3 3.6]	gtk_toggle_button_set_mode [Gtk3 3.6]
gtk_toggle_button_toggled [Gtk3 3.6]	gtk_toggle_tool_button_get_active [Gtk3 3.6]
gtk_toggle_tool_button_new [Gtk3 3.6]	gtk_toggle_tool_button_new_from_stock [Gtk3 3.6]
gtk_toggle_tool_button_set_active [Gtk3 3.6]	gtk_tool_button_get_icon_name [Gtk3 3.6]
gtk_tool_button_get_icon_widget [Gtk3 3.6]	gtk_tool_button_get_label [Gtk3 3.6]
gtk_tool_button_get_label_widget [Gtk3 3.6]	gtk_tool_button_get_stock_id [Gtk3 3.6]

gtk_tool_button_get_use_underline [Gtk3 3.6]	gtk_tool_button_new [Gtk3 3.6]
gtk_tool_button_new_from_stock [Gtk3 3.6]	gtk_tool_button_set_icon_name [Gtk3 3.6]
gtk_tool_button_set_icon_widget [Gtk3 3.6]	gtk_tool_button_set_label [Gtk3 3.6]
gtk_tool_button_set_label_widget [Gtk3 3.6]	gtk_tool_button_set_stock_id [Gtk3 3.6]
gtk_tool_button_set_use_underline [Gtk3 3.6]	gtk_tool_item_get_ellipsize_mode [Gtk3 3.6]
gtk_tool_item_get_expand [Gtk3 3.6]	gtk_tool_item_get_homogeneous [Gtk3 3.6]
gtk_tool_item_get_icon_size [Gtk3 3.6]	gtk_tool_item_get_is_important [Gtk3 3.6]
gtk_tool_item_get_orientation [Gtk3 3.6]	gtk_tool_item_get_proxy_menu_item [Gtk3 3.6]
gtk_tool_item_get_relief_style [Gtk3 3.6]	gtk_tool_item_get_text_alignment [Gtk3 3.6]
gtk_tool_item_get_text_orientation [Gtk3 3.6]	gtk_tool_item_get_text_size_group [Gtk3 3.6]
gtk_tool_item_get_toolbar_style [Gtk3 3.6]	gtk_tool_item_get_use_drag_window [Gtk3 3.6]
gtk_tool_item_get_visible_horizontal [Gtk3 3.6]	gtk_tool_item_get_visible_vertical [Gtk3 3.6]
gtk_tool_item_group_get_collapsed [Gtk3 3.6]	gtk_tool_item_group_get_drop_item [Gtk3 3.6]
gtk_tool_item_group_get_ellipsize [Gtk3 3.6]	gtk_tool_item_group_get_header_relief [Gtk3 3.6]
gtk_tool_item_group_get_item_position [Gtk3 3.6]	gtk_tool_item_group_get_label [Gtk3 3.6]
gtk_tool_item_group_get_label_widget [Gtk3 3.6]	gtk_tool_item_group_get_n_items [Gtk3 3.6]
gtk_tool_item_group_get_nth_item [Gtk3 3.6]	gtk_tool_item_group_insert [Gtk3 3.6]
gtk_tool_item_group_new [Gtk3 3.6]	gtk_tool_item_group_set_collapsed [Gtk3 3.6]
gtk_tool_item_group_set_ellipsize [Gtk3 3.6]	gtk_tool_item_group_set_header_relief [Gtk3 3.6]
gtk_tool_item_group_set_item_position [Gtk3 3.6]	gtk_tool_item_group_set_label [Gtk3 3.6]
gtk_tool_item_group_set_label_widget [Gtk3 3.6]	gtk_tool_item_new [Gtk3 3.6]

gtk_tool_item_rebuild_menu [Gtk3 3.6]	gtk_tool_item_retrieve_proxy_menu_item [Gtk3 3.6]
gtk_tool_item_set_expand [Gtk3 3.6]	gtk_tool_item_set_homogeneous [Gtk3 3.6]
gtk_tool_item_set_is_important [Gtk3 3.6]	gtk_tool_item_set_proxy_menu_item [Gtk3 3.6]
gtk_tool_item_set_tooltip_markup [Gtk3 3.6]	gtk_tool_item_set_tooltip_text [Gtk3 3.6]
gtk_tool_item_set_use_drag_window [Gtk3 3.6]	gtk_tool_item_set_visible_horizontal [Gtk3 3.6]
gtk_tool_item_set_visible_vertical [Gtk3 3.6]	gtk_tool_item_toolbar_reconfigured [Gtk3 3.6]
gtk_tool_palette_add_drag_dest [Gtk3 3.6]	gtk_tool_palette_get_drag_item [Gtk3 3.6]
gtk_tool_palette_get_drag_target_group [Gtk3 3.6]	gtk_tool_palette_get_drag_target_item [Gtk3 3.6]
gtk_tool_palette_get_drop_group [Gtk3 3.6]	gtk_tool_palette_get_drop_item [Gtk3 3.6]
gtk_tool_palette_get_exclusive [Gtk3 3.6]	gtk_tool_palette_get_expand [Gtk3 3.6]
gtk_tool_palette_get_group_position [Gtk3 3.6]	gtk_tool_palette_get_hadjustment [Gtk3 3.6]
gtk_tool_palette_get_icon_size [Gtk3 3.6]	gtk_tool_palette_get_style [Gtk3 3.6]
gtk_tool_palette_get_vadjustment [Gtk3 3.6]	gtk_tool_palette_new [Gtk3 3.6]
gtk_tool_palette_set_drag_source [Gtk3 3.6]	gtk_tool_palette_set_exclusive [Gtk3 3.6]
gtk_tool_palette_set_expand [Gtk3 3.6]	gtk_tool_palette_set_group_position [Gtk3 3.6]
gtk_tool_palette_set_icon_size [Gtk3 3.6]	gtk_tool_palette_set_style [Gtk3 3.6]
gtk_tool_palette_unset_icon_size [Gtk3 3.6]	gtk_tool_palette_unset_style [Gtk3 3.6]
gtk_tool_shell_get_ellipsize_mode [Gtk3 3.6]	gtk_tool_shell_get_icon_size [Gtk3 3.6]
gtk_tool_shell_get_orientation [Gtk3 3.6]	gtk_tool_shell_get_relief_style [Gtk3 3.6]
gtk_tool_shell_get_style [Gtk3 3.6]	gtk_tool_shell_get_text_alignment [Gtk3 3.6]
gtk_tool_shell_get_text_orientation [Gtk3 3.6]	gtk_tool_shell_get_text_size_group [Gtk3 3.6]

gtk_tool_shell_rebuild_menu [Gtk3 3.6]	gtk_toolbar_get_drop_index [Gtk3 3.6]
gtk_toolbar_get_icon_size [Gtk3 3.6]	gtk_toolbar_get_item_index [Gtk3 3.6]
gtk_toolbar_get_n_items [Gtk3 3.6]	gtk_toolbar_get_nth_item [Gtk3 3.6]
gtk_toolbar_get_relief_style [Gtk3 3.6]	gtk_toolbar_get_show_arrow [Gtk3 3.6]
gtk_toolbar_get_style [Gtk3 3.6]	gtk_toolbar_insert [Gtk3 3.6]
gtk_toolbar_new [Gtk3 3.6]	gtk_toolbar_set_drop_highlight_item [Gtk3 3.6]
gtk_toolbar_set_icon_size [Gtk3 3.6]	gtk_toolbar_set_show_arrow [Gtk3 3.6]
gtk_toolbar_set_style [Gtk3 3.6]	gtk_toolbar_unset_icon_size [Gtk3 3.6]
gtk_toolbar_unset_style [Gtk3 3.6]	gtk_tooltip_set_custom [Gtk3 3.6]
gtk_tooltip_set_icon [Gtk3 3.6]	gtk_tooltip_set_icon_from_gicon [Gtk3 3.6]
gtk_tooltip_set_icon_from_icon_name [Gtk3 3.6]	gtk_tooltip_set_icon_from_stock [Gtk3 3.6]
gtk_tooltip_set_markup [Gtk3 3.6]	gtk_tooltip_set_text [Gtk3 3.6]
gtk_tooltip_set_tip_area [Gtk3 3.6]	gtk_tooltip_trigger_tooltip_query [Gtk3 3.6]
gtk_tree_drag_dest_drag_data_received [Gtk3 3.6]	gtk_tree_drag_dest_row_drop_possible [Gtk3 3.6]
gtk_tree_drag_source_drag_data_delete [Gtk3 3.6]	gtk_tree_drag_source_drag_data_get [Gtk3 3.6]
gtk_tree_drag_source_row_draggable [Gtk3 3.6]	gtk_tree_get_row_drag_data [Gtk3 3.6]
gtk_tree_iter_copy [Gtk3 3.6]	gtk_tree_iter_free [Gtk3 3.6]
gtk_tree_model_filter_clear_cache [Gtk3 3.6]	gtk_tree_model_filter_convert_child_iter_to_iter [Gtk3 3.6]
gtk_tree_model_filter_convert_child_path_to_path [Gtk3 3.6]	gtk_tree_model_filter_convert_iter_to_child_iter [Gtk3 3.6]
gtk_tree_model_filter_convert_path_to_child_path [Gtk3 3.6]	gtk_tree_model_filter_get_model [Gtk3 3.6]
gtk_tree_model_filter_new [Gtk3 3.6]	gtk_tree_model_filter_refilter [Gtk3 3.6]
gtk_tree_model_filter_set_modify_func [Gtk3 3.6]	gtk_tree_model_filter_set_visible_column [Gtk3 3.6]
gtk_tree_model_filter_set_visible_func [Gtk3 3.6]	gtk_tree_model_foreach [Gtk3 3.6]

gtk_tree_model_get [Gtk3 3.6]	gtk_tree_model_get_column_type [Gtk3 3.6]
gtk_tree_model_get_flags [Gtk3 3.6]	gtk_tree_model_get_iter [Gtk3 3.6]
gtk_tree_model_get_iter_first [Gtk3 3.6]	gtk_tree_model_get_iter_from_string [Gtk3 3.6]
gtk_tree_model_get_n_columns [Gtk3 3.6]	gtk_tree_model_get_path [Gtk3 3.6]
gtk_tree_model_get_string_from_iter [Gtk3 3.6]	gtk_tree_model_get_valist [Gtk3 3.6]
gtk_tree_model_get_value [Gtk3 3.6]	gtk_tree_model_iter_children [Gtk3 3.6]
gtk_tree_model_iter_has_child [Gtk3 3.6]	gtk_tree_model_iter_n_children [Gtk3 3.6]
gtk_tree_model_iter_next [Gtk3 3.6]	gtk_tree_model_iter_nth_child [Gtk3 3.6]
gtk_tree_model_iter_parent [Gtk3 3.6]	gtk_tree_model_iter_previous [Gtk3 3.6]
gtk_tree_model_ref_node [Gtk3 3.6]	gtk_tree_model_row_changed [Gtk3 3.6]
gtk_tree_model_row_deleted [Gtk3 3.6]	gtk_tree_model_row_has_child_toggled [Gtk3 3.6]
gtk_tree_model_row_inserted [Gtk3 3.6]	gtk_tree_model_rows_reordered [Gtk3 3.6]
gtk_tree_model_sort_clear_cache [Gtk3 3.6]	gtk_tree_model_sort_convert_child_iter_to_iter [Gtk3 3.6]
gtk_tree_model_sort_convert_child_path_to_path [Gtk3 3.6]	gtk_tree_model_sort_convert_iter_to_child_iter [Gtk3 3.6]
gtk_tree_model_sort_convert_path_to_child_path [Gtk3 3.6]	gtk_tree_model_sort_get_model [Gtk3 3.6]
gtk_tree_model_sort_iter_is_valid [Gtk3 3.6]	gtk_tree_model_sort_new_with_model [Gtk3 3.6]
gtk_tree_model_sort_reset_default_sort_func [Gtk3 3.6]	gtk_tree_model_unref_node [Gtk3 3.6]
gtk_tree_path_append_index [Gtk3 3.6]	gtk_tree_path_compare [Gtk3 3.6]
gtk_tree_path_copy [Gtk3 3.6]	gtk_tree_path_down [Gtk3 3.6]
gtk_tree_path_free [Gtk3 3.6]	gtk_tree_path_get_depth [Gtk3 3.6]
gtk_tree_path_get_indices [Gtk3 3.6]	gtk_tree_path_get_indices_with_depth [Gtk3 3.6]
gtk_tree_path_is_ancestor [Gtk3 3.6]	gtk_tree_path_is_descendant [Gtk3 3.6]

gtk_tree_path_new [Gtk3 3.6]	gtk_tree_path_new_first [Gtk3 3.6]
gtk_tree_path_new_from_indices [Gtk3 3.6]	gtk_tree_path_new_from_string [Gtk3 3.6]
gtk_tree_path_next [Gtk3 3.6]	gtk_tree_path_prepend_index [Gtk3 3.6]
gtk_tree_path_prev [Gtk3 3.6]	gtk_tree_path_to_string [Gtk3 3.6]
gtk_tree_path_up [Gtk3 3.6]	gtk_tree_row_reference_copy [Gtk3 3.6]
gtk_tree_row_reference_deleted [Gtk3 3.6]	gtk_tree_row_reference_free [Gtk3 3.6]
gtk_tree_row_reference_get_model [Gtk3 3.6]	gtk_tree_row_reference_get_path [Gtk3 3.6]
gtk_tree_row_reference_inserted [Gtk3 3.6]	gtk_tree_row_reference_new [Gtk3 3.6]
gtk_tree_row_reference_new_proxy [Gtk3 3.6]	gtk_tree_row_reference_reordered [Gtk3 3.6]
gtk_tree_row_reference_valid [Gtk3 3.6]	gtk_tree_selection_count_selected_rows [Gtk3 3.6]
gtk_tree_selection_get_mode [Gtk3 3.6]	gtk_tree_selection_get_select_function [Gtk3 3.6]
gtk_tree_selection_get_selected [Gtk3 3.6]	gtk_tree_selection_get_selected_rows [Gtk3 3.6]
gtk_tree_selection_get_tree_view [Gtk3 3.6]	gtk_tree_selection_get_user_data [Gtk3 3.6]
gtk_tree_selection_iter_is_selected [Gtk3 3.6]	gtk_tree_selection_path_is_selected [Gtk3 3.6]
gtk_tree_selection_select_all [Gtk3 3.6]	gtk_tree_selection_select_iter [Gtk3 3.6]
gtk_tree_selection_select_path [Gtk3 3.6]	gtk_tree_selection_select_range [Gtk3 3.6]
gtk_tree_selection_selected_foreach [Gtk3 3.6]	gtk_tree_selection_set_mode [Gtk3 3.6]
gtk_tree_selection_set_select_function [Gtk3 3.6]	gtk_tree_selection_unselect_all [Gtk3 3.6]
gtk_tree_selection_unselect_iter [Gtk3 3.6]	gtk_tree_selection_unselect_path [Gtk3 3.6]
gtk_tree_selection_unselect_range [Gtk3 3.6]	gtk_tree_set_row_drag_data [Gtk3 3.6]
gtk_tree_sortable_get_sort_column_id [Gtk3 3.6]	gtk_tree_sortable_has_default_sort_function [Gtk3 3.6]
gtk_tree_sortable_set_default_sort_function [Gtk3 3.6]	gtk_tree_sortable_set_sort_column_id [Gtk3 3.6]

gtk_tree_sortable_set_sort_func [Gtk3 3.6]	gtk_tree_sortable_sort_column_changed [Gtk3 3.6]
gtk_tree_store_append [Gtk3 3.6]	gtk_tree_store_clear [Gtk3 3.6]
gtk_tree_store_insert [Gtk3 3.6]	gtk_tree_store_insert_after [Gtk3 3.6]
gtk_tree_store_insert_before [Gtk3 3.6]	gtk_tree_store_insert_with_values [Gtk3 3.6]
gtk_tree_store_insert_with_valuesv [Gtk3 3.6]	gtk_tree_store_is_ancestor [Gtk3 3.6]
gtk_tree_store_iter_depth [Gtk3 3.6]	gtk_tree_store_iter_is_valid [Gtk3 3.6]
gtk_tree_store_move_after [Gtk3 3.6]	gtk_tree_store_move_before [Gtk3 3.6]
gtk_tree_store_new [Gtk3 3.6]	gtk_tree_store_newv [Gtk3 3.6]
gtk_tree_store_prepend [Gtk3 3.6]	gtk_tree_store_remove [Gtk3 3.6]
gtk_tree_store_reorder [Gtk3 3.6]	gtk_tree_store_set [Gtk3 3.6]
gtk_tree_store_set_column_types [Gtk3 3.6]	gtk_tree_store_set_valist [Gtk3 3.6]
gtk_tree_store_set_value [Gtk3 3.6]	gtk_tree_store_set_valuesv [Gtk3 3.6]
gtk_tree_store_swap [Gtk3 3.6]	gtk_tree_view_append_column [Gtk3 3.6]
gtk_tree_view_collapse_all [Gtk3 3.6]	gtk_tree_view_collapse_row [Gtk3 3.6]
gtk_tree_view_column_add_attribute [Gtk3 3.6]	gtk_tree_view_column_cell_get_position [Gtk3 3.6]
gtk_tree_view_column_cell_get_size [Gtk3 3.6]	gtk_tree_view_column_cell_is_visible [Gtk3 3.6]
gtk_tree_view_column_cell_set_cell_data [Gtk3 3.6]	gtk_tree_view_column_clear [Gtk3 3.6]
gtk_tree_view_column_clear_attributes [Gtk3 3.6]	gtk_tree_view_column_clicked [Gtk3 3.6]
gtk_tree_view_column_focus_cell [Gtk3 3.6]	gtk_tree_view_column_get_alignment [Gtk3 3.6]
gtk_tree_view_column_get_button [Gtk3 3.6]	gtk_tree_view_column_get_clickable [Gtk3 3.6]
gtk_tree_view_column_get_expand [Gtk3 3.6]	gtk_tree_view_column_get_fixed_width [Gtk3 3.6]
gtk_tree_view_column_get_max_width [Gtk3 3.6]	gtk_tree_view_column_get_min_width [Gtk3 3.6]
gtk_tree_view_column_get_reorderable [Gtk3 3.6]	gtk_tree_view_column_get_resizable [Gtk3 3.6]

gtk_tree_view_column_get_sizing [Gtk3 3.6]	gtk_tree_view_column_get_sort_column_id [Gtk3 3.6]
gtk_tree_view_column_get_sort_indicator [Gtk3 3.6]	gtk_tree_view_column_get_sort_order [Gtk3 3.6]
gtk_tree_view_column_get_spacing [Gtk3 3.6]	gtk_tree_view_column_get_title [Gtk3 3.6]
gtk_tree_view_column_get_tree_view [Gtk3 3.6]	gtk_tree_view_column_get_visible [Gtk3 3.6]
gtk_tree_view_column_get_widget [Gtk3 3.6]	gtk_tree_view_column_get_width [Gtk3 3.6]
gtk_tree_view_column_get_x_offset [Gtk3 3.6]	gtk_tree_view_column_new [Gtk3 3.6]
gtk_tree_view_column_new_with_area [Gtk3 3.6]	gtk_tree_view_column_new_with_attributes [Gtk3 3.6]
gtk_tree_view_column_pack_end [Gtk3 3.6]	gtk_tree_view_column_pack_start [Gtk3 3.6]
gtk_tree_view_column_queue_resize [Gtk3 3.6]	gtk_tree_view_column_set_alignment [Gtk3 3.6]
gtk_tree_view_column_set_attributes [Gtk3 3.6]	gtk_tree_view_column_set_cell_data_func [Gtk3 3.6]
gtk_tree_view_column_set_clickable [Gtk3 3.6]	gtk_tree_view_column_set_expand [Gtk3 3.6]
gtk_tree_view_column_set_fixed_width [Gtk3 3.6]	gtk_tree_view_column_set_max_width [Gtk3 3.6]
gtk_tree_view_column_set_min_width [Gtk3 3.6]	gtk_tree_view_column_set_reorderable [Gtk3 3.6]
gtk_tree_view_column_set_resizable [Gtk3 3.6]	gtk_tree_view_column_set_sizing [Gtk3 3.6]
gtk_tree_view_column_set_sort_column_id [Gtk3 3.6]	gtk_tree_view_column_set_sort_indicator [Gtk3 3.6]
gtk_tree_view_column_set_sort_order [Gtk3 3.6]	gtk_tree_view_column_set_spacing [Gtk3 3.6]
gtk_tree_view_column_set_title [Gtk3 3.6]	gtk_tree_view_column_set_visible [Gtk3 3.6]
gtk_tree_view_column_set_widget [Gtk3 3.6]	gtk_tree_view_columns_autosize [Gtk3 3.6]
gtk_tree_view_convert_bin_window_to_tree_coords [Gtk3 3.6]	gtk_tree_view_convert_bin_window_to_widget_coords [Gtk3 3.6]
gtk_tree_view_convert_tree_to_bin_window_coords [Gtk3 3.6]	gtk_tree_view_convert_tree_to_widget_coords [Gtk3 3.6]
gtk_tree_view_convert_widget_to_bin_window_coords [Gtk3 3.6]	gtk_tree_view_convert_widget_to_tree_coords [Gtk3 3.6]

gtk_tree_view_create_row_drag_icon [Gtk3 3.6]	gtk_tree_view_enable_model_drag_dest [Gtk3 3.6]
gtk_tree_view_enable_model_drag_source [Gtk3 3.6]	gtk_tree_view_expand_all [Gtk3 3.6]
gtk_tree_view_expand_row [Gtk3 3.6]	gtk_tree_view_expand_to_path [Gtk3 3.6]
gtk_tree_view_get_background_area [Gtk3 3.6]	gtk_tree_view_get_bin_window [Gtk3 3.6]
gtk_tree_view_get_cell_area [Gtk3 3.6]	gtk_tree_view_get_column [Gtk3 3.6]
gtk_tree_view_get_columns [Gtk3 3.6]	gtk_tree_view_get_cursor [Gtk3 3.6]
gtk_tree_view_get_dest_row_at_pos [Gtk3 3.6]	gtk_tree_view_get_drag_dest_row [Gtk3 3.6]
gtk_tree_view_get_enable_search [Gtk3 3.6]	gtk_tree_view_get_enable_tree_lines [Gtk3 3.6]
gtk_tree_view_get_expander_column [Gtk3 3.6]	gtk_tree_view_get_fixed_height_mode [Gtk3 3.6]
gtk_tree_view_get_grid_lines [Gtk3 3.6]	gtk_tree_view_get_hadjustment [Gtk3 3.6]
gtk_tree_view_get_headers_clickable [Gtk3 3.6]	gtk_tree_view_get_headers_visible [Gtk3 3.6]
gtk_tree_view_get_hover_expand [Gtk3 3.6]	gtk_tree_view_get_hover_selection [Gtk3 3.6]
gtk_tree_view_get_level_indentation [Gtk3 3.6]	gtk_tree_view_get_model [Gtk3 3.6]
gtk_tree_view_get_n_columns [Gtk3 3.6]	gtk_tree_view_get_path_at_pos [Gtk3 3.6]
gtk_tree_view_get_reorderable [Gtk3 3.6]	gtk_tree_view_get_row_separator_func [Gtk3 3.6]
gtk_tree_view_get_rubber_banding [Gtk3 3.6]	gtk_tree_view_get_rules_hint [Gtk3 3.6]
gtk_tree_view_get_search_column [Gtk3 3.6]	gtk_tree_view_get_search_entry [Gtk3 3.6]
gtk_tree_view_get_search_equal_func [Gtk3 3.6]	gtk_tree_view_get_search_position_func [Gtk3 3.6]
gtk_tree_view_get_selection [Gtk3 3.6]	gtk_tree_view_get_show_expanders [Gtk3 3.6]
gtk_tree_view_get_tooltip_column [Gtk3 3.6]	gtk_tree_view_get_tooltip_context [Gtk3 3.6]
gtk_tree_view_get_vadjustment [Gtk3 3.6]	gtk_tree_view_get_visible_range [Gtk3 3.6]

gtk_tree_view_get_visible_rect [Gtk3 3.6]	gtk_tree_view_insert_column [Gtk3 3.6]
gtk_tree_view_insert_column_with_attributes [Gtk3 3.6]	gtk_tree_view_insert_column_with_data_func [Gtk3 3.6]
gtk_tree_view_is_blank_at_pos [Gtk3 3.6]	gtk_tree_view_is_rubber_banding_active [Gtk3 3.6]
gtk_tree_view_map_expanded_rows [Gtk3 3.6]	gtk_tree_view_move_column_after [Gtk3 3.6]
gtk_tree_view_new [Gtk3 3.6]	gtk_tree_view_new_with_model [Gtk3 3.6]
gtk_tree_view_remove_column [Gtk3 3.6]	gtk_tree_view_row_activated [Gtk3 3.6]
gtk_tree_view_row_expanded [Gtk3 3.6]	gtk_tree_view_scroll_to_cell [Gtk3 3.6]
gtk_tree_view_scroll_to_point [Gtk3 3.6]	gtk_tree_view_set_column_drag_function [Gtk3 3.6]
gtk_tree_view_set_cursor [Gtk3 3.6]	gtk_tree_view_set_cursor_on_cell [Gtk3 3.6]
gtk_tree_view_set_destroy_count_func [Gtk3 3.6]	gtk_tree_view_set_drag_dest_row [Gtk3 3.6]
gtk_tree_view_set_enable_search [Gtk3 3.6]	gtk_tree_view_set_enable_tree_lines [Gtk3 3.6]
gtk_tree_view_set_expander_column [Gtk3 3.6]	gtk_tree_view_set_fixed_height_mode [Gtk3 3.6]
gtk_tree_view_set_grid_lines [Gtk3 3.6]	gtk_tree_view_set_hadjustment [Gtk3 3.6]
gtk_tree_view_set_headers_clickable [Gtk3 3.6]	gtk_tree_view_set_headers_visible [Gtk3 3.6]
gtk_tree_view_set_hover_expand [Gtk3 3.6]	gtk_tree_view_set_hover_selection [Gtk3 3.6]
gtk_tree_view_set_level_indentation [Gtk3 3.6]	gtk_tree_view_set_model [Gtk3 3.6]
gtk_tree_view_set_reorderable [Gtk3 3.6]	gtk_tree_view_set_row_separator_func [Gtk3 3.6]
gtk_tree_view_set_rubber_banding [Gtk3 3.6]	gtk_tree_view_set_rules_hint [Gtk3 3.6]
gtk_tree_view_set_search_column [Gtk3 3.6]	gtk_tree_view_set_search_entry [Gtk3 3.6]
gtk_tree_view_set_search_equal_func [Gtk3 3.6]	gtk_tree_view_set_search_position_func [Gtk3 3.6]
gtk_tree_view_set_show_expanders [Gtk3 3.6]	gtk_tree_view_set_tooltip_cell [Gtk3 3.6]

gtk_tree_view_set_tooltip_column [Gtk3 3.6]	gtk_tree_view_set_tooltip_row [Gtk3 3.6]
gtk_tree_view_set_vadjustment [Gtk3 3.6]	gtk_tree_view_unset_rows_drag_des t [Gtk3 3.6]
gtk_tree_view_unset_rows_drag_sou rce [Gtk3 3.6]	gtk_true [Gtk3 3.6]
gtk_ui_manager_add_ui [Gtk3 3.6]	gtk_ui_manager_add_ui_from_file [Gtk3 3.6]
gtk_ui_manager_add_ui_from_resou rce [Gtk3 3.6]	gtk_ui_manager_add_ui_from_string [Gtk3 3.6]
gtk_ui_manager_ensure_update [Gtk3 3.6]	gtk_ui_manager_get_accel_group [Gtk3 3.6]
gtk_ui_manager_get_action [Gtk3 3.6]	gtk_ui_manager_get_action_groups [Gtk3 3.6]
gtk_ui_manager_get_add_tearoffs [Gtk3 3.6]	gtk_ui_manager_get_toplevels [Gtk3 3.6]
gtk_ui_manager_get_ui [Gtk3 3.6]	gtk_ui_manager_get_widget [Gtk3 3.6]
gtk_ui_manager_insert_action_group [Gtk3 3.6]	gtk_ui_manager_new [Gtk3 3.6]
gtk_ui_manager_new_merge_id [Gtk3 3.6]	gtk_ui_manager_remove_action_gro up [Gtk3 3.6]
gtk_ui_manager_remove_ui [Gtk3 3.6]	gtk_ui_manager_set_add_tearoffs [Gtk3 3.6]
gtk_viewport_get_bin_window [Gtk3 3.6]	gtk_viewport_get_hadjustment [Gtk3 3.6]
gtk_viewport_get_shadow_type [Gtk3 3.6]	gtk_viewport_get_vadjustment [Gtk3 3.6]
gtk_viewport_get_view_window [Gtk3 3.6]	gtk_viewport_new [Gtk3 3.6]
gtk_viewport_set_hadjustment [Gtk3 3.6]	gtk_viewport_set_shadow_type [Gtk3 3.6]
gtk_viewport_set_vadjustment [Gtk3 3.6]	gtk_volume_button_new [Gtk3 3.6]
gtk_widget_activate [Gtk3 3.6]	gtk_widget_add_accelerator [Gtk3 3.6]
gtk_widget_add_device_events [Gtk3 3.6]	gtk_widget_add_events [Gtk3 3.6]
gtk_widget_add_mnemonic_label [Gtk3 3.6]	gtk_widget_can_activate_accel [Gtk3 3.6]
gtk_widget_child_focus [Gtk3 3.6]	gtk_widget_child_notify [Gtk3 3.6]

gtk_widget_class_find_style_property [Gtk3 3.6]	gtk_widget_class_install_style_property [Gtk3 3.6]
gtk_widget_class_install_style_property_parser [Gtk3 3.6]	gtk_widget_class_list_style_properties [Gtk3 3.6]
gtk_widget_class_set_accessible_role [Gtk3 3.6]	gtk_widget_class_set_accessible_type [Gtk3 3.6]
gtk_widget_compute_expand [Gtk3 3.6]	gtk_widget_create_pango_context [Gtk3 3.6]
gtk_widget_create_pango_layout [Gtk3 3.6]	gtk_widget_destroy [Gtk3 3.6]
gtk_widget_destroyed [Gtk3 3.6]	gtk_widget_device_is_shadowed [Gtk3 3.6]
gtk_widget_draw [Gtk3 3.6]	gtk_widget_error_bell [Gtk3 3.6]
gtk_widget_event [Gtk3 3.6]	gtk_widget_freeze_child_notify [Gtk3 3.6]
gtk_widget_get_accessible [Gtk3 3.6]	gtk_widget_get_allocated_height [Gtk3 3.6]
gtk_widget_get_allocated_width [Gtk3 3.6]	gtk_widget_get_allocation [Gtk3 3.6]
gtk_widget_get_ancestor [Gtk3 3.6]	gtk_widget_get_app_paintable [Gtk3 3.6]
gtk_widget_get_can_default [Gtk3 3.6]	gtk_widget_get_can_focus [Gtk3 3.6]
gtk_widget_get_child_requisition [Gtk3 3.6]	gtk_widget_get_child_visible [Gtk3 3.6]
gtk_widget_get_clipboard [Gtk3 3.6]	gtk_widget_get_composite_name [Gtk3 3.6]
gtk_widget_get_default_direction [Gtk3 3.6]	gtk_widget_get_device_enabled [Gtk3 3.6]
gtk_widget_get_device_events [Gtk3 3.6]	gtk_widget_get_direction [Gtk3 3.6]
gtk_widget_get_display [Gtk3 3.6]	gtk_widget_get_double_buffered [Gtk3 3.6]
gtk_widget_get_events [Gtk3 3.6]	gtk_widget_get_halign [Gtk3 3.6]
gtk_widget_get_has_tooltip [Gtk3 3.6]	gtk_widget_get_has_window [Gtk3 3.6]
gtk_widget_get_hexpand [Gtk3 3.6]	gtk_widget_get_hexpand_set [Gtk3 3.6]
gtk_widget_get_mapped [Gtk3 3.6]	gtk_widget_get_margin_bottom [Gtk3 3.6]
gtk_widget_get_margin_left [Gtk3 3.6]	gtk_widget_get_margin_right [Gtk3 3.6]

gtk_widget_get_margin_top [Gtk3 3.6]	gtk_widget_get_modifier_mask [Gtk3 3.6]
gtk_widget_get_name [Gtk3 3.6]	gtk_widget_get_no_show_all [Gtk3 3.6]
gtk_widget_get_pango_context [Gtk3 3.6]	gtk_widget_get_parent [Gtk3 3.6]
gtk_widget_get_parent_window [Gtk3 3.6]	gtk_widget_get_path [Gtk3 3.6]
gtk_widget_get_pointer [Gtk3 3.6]	gtk_widget_get_preferred_height [Gtk3 3.6]
gtk_widget_get_preferred_height_for_width [Gtk3 3.6]	gtk_widget_get_preferred_size [Gtk3 3.6]
gtk_widget_get_preferred_width [Gtk3 3.6]	gtk_widget_get_preferred_width_for_height [Gtk3 3.6]
gtk_widget_get_realized [Gtk3 3.6]	gtk_widget_get_receives_default [Gtk3 3.6]
gtk_widget_get_request_mode [Gtk3 3.6]	gtk_widget_get_requisition [Gtk3 3.6]
gtk_widget_get_root_window [Gtk3 3.6]	gtk_widget_get_screen [Gtk3 3.6]
gtk_widget_get_sensitive [Gtk3 3.6]	gtk_widget_get_settings [Gtk3 3.6]
gtk_widget_get_size_request [Gtk3 3.6]	gtk_widget_get_state [Gtk3 3.6]
gtk_widget_get_state_flags [Gtk3 3.6]	gtk_widget_get_style_context [Gtk3 3.6]
gtk_widget_get_support_multidevice [Gtk3 3.6]	gtk_widget_get_tooltip_markup [Gtk3 3.6]
gtk_widget_get_tooltip_text [Gtk3 3.6]	gtk_widget_get_tooltip_window [Gtk3 3.6]
gtk_widget_get_toplevel [Gtk3 3.6]	gtk_widget_get_valign [Gtk3 3.6]
gtk_widget_get_vexpand [Gtk3 3.6]	gtk_widget_get_vexpand_set [Gtk3 3.6]
gtk_widget_get_visible [Gtk3 3.6]	gtk_widget_get_visual [Gtk3 3.6]
gtk_widget_get_window [Gtk3 3.6]	gtk_widget_grab_default [Gtk3 3.6]
gtk_widget_grab_focus [Gtk3 3.6]	gtk_widget_has_default [Gtk3 3.6]
gtk_widget_has_focus [Gtk3 3.6]	gtk_widget_has_grab [Gtk3 3.6]
gtk_widget_has_screen [Gtk3 3.6]	gtk_widget_has_visible_focus [Gtk3 3.6]
gtk_widget_hide [Gtk3 3.6]	gtk_widget_hide_on_delete [Gtk3 3.6]

gtk_widget_in_destruction [Gtk3 3.6]	gtk_widget_input_shape_combine_region [Gtk3 3.6]
gtk_widget_insert_action_group [Gtk3 3.6]	gtk_widget_intersect [Gtk3 3.6]
gtk_widget_is_ancestor [Gtk3 3.6]	gtk_widget_is_composited [Gtk3 3.6]
gtk_widget_is_drawable [Gtk3 3.6]	gtk_widget_is_focus [Gtk3 3.6]
gtk_widget_is_sensitive [Gtk3 3.6]	gtk_widget_is_toplevel [Gtk3 3.6]
gtk_widget_keynav_failed [Gtk3 3.6]	gtk_widget_list_accel_closures [Gtk3 3.6]
gtk_widget_list_mnemonic_labels [Gtk3 3.6]	gtk_widget_map [Gtk3 3.6]
gtk_widget_mnemonic_activate [Gtk3 3.6]	gtk_widget_new [Gtk3 3.6]
gtk_widget_override_background_color [Gtk3 3.6]	gtk_widget_override_color [Gtk3 3.6]
gtk_widget_override_cursor [Gtk3 3.6]	gtk_widget_override_font [Gtk3 3.6]
gtk_widget_override_symbolic_color [Gtk3 3.6]	gtk_widget_path_append_for_widget [Gtk3 3.6]
gtk_widget_path_append_type [Gtk3 3.6]	gtk_widget_path_append_with_siblings [Gtk3 3.6]
gtk_widget_path_copy [Gtk3 3.6]	gtk_widget_path_free [Gtk3 3.6]
gtk_widget_path_get_object_type [Gtk3 3.6]	gtk_widget_path_has_parent [Gtk3 3.6]
gtk_widget_path_is_type [Gtk3 3.6]	gtk_widget_path_iter_add_class [Gtk3 3.6]
gtk_widget_path_iter_add_region [Gtk3 3.6]	gtk_widget_path_iter_clear_classes [Gtk3 3.6]
gtk_widget_path_iter_clear_regions [Gtk3 3.6]	gtk_widget_path_iter_get_name [Gtk3 3.6]
gtk_widget_path_iter_get_object_type [Gtk3 3.6]	gtk_widget_path_iter_get_sibling_index [Gtk3 3.6]
gtk_widget_path_iter_get_siblings [Gtk3 3.6]	gtk_widget_path_iter_has_class [Gtk3 3.6]
gtk_widget_path_iter_has_name [Gtk3 3.6]	gtk_widget_path_iter_has_qclass [Gtk3 3.6]
gtk_widget_path_iter_has_qname [Gtk3 3.6]	gtk_widget_path_iter_has_qregion [Gtk3 3.6]
gtk_widget_path_iter_has_region [Gtk3 3.6]	gtk_widget_path_iter_list_classes [Gtk3 3.6]

gtk_widget_path_iter_list_regions [Gtk3 3.6]	gtk_widget_path_iter_remove_class [Gtk3 3.6]
gtk_widget_path_iter_remove_region [Gtk3 3.6]	gtk_widget_path_iter_set_name [Gtk3 3.6]
gtk_widget_path_iter_set_object_type [Gtk3 3.6]	gtk_widget_path_length [Gtk3 3.6]
gtk_widget_path_new [Gtk3 3.6]	gtk_widget_path_prepend_type [Gtk3 3.6]
gtk_widget_path_ref [Gtk3 3.6]	gtk_widget_path_to_string [Gtk3 3.6]
gtk_widget_path_unref [Gtk3 3.6]	gtk_widget_pop_composite_child [Gtk3 3.6]
gtk_widget_push_composite_child [Gtk3 3.6]	gtk_widget_queue_compute_expand [Gtk3 3.6]
gtk_widget_queue_draw [Gtk3 3.6]	gtk_widget_queue_draw_area [Gtk3 3.6]
gtk_widget_queue_draw_region [Gtk3 3.6]	gtk_widget_queue_resize [Gtk3 3.6]
gtk_widget_queue_resize_no_redraw [Gtk3 3.6]	gtk_widget_realize [Gtk3 3.6]
gtk_widget_region_intersect [Gtk3 3.6]	gtk_widget_remove_accelerator [Gtk3 3.6]
gtk_widget_remove_mnemonic_label [Gtk3 3.6]	gtk_widget_render_icon_pixbuf [Gtk3 3.6]
gtk_widget_reparent [Gtk3 3.6]	gtk_widget_reset_style [Gtk3 3.6]
gtk_widget_send_expose [Gtk3 3.6]	gtk_widget_send_focus_change [Gtk3 3.6]
gtk_widget_set_accel_path [Gtk3 3.6]	gtk_widget_set_allocation [Gtk3 3.6]
gtk_widget_set_app_paintable [Gtk3 3.6]	gtk_widget_set_can_default [Gtk3 3.6]
gtk_widget_set_can_focus [Gtk3 3.6]	gtk_widget_set_child_visible [Gtk3 3.6]
gtk_widget_set_composite_name [Gtk3 3.6]	gtk_widget_set_default_direction [Gtk3 3.6]
gtk_widget_set_device_enabled [Gtk3 3.6]	gtk_widget_set_device_events [Gtk3 3.6]
gtk_widget_set_direction [Gtk3 3.6]	gtk_widget_set_double_buffered [Gtk3 3.6]
gtk_widget_set_events [Gtk3 3.6]	gtk_widget_set_halign [Gtk3 3.6]
gtk_widget_set_has_tooltip [Gtk3 3.6]	gtk_widget_set_has_window [Gtk3 3.6]

gtk_widget_set_hexpand [Gtk3 3.6]	gtk_widget_set_hexpand_set [Gtk3 3.6]
gtk_widget_set_mapped [Gtk3 3.6]	gtk_widget_set_margin_bottom [Gtk3 3.6]
gtk_widget_set_margin_left [Gtk3 3.6]	gtk_widget_set_margin_right [Gtk3 3.6]
gtk_widget_set_margin_top [Gtk3 3.6]	gtk_widget_set_name [Gtk3 3.6]
gtk_widget_set_no_show_all [Gtk3 3.6]	gtk_widget_set_parent [Gtk3 3.6]
gtk_widget_set_parent_window [Gtk3 3.6]	gtk_widget_set_realized [Gtk3 3.6]
gtk_widget_set_receives_default [Gtk3 3.6]	gtk_widget_set_redraw_on_allocate [Gtk3 3.6]
gtk_widget_set_sensitive [Gtk3 3.6]	gtk_widget_set_size_request [Gtk3 3.6]
gtk_widget_set_state [Gtk3 3.6]	gtk_widget_set_state_flags [Gtk3 3.6]
gtk_widget_set_support_multidevice [Gtk3 3.6]	gtk_widget_set_tooltip_markup [Gtk3 3.6]
gtk_widget_set_tooltip_text [Gtk3 3.6]	gtk_widget_set_tooltip_window [Gtk3 3.6]
gtk_widget_set_valign [Gtk3 3.6]	gtk_widget_set_vexpand [Gtk3 3.6]
gtk_widget_set_vexpand_set [Gtk3 3.6]	gtk_widget_set_visible [Gtk3 3.6]
gtk_widget_set_visual [Gtk3 3.6]	gtk_widget_set_window [Gtk3 3.6]
gtk_widget_shape_combine_region [Gtk3 3.6]	gtk_widget_show [Gtk3 3.6]
gtk_widget_show_all [Gtk3 3.6]	gtk_widget_show_now [Gtk3 3.6]
gtk_widget_size_allocate [Gtk3 3.6]	gtk_widget_size_request [Gtk3 3.6]
gtk_widget_style_get [Gtk3 3.6]	gtk_widget_style_get_property [Gtk3 3.6]
gtk_widget_style_get_valist [Gtk3 3.6]	gtk_widget_thaw_child_notify [Gtk3 3.6]
gtk_widget_translate_coordinates [Gtk3 3.6]	gtk_widget_trigger_tooltip_query [Gtk3 3.6]
gtk_widget_unmap [Gtk3 3.6]	gtk_widget_unparent [Gtk3 3.6]
gtk_widget_unrealize [Gtk3 3.6]	gtk_widget_unset_state_flags [Gtk3 3.6]
gtk_window_activate_default [Gtk3 3.6]	gtk_window_activate_focus [Gtk3 3.6]

gtk_window_activate_key [Gtk3 3.6]	gtk_window_add_accel_group [Gtk3 3.6]
gtk_window_add_mnemonic [Gtk3 3.6]	gtk_window_begin_move_drag [Gtk3 3.6]
gtk_window_begin_resize_drag [Gtk3 3.6]	gtk_window_deiconify [Gtk3 3.6]
gtk_window_fullscreen [Gtk3 3.6]	gtk_window_get_accept_focus [Gtk3 3.6]
gtk_window_get_application [Gtk3 3.6]	gtk_window_get_attached_to [Gtk3 3.6]
gtk_window_get_decorated [Gtk3 3.6]	gtk_window_get_default_icon_list [Gtk3 3.6]
gtk_window_get_default_icon_name [Gtk3 3.6]	gtk_window_get_default_size [Gtk3 3.6]
gtk_window_get_default_widget [Gtk3 3.6]	gtk_window_get_deletable [Gtk3 3.6]
gtk_window_get_destroy_with_parent [Gtk3 3.6]	gtk_window_get_focus [Gtk3 3.6]
gtk_window_get_focus_on_map [Gtk3 3.6]	gtk_window_get_focus_visible [Gtk3 3.6]
gtk_window_get_gravity [Gtk3 3.6]	gtk_window_get_group [Gtk3 3.6]
gtk_window_get_has_resize_grip [Gtk3 3.6]	gtk_window_get_hide_titlebar_when_maximized [Gtk3 3.6]
gtk_window_get_icon [Gtk3 3.6]	gtk_window_get_icon_list [Gtk3 3.6]
gtk_window_get_icon_name [Gtk3 3.6]	gtk_window_get_mnemonic_modifier [Gtk3 3.6]
gtk_window_get_mnemonics_visible [Gtk3 3.6]	gtk_window_get_modal [Gtk3 3.6]
gtk_window_get_opacity [Gtk3 3.6]	gtk_window_get_position [Gtk3 3.6]
gtk_window_get_resizable [Gtk3 3.6]	gtk_window_get_resize_grip_area [Gtk3 3.6]
gtk_window_get_role [Gtk3 3.6]	gtk_window_get_screen [Gtk3 3.6]
gtk_window_get_size [Gtk3 3.6]	gtk_window_get_skip_pager_hint [Gtk3 3.6]
gtk_window_get_skip_taskbar_hint [Gtk3 3.6]	gtk_window_get_title [Gtk3 3.6]
gtk_window_get_transient_for [Gtk3 3.6]	gtk_window_get_type_hint [Gtk3 3.6]
gtk_window_get_urgency_hint [Gtk3 3.6]	gtk_window_get_window_type [Gtk3 3.6]

gtk_window_group_add_window [Gtk3 3.6]	gtk_window_group_get_current_device_grab [Gtk3 3.6]
gtk_window_group_get_current_grab [Gtk3 3.6]	gtk_window_group_list_windows [Gtk3 3.6]
gtk_window_group_new [Gtk3 3.6]	gtk_window_group_remove_window [Gtk3 3.6]
gtk_window_has_group [Gtk3 3.6]	gtk_window_has_toplevel_focus [Gtk3 3.6]
gtk_window_iconify [Gtk3 3.6]	gtk_window_is_active [Gtk3 3.6]
gtk_window_list_toplevels [Gtk3 3.6]	gtk_window_maximize [Gtk3 3.6]
gtk_window_mnemonic_activate [Gtk3 3.6]	gtk_window_move [Gtk3 3.6]
gtk_window_new [Gtk3 3.6]	gtk_window_parse_geometry [Gtk3 3.6]
gtk_window_present [Gtk3 3.6]	gtk_window_present_with_time [Gtk3 3.6]
gtk_window_propagate_key_event [Gtk3 3.6]	gtk_window_remove_accel_group [Gtk3 3.6]
gtk_window_remove_mnemonic [Gtk3 3.6]	gtk_window_reshow_with_initial_size [Gtk3 3.6]
gtk_window_resize [Gtk3 3.6]	gtk_window_resize_grip_is_visible [Gtk3 3.6]
gtk_window_resize_to_geometry [Gtk3 3.6]	gtk_window_set_accept_focus [Gtk3 3.6]
gtk_window_set_application [Gtk3 3.6]	gtk_window_set_attached_to [Gtk3 3.6]
gtk_window_set_auto_startup_notification [Gtk3 3.6]	gtk_window_set_decorated [Gtk3 3.6]
gtk_window_set_default [Gtk3 3.6]	gtk_window_set_default_geometry [Gtk3 3.6]
gtk_window_set_default_icon [Gtk3 3.6]	gtk_window_set_default_icon_from_file [Gtk3 3.6]
gtk_window_set_default_icon_list [Gtk3 3.6]	gtk_window_set_default_icon_name [Gtk3 3.6]
gtk_window_set_default_size [Gtk3 3.6]	gtk_window_set_deletable [Gtk3 3.6]
gtk_window_set_destroy_with_parent [Gtk3 3.6]	gtk_window_set_focus [Gtk3 3.6]
gtk_window_set_focus_on_map [Gtk3 3.6]	gtk_window_set_focus_visible [Gtk3 3.6]
gtk_window_set_geometry_hints [Gtk3 3.6]	gtk_window_set_gravity [Gtk3 3.6]

gtk_window_set_has_resize_grip [Gtk3 3.6]	gtk_window_set_has_user_ref_count [Gtk3 3.6]
gtk_window_set_hide_titlebar_when_maximized [Gtk3 3.6]	gtk_window_set_icon [Gtk3 3.6]
gtk_window_set_icon_from_file [Gtk3 3.6]	gtk_window_set_icon_list [Gtk3 3.6]
gtk_window_set_icon_name [Gtk3 3.6]	gtk_window_set_keep_above [Gtk3 3.6]
gtk_window_set_keep_below [Gtk3 3.6]	gtk_window_set_mnemonic_modifier [Gtk3 3.6]
gtk_window_set_mnemonics_visible [Gtk3 3.6]	gtk_window_set_modal [Gtk3 3.6]
gtk_window_set_opacity [Gtk3 3.6]	gtk_window_set_position [Gtk3 3.6]
gtk_window_set_resizable [Gtk3 3.6]	gtk_window_set_role [Gtk3 3.6]
gtk_window_set_screen [Gtk3 3.6]	gtk_window_set_skip_pager_hint [Gtk3 3.6]
gtk_window_set_skip_taskbar_hint [Gtk3 3.6]	gtk_window_set_startup_id [Gtk3 3.6]
gtk_window_set_title [Gtk3 3.6]	gtk_window_set_transient_for [Gtk3 3.6]
gtk_window_set_type_hint [Gtk3 3.6]	gtk_window_set_urgency_hint [Gtk3 3.6]
gtk_window_set_wmclass [Gtk3 3.6]	gtk_window_stick [Gtk3 3.6]
gtk_window_unfullscreen [Gtk3 3.6]	gtk_window_unmaximize [Gtk3 3.6]
gtk_window_unstick [Gtk3 3.6]	

7.5 Data Definitions for libgtk-3

This section defines global identifiers and their values that are associated with interfaces contained in libgtk-3. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

7.5.1 gtk-3.0/gtk/gtk.h

```

#define _GTKX_H_INSIDE_
#define _GTK_ABOUT_DIALOG_H_
#define _GTK_ACCEL_GROUP_H_
#define _GTK_ACCEL_LABEL_H_
#define _GTK_ACCEL_MAP_H_
#define _GTK_ACCESSIBLE_H_
#define _GTK_ACTIONABLE_H_
#define _GTK_ACTION_GROUP_H_
#define _GTK_ACTION_H_
#define _GTK_ACTIVATABLE_H_
#define _GTK_ADJUSTMENT_H_
#define _GTK_ALIGNMENT_H_
#define _GTK_APPLICATION_H_
#define _GTK_APPLICATION_WINDOW_H_
#define _GTK_APP_CHOOSER_BUTTON_H_
#define _GTK_APP_CHOOSER_DIALOG_H_
#define _GTK_APP_CHOOSER_H_
#define _GTK_APP_CHOOSER_WIDGET_H_
#define _GTK_ARROW_H_
#define _GTK_ASPECT_FRAME_H_
#define _GTK_BINDINGS_H_
#define _GTK_BIN_H_
#define _GTK_BORDER_H_
#define _GTK_BOX_H_
#define _GTK_BUILDABLE_H_
#define _GTK_BUILDER_H_
#define _GTK_BUTTON_BOX_H_
#define _GTK_BUTTON_H_
#define _GTK_CALENDAR_H_
#define _GTK_CELL_AREA_BOX_H_
#define _GTK_CELL_AREA_CONTEXT_H_
#define _GTK_CELL_AREA_H_
#define _GTK_CELL_EDITABLE_H_
#define _GTK_CELL_LAYOUT_H_
#define _GTK_CELL_RENDERER_ACCEL_H_
#define _GTK_CELL_RENDERER_COMBO_H_
#define _GTK_CELL_RENDERER_H_
#define _GTK_CELL_RENDERER_PIXBUF_H_
#define _GTK_CELL_RENDERER_PROGRESS_H_
#define _GTK_CELL_RENDERER_SPINNER_H_
#define _GTK_CELL_RENDERER_TEXT_H_
#define _GTK_CELL_RENDERER_TOGGLE_H_
#define _GTK_CELL_VIEW_H_
#define _GTK_CHECK_BUTTON_H_
#define _GTK_CHECK_MENU_ITEM_H_
#define _GTK_CLIPBOARD_H_
#define _GTK_COLOR_BUTTON_H_
#define _GTK_COLOR_CHOOSER_DIALOG_H_
#define _GTK_COLOR_CHOOSER_H_
#define _GTK_COLOR_CHOOSER_WIDGET_H_
#define _GTK_COLOR_UTILS_H_
#define _GTK_COMBO_BOX_H_
#define _GTK_COMBO_BOX_TEXT_H_
#define _GTK_CONTAINER_H_
#define _GTK_CSS_PROVIDER_H_
#define _GTK_CSS_SECTION_H_
#define _GTK_DEBUG_H_
#define _GTK_DIALOG_H_
#define _GTK_DND_H_
#define _GTK_DRAWING_AREA_H_
#define _GTK_EDITABLE_H_
#define _GTK_ENTRY_BUFFER_H_
#define _GTK_ENTRY_COMPLETION_H_
#define _GTK_ENTRY_H_
#define _GTK_ENUMS_H_
#define _GTK_EVENT_BOX_H_

```

```

#define __GTK_EXPANDER_H__
#define __GTK_FILE_CHOOSER_BUTTON_H__
#define __GTK_FILE_CHOOSER_DIALOG_H__
#define __GTK_FILE_CHOOSER_H__
#define __GTK_FILE_CHOOSER_WIDGET_H__
#define __GTK_FILE_FILTER_H__
#define __GTK_FIXED_H__
#define __GTK_FONT_BUTTON_H__
#define __GTK_FONT_CHOOSER_DIALOG_H__
#define __GTK_FONT_CHOOSER_H__
#define __GTK_FONT_CHOOSER_WIDGET_H__
#define __GTK_FRAME_H__
#define __GTK_GRADIENT_H__
#define __GTK_GRID_H__
#define __GTK_H_INSIDE__
#define __GTK_H__
#define __GTK_ICON_FACTORY_H__
#define __GTK_ICON_THEME_H__
#define __GTK_ICON_VIEW_H__
#define __GTK_IMAGE_H__
#define __GTK_IMAGE_MENU_ITEM_H__
#define __GTK_IM_CONTEXT_H__
#define __GTK_IM_CONTEXT_INFO_H__
#define __GTK_IM_CONTEXT_SIMPLE_H__
#define __GTK_IM_MODULE_H__
#define __GTK_IM_MULTICONTEXT_H__
#define __GTK_INFO_BAR_H__
#define __GTK_INVISIBLE_H__
#define __GTK_LABEL_H__
#define __GTK_LAYOUT_H__
#define __GTK_LEVEL_BAR_H__
#define __GTK_LIST_STORE_H__
#define __GTK_LOCK_BUTTON_H__
#define __GTK_MAIN_H__
#define __GTK_MENU_BAR_H__
#define __GTK_MENU_BUTTON_H__
#define __GTK_MENU_H__
#define __GTK_MENU_ITEM_H__
#define __GTK_MENU_SHELL_H__
#define __GTK_MENU_TOOL_BUTTON_H__
#define __GTK_MESSAGE_DIALOG_H__
#define __GTK_MISC_H__
#define __GTK_MODULES_H__
#define __GTK_MOUNT_OPERATION_H__
#define __GTK_NOTEBOOK_H__
#define __GTK_NUMERABLE_ICON_H__
#define __GTK_OFFSCREEN_WINDOW_H__
#define __GTK_ORIENTABLE_H__
#define __GTK_OVERLAY_H__
#define __GTK_PANED_H__
#define __GTK_PLUG_H__
#define __GTK_PROGRESS_BAR_H__
#define __GTK_RADIO_ACTION_H__
#define __GTK_RADIO_BUTTON_H__
#define __GTK_RADIO_MENU_ITEM_H__
#define __GTK_RADIO_TOOL_BUTTON_H__
#define __GTK_RANGE_H__
#define __GTK_RECENT_ACTION_H__
#define __GTK_SCALE_BUTTON_H__
#define __GTK_SCALE_H__
#define __GTK_SCROLLABLE_H__
#define __GTK_SCROLLBAR_H__
#define __GTK_SCROLLED_WINDOW_H__
#define __GTK_SEARCH_ENTRY_H__
#define __GTK_SELECTION_H__
#define __GTK_SEPARATOR_H__

```

```

#define __GTK_SEPARATOR_MENU_ITEM_H__
#define __GTK_SEPARATOR_TOOL_ITEM_H__
#define __GTK_SETTINGS_H__
#define __GTK_SHOW_H__
#define __GTK_SIZE_GROUP_H__
#define __GTK_SIZE_REQUEST_H__
#define __GTK_SOCKET_H__
#define __GTK_SPINNER_H__
#define __GTK_SPIN_BUTTON_H__
#define __GTK_STATUSBAR_H__
#define __GTK_STOCK_H__
#define __GTK_STYLE_CONTEXT_H__
#define __GTK_STYLE_PROPERTIES_H__
#define __GTK_STYLE_PROVIDER_H__
#define __GTK_SWITCH_H__
#define __GTK_SYMBOLIC_COLOR_H__
#define __GTK_TEST_UTILS_H__
#define __GTK_TEXT_ATTRIBUTES_H__
#define __GTK_TEXT_BUFFER_H__
#define __GTK_TEXT_CHILD_H__
#define __GTK_TEXT_ITER_H__
#define __GTK_TEXT_MARK_H__
#define __GTK_TEXT_TAG_H__
#define __GTK_TEXT_TAG_TABLE_H__
#define __GTK_TEXT_VIEW_H__
#define __GTK_THEMING_ENGINE_H__
#define __GTK_TOGGLE_ACTION_H__
#define __GTK_TOGGLE_BUTTON_H__
#define __GTK_TOGGLE_TOOL_BUTTON_H__
#define __GTK_TOOLBAR_H__
#define __GTK_TOOLTIP_H__
#define __GTK_TOOL_BUTTON_H__
#define __GTK_TOOL_ITEM_GROUP_H__
#define __GTK_TOOL_ITEM_H__
#define __GTK_TOOL_PALETTE_H__
#define __GTK_TOOL_SHELL_H__
#define __GTK_TREE_DND_H__
#define __GTK_TREE_MODEL_FILTER_H__
#define __GTK_TREE_MODEL_H__
#define __GTK_TREE_MODEL_SORT_H__
#define __GTK_TREE_SELECTION_H__
#define __GTK_TREE_SORTABLE_H__
#define __GTK_TREE_STORE_H__
#define __GTK_TREE_VIEW_COLUMN_H__
#define __GTK_TREE_VIEW_H__
#define __GTK_TYPES_H__
#define __GTK_TYPE_BUILTINS_H__
#define __GTK_UI_MANAGER_H__
#define __GTK_VERSION_H__
#define __GTK_VIEWPORT_H__
#define __GTK_VOLUME_BUTTON_H__
#define __GTK_WIDGET_H__
#define __GTK_WIDGET_PATH_H__
#define __GTK_WINDOW_H__
#define __GTK_X_H__
#define GTK_TYPE_ACTIONABLE (gtk_actionable_get_type ())
#define GTK_TYPE_ACTIVATABLE (gtk_activatable_get_type ())
#define GTK_TYPE_ALIGN (gtk_align_get_type ())
#define GTK_TYPE_APPLICATION (gtk_application_get_type ())
#define GTK_TYPE_APPLICATION_INHIBIT_FLAGS
(gtk_application_inhibit_flags_get_type ())
#define GTK_TYPE_APPLICATION_WINDOW
(gtk_application_window_get_type ())
#define GTK_TYPE_APP_CHOOSER_BUTTON
(gtk_app_chooser_button_get_type ())

```

```

#define GTK_TYPE_APP_CHOOSER_DIALOG
(gtk_app_chooser_dialog_get_type ())
#define GTK_TYPE_APP_CHOOSER (gtk_app_chooser_get_type ())
#define GTK_TYPE_APP_CHOOSER_WIDGET
(gtk_app_chooser_widget_get_type ())
#define GTK_TYPE_ARROW_PLACEMENT
(gtk_arrow_placement_get_type ())
#define GTK_TYPE_ASSISTANT_PAGE_TYPE
(gtk_assistant_page_type_get_type ())
#define GTK_TYPE_BORDER_STYLE (gtk_border_style_get_type ())
#define GTK_TYPE_BUILDBABLE (gtk_buildable_get_type ())
#define GTK_TYPE_BUILDER_ERROR (gtk_builder_error_get_type ())
#define GTK_BUILDER_ERROR (gtk_builder_error_quark ())
#define GTK_TYPE_BUILDER (gtk_builder_get_type ())
#define GTK_TYPE_CELL_AREA_BOX (gtk_cell_area_box_get_type ())
#define GTK_TYPE_CELL_AREA_CONTEXT
(gtk_cell_area_context_get_type ())
#define GTK_TYPE_CELL_AREA (gtk_cell_area_get_type ())
#define GTK_TYPE_CELL_RENDERER_SPINNER
(gtk_cell_renderer_spinner_get_type ())
#define GTK_TYPE_COLOR_CHOOSER_DIALOG
(gtk_color_chooser_dialog_get_type ())
#define GTK_TYPE_COLOR_CHOOSER (gtk_color_chooser_get_type ())
#define GTK_TYPE_COLOR_CHOOSER_WIDGET
(gtk_color_chooser_widget_get_type ())
#define GTK_TYPE_COMBO_BOX_TEXT (gtk_combo_box_text_get_type ())
#define GTK_TYPE_CSS_PROVIDER_ERROR
(gtk_css_provider_error_get_type ())
#define GTK_CSS_PROVIDER_ERROR (gtk_css_provider_error_quark ())
#define GTK_TYPE_CSS_PROVIDER (gtk_css_provider_get_type ())
#define GTK_TYPE_CSS_SECTION (gtk_css_section_get_type ())
#define GTK_TYPE_CSS_SECTION_TYPE
(gtk_css_section_type_get_type ())
#define GTK_TYPE_DRAG_RESULT (gtk_drag_result_get_type ())
#define GTK_TYPE_ENTRY_BUFFER (gtk_entry_buffer_get_type ())
#define GTK_TYPE_ENTRY_ICON_POSITION
(gtk_entry_icon_position_get_type ())
#define GTK_TYPE_FILE_CHOOSER_CONFIRMATION
(gtk_file_chooser_confirmation_get_type ())
#define GTK_TYPE_FONT_CHOOSER_DIALOG
(gtk_font_chooser_dialog_get_type ())
#define GTK_TYPE_FONT_CHOOSER (gtk_font_chooser_get_type ())
#define GTK_TYPE_FONT_CHOOSER_WIDGET
(gtk_font_chooser_widget_get_type ())
#define GTK_TYPE_GRADIENT (gtk_gradient_get_type ())
#define GTK_TYPE_GRID (gtk_grid_get_type ())
#define GTK_TYPE_ICON_VIEW_DROP_POSITION
(gtk_icon_view_drop_position_get_type ())
#define GTK_TYPE_INFO_BAR (gtk_info_bar_get_type ())
#define GTK_TYPE_INPUT_HINTS (gtk_input_hints_get_type ())
#define GTK_TYPE_INPUT_PURPOSE (gtk_input_purpose_get_type ())
#define GTK_TYPE_JUNCTION_SIDES (gtk_junction_sides_get_type ())
#define GTK_TYPE_LEVEL_BAR (gtk_level_bar_get_type ())
#define GTK_TYPE_LEVEL_BAR_MODE (gtk_level_bar_mode_get_type ())
#define GTK_TYPE_LICENSE (gtk_license_get_type ())
#define GTK_TYPE_LOCK_BUTTON (gtk_lock_button_get_type ())
#define GTK_TYPE_MENU_BUTTON (gtk_menu_button_get_type ())
#define GTK_TYPE_MOUNT_OPERATION
(gtk_mount_operation_get_type ())
#define GTK_TYPE_NUMBER_UP_LAYOUT
(gtk_number_up_layout_get_type ())
#define GTK_TYPE_NUMERABLE_ICON (gtk_numerable_icon_get_type ())
#define GTK_TYPE_OFFSCREEN_WINDOW
(gtk_offscreen_window_get_type ())
#define GTK_TYPE_ORIENTABLE (gtk_orientable_get_type ())
#define GTK_TYPE_OVERLAY (gtk_overlay_get_type ())

```



```

#define GTK_TYPE_PACK_DIRECTION (gtk_pack_direction_get_type ())
#define GTK_TYPE_RECENT_ACTION (gtk_recent_action_get_type ())
#define GTK_TYPE_REGION_FLAGS (gtk_region_flags_get_type ())
#define GTK_TYPE_SCALE_BUTTON (gtk_scale_button_get_type ())
#define GTK_TYPE_SCROLLABLE (gtk_scrollable_get_type ())
#define GTK_TYPE_SCROLLABLE_POLICY (gtk_scrollable_policy_get_type ())
#define GTK_TYPE_SEARCH_ENTRY (gtk_search_entry_get_type ())
#define GTK_TYPE_SIZE_REQUEST_MODE (gtk_size_request_mode_get_type ())
#define GTK_TYPE_SPINNER (gtk_spinner_get_type ())
#define GTK_TYPE_STATE_FLAGS (gtk_state_flags_get_type ())
#define GTK_TYPE_STYLE_CONTEXT (gtk_style_context_get_type ())
#define GTK_TYPE_STYLE_PROPERTIES (gtk_style_properties_get_type ())
#define GTK_TYPE_STYLE_PROVIDER (gtk_style_provider_get_type ())
#define GTK_TYPE_SWITCH (gtk_switch_get_type ())
#define GTK_TYPE_SYMBOLIC_COLOR (gtk_symbolic_color_get_type ())
#define GTK_TYPE_THEMING_ENGINE (gtk_theming_engine_get_type ())
#define GTK_TYPE_TOOLTIP (gtk_tooltip_get_type ())
#define GTK_TYPE_TOOL_ITEM_GROUP (gtk_tool_item_group_get_type ())
#define GTK_TYPE_TOOL_PALETTE_DRAG_TARGETS (gtk_tool_palette_drag_targets_get_type ())
#define GTK_TYPE_TOOL_PALETTE (gtk_tool_palette_get_type ())
#define GTK_TYPE_TOOL_SHELL (gtk_tool_shell_get_type ())
#define GTK_TYPE_VOLUME_BUTTON (gtk_volume_button_get_type ())
#define GTK_TYPE_WIDGET_PATH (gtk_widget_path_get_type ())
#define GTK_CSS_PROVIDER_CLASS(c) (G_TYPE_CHECK_CLASS_CAST((c), GTK_TYPE_CSS_PROVIDER, GtkCssProviderClass))
#define GTK_STYLE_CONTEXT_CLASS(c) (G_TYPE_CHECK_CLASS_CAST((c), GTK_TYPE_STYLE_CONTEXT, GtkStyleContextClass))
#define GTK_STYLE_PROPERTIES_CLASS(c) (G_TYPE_CHECK_CLASS_CAST((c), GTK_TYPE_STYLE_PROPERTIES, GtkStylePropertiesClass))
#define GTK_THEMING_ENGINE_CLASS(c) (G_TYPE_CHECK_CLASS_CAST((c), GTK_TYPE_THEMING_ENGINE, GtkThemingEngineClass))
#define GTK_APPLICATION_WINDOW_CLASS(class) (G_TYPE_CHECK_CLASS_CAST((class), GTK_TYPE_APPLICATION_WINDOW, GtkApplicationWindowClass))
#define GTK_LOCK_BUTTON_CLASS(k) (G_TYPE_CHECK_CLASS_CAST((k), GTK_LOCK_BUTTON, GtkLockButtonClass))
#define GTK_OFFSCREEN_WINDOW_CLASS(k) (G_TYPE_CHECK_CLASS_CAST((k), GTK_TYPE_OFFSCREEN_WINDOW, GtkOffscreenWindowClass))
#define GTK_APPLICATION_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_APPLICATION, GtkApplicationClass))
#define GTK_APP_CHOOSER_BUTTON_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_APP_CHOOSER_BUTTON, GtkAppChooserButtonClass))
#define GTK_APP_CHOOSER_DIALOG_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_APP_CHOOSER_DIALOG, GtkAppChooserDialogClass))
#define GTK_APP_CHOOSER_WIDGET_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_APP_CHOOSER_WIDGET, GtkAppChooserWidgetClass))
#define GTK_BUILDER_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_BUILDER, GtkBuilderClass))
#define GTK_CELL_AREA_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_CELL_AREA, GtkCellAreaClass))
#define GTK_CELL_AREA_BOX_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_CELL_AREA_BOX, GtkCellAreaBoxClass))
#define GTK_CELL_AREA_CONTEXT_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_CELL_AREA_CONTEXT, GtkCellAreaContextClass))
#define GTK_CELL_RENDERER_SPINNER_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST((klass), GTK_TYPE_CELL_RENDERER_SPINNER, GtkCellRendererSpinnerClass))

```



```

#define GTK_COLOR_CHOOSER_DIALOG_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GTK_TYPE_COLOR_CHOOSER_DIALOG,
GtkColorChooserDialogClass))
#define GTK_COLOR_CHOOSER_WIDGET_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GTK_TYPE_COLOR_CHOOSER_WIDGET,
GtkColorChooserWidgetClass))
#define GTK_COMBO_BOX_TEXT_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_COMBO_BOX_TEXT, GtkComboBoxTextClass))
#define GTK_ENTRY_BUFFER_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_ENTRY_BUFFER, GtkEntryBufferClass))
#define GTK_FONT_CHOOSER_DIALOG_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GTK_TYPE_FONT_CHOOSER_DIALOG,
GtkFontChooserDialogClass))
#define GTK_FONT_CHOOSER_WIDGET_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass), GTK_TYPE_FONT_CHOOSER_WIDGET,
GtkFontChooserWidgetClass))
#define GTK_GRID_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST ((klass),
GTK_TYPE_GRID, GtkGridClass))
#define GTK_LEVEL_BAR_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_LEVEL_BAR, GtkLevelBarClass))
#define GTK_MENU_BUTTON_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_MENU_BUTTON, GtkMenuButtonClass))
#define GTK_NUMERABLE_ICON_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_NUMERABLE_ICON, GtkNumerableIconClass))
#define GTK_OVERLAY_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_OVERLAY, GtkOverlayClass))
#define GTK_RECENT_ACTION_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_RECENT_ACTION, GtkRecentActionClass))
#define GTK_SCALE_BUTTON_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_SCALE_BUTTON, GtkScaleButtonClass))
#define GTK_SEARCH_ENTRY_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_SEARCH_ENTRY, GtkSearchEntryClass))
#define GTK_SWITCH_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST ((klass),
GTK_TYPE_SWITCH, GtkSwitchClass))
#define GTK_VOLUME_BUTTON_CLASS(klass) (G_TYPE_CHECK_CLASS_CAST
((klass), GTK_TYPE_VOLUME_BUTTON, GtkVolumeButtonClass))
#define GTK_ACTIVATABLE_CLASS(obj) (G_TYPE_CHECK_CLASS_CAST
((obj), GTK_TYPE_ACTIVATABLE, GtkActivatableIface))
#define GTK_BUILDABLE_CLASS(obj) (G_TYPE_CHECK_CLASS_CAST
((obj), GTK_TYPE_BUILDABLE, GtkBuildableIface))
#define GTK_SPINNER_CLASS(obj) (G_TYPE_CHECK_CLASS_CAST ((obj),
GTK_TYPE_SPINNER, GtkSpinnerClass))
#define GTK_ORIENTABLE_CLASS(vtable) (G_TYPE_CHECK_CLASS_CAST
((vtable), GTK_TYPE_ORIENTABLE, GtkOrientableIface))
#define GTK_TOOL_ITEM_GROUP_CLASS(cls) (G_TYPE_CHECK_CLASS_CAST
(cls, GTK_TYPE_TOOL_ITEM_GROUP, GtkToolItemGroupClass))
#define GTK_TOOL_PALETTE_CLASS(cls) (G_TYPE_CHECK_CLASS_CAST
(cls, GTK_TYPE_TOOL_PALETTE, GtkToolPaletteClass))
#define GTK_MOUNT_OPERATION_CLASS(k)
(G_TYPE_CHECK_CLASS_CAST ((k),
GTK_TYPE_MOUNT_OPERATION,
GtkMountOperationClass))
#define GTK_INFO_BAR_CLASS(klass)
(G_TYPE_CHECK_CLASS_CAST ((klass),
GTK_TYPE_INFO_BAR,
GtkInfoBarClass))
#define GTK_IS_CSS_PROVIDER_CLASS(c) (G_TYPE_CHECK_CLASS_TYPE
((c), GTK_TYPE_CSS_PROVIDER))
#define GTK_IS_STYLE_CONTEXT_CLASS(c) (G_TYPE_CHECK_CLASS_TYPE
((c), GTK_TYPE_STYLE_CONTEXT))
#define GTK_IS_STYLE_PROPERTIES_CLASS(c)
(G_TYPE_CHECK_CLASS_TYPE ((c), GTK_TYPE_STYLE_PROPERTIES))
#define GTK_IS_THEMING_ENGINE_CLASS(c) (G_TYPE_CHECK_CLASS_TYPE
((c), GTK_TYPE_THEMING_ENGINE))
#define GTK_IS_APPLICATION_WINDOW_CLASS(class)
(G_TYPE_CHECK_CLASS_TYPE ((class), GTK_TYPE_APPLICATION_WINDOW))
#define GTK_IS_LOCK_BUTTON_CLASS(k) (G_TYPE_CHECK_CLASS_TYPE
((k), GTK_TYPE_LOCK_BUTTON))

```

```

#define GTK_IS_MOUNT_OPERATION_CLASS(k) (G_TYPE_CHECK_CLASS_TYPE
((k), GTK_TYPE_MOUNT_OPERATION))
#define GTK_IS_OFFSCREEN_WINDOW_CLASS(k)
(G_TYPE_CHECK_CLASS_TYPE ((k), GTK_TYPE_OFFSCREEN_WINDOW))
#define GTK_IS_APPLICATION_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_APPLICATION))
#define GTK_IS_APP_CHOOSER_BUTTON_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_APP_CHOOSER_BUTTON))
#define GTK_IS_APP_CHOOSER_DIALOG_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_APP_CHOOSER_DIALOG))
#define GTK_IS_APP_CHOOSER_WIDGET_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_APP_CHOOSER_WIDGET))
#define GTK_IS_BUILDER_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_BUILDER))
#define GTK_IS_CELL_AREA_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_CELL_AREA))
#define GTK_IS_CELL_AREA_BOX_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_CELL_AREA_BOX))
#define GTK_IS_CELL_AREA_CONTEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_CELL_AREA_CONTEXT))
#define GTK_IS_CELL_RENDERER_SPINNER_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_CELL_RENDERER_SPINNER))
#define GTK_IS_COLOR_CHOOSER_DIALOG_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_COLOR_CHOOSER_DIALOG))
#define GTK_IS_COLOR_CHOOSER_WIDGET_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_COLOR_CHOOSER_WIDGET))
#define GTK_IS_COMBO_BOX_TEXT_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_COMBO_BOX_TEXT))
#define GTK_IS_ENTRY_BUFFER_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_ENTRY_BUFFER))
#define GTK_IS_FONT_CHOOSER_DIALOG_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_FONT_CHOOSER_DIALOG))
#define GTK_IS_FONT_CHOOSER_WIDGET_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_FONT_CHOOSER_WIDGET))
#define GTK_IS_GRID_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_GRID))
#define GTK_IS_INFO_BAR_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_INFO_BAR))
#define GTK_IS_LEVEL_BAR_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_LEVEL_BAR))
#define GTK_IS_MENU_BUTTON_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_MENU_BUTTON))
#define GTK_IS_NUMERABLE_ICON_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_NUMERABLE_ICON))
#define GTK_IS_OVERLAY_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_OVERLAY))
#define GTK_IS_RECENT_ACTION_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_RECENT_ACTION))
#define GTK_IS_SCALE_BUTTON_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_SCALE_BUTTON))
#define GTK_IS_SEARCH_ENTRY_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_SEARCH_ENTRY))
#define GTK_IS_SWITCH_CLASS(klass) (G_TYPE_CHECK_CLASS_TYPE
((klass), GTK_TYPE_SWITCH))
#define GTK_IS_VOLUME_BUTTON_CLASS(klass)
(G_TYPE_CHECK_CLASS_TYPE ((klass), GTK_TYPE_VOLUME_BUTTON))
#define GTK_IS_SPINNER_CLASS(obj) (G_TYPE_CHECK_CLASS_TYPE
((obj), GTK_TYPE_SPINNER))
#define GTK_IS_ORIENTABLE_CLASS(vtable) (G_TYPE_CHECK_CLASS_TYPE
((vtable), GTK_TYPE_ORIENTABLE))
#define GTK_IS_TOOL_ITEM_GROUP_CLASS(obj)
(G_TYPE_CHECK_CLASS_TYPE (obj, GTK_TYPE_TOOL_ITEM_GROUP))
#define GTK_IS_TOOL_PALETTE_CLASS(obj) (G_TYPE_CHECK_CLASS_TYPE
(obj, GTK_TYPE_TOOL_PALETTE))
#define GTK_ACTIONABLE(inst) (G_TYPE_CHECK_INSTANCE_CAST
((inst), GTK_TYPE_ACTIONABLE, GtkActionable))

```

```

#define GTK_APPLICATION_WINDOW(inst)
(G_TYPE_CHECK_INSTANCE_CAST ((inst), GTK_TYPE_APPLICATION_WINDOW,
GtkApplicationWindow))
#define GTK_CSS_PROVIDER(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_CSS_PROVIDER, GtkCssProvider))
#define GTK_LOCK_BUTTON(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_LOCK_BUTTON, GtkLockButton))
#define GTK_MOUNT_OPERATION(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_MOUNT_OPERATION, GtkMountOperation))
#define GTK_OFFSCREEN_WINDOW(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_OFFSCREEN_WINDOW, GtkOffscreenWindow))
#define GTK_STYLE_CONTEXT(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_STYLE_CONTEXT, GtkStyleContext))
#define GTK_STYLE_PROPERTIES(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_STYLE_PROPERTIES, GtkStyleProperties))
#define GTK_STYLE_PROVIDER(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_STYLE_PROVIDER, GtkStyleProvider))
#define GTK_THEMING_ENGINE(o) (G_TYPE_CHECK_INSTANCE_CAST ((o),
GTK_TYPE_THEMING_ENGINE, GtkThemingEngine))
#define GTK_ACTIVATABLE(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_ACTIVATABLE, GtkActivatable))
#define GTK_APPLICATION(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_APPLICATION, GtkApplication))
#define GTK_APP_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_APP_CHOOSER, GtkAppChooser))
#define GTK_APP_CHOOSER_BUTTON(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_APP_CHOOSER_BUTTON,
GtkAppChooserButton))
#define GTK_APP_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_APP_CHOOSER_DIALOG,
GtkAppChooserDialog))
#define GTK_APP_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_APP_CHOOSER_WIDGET,
GtkAppChooserWidget))
#define GTK_BUILDBABLE(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_BUILDBABLE, GtkBuildable))
#define GTK_BUILDER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_BUILDER, GtkBuilder))
#define GTK_CELL_AREA(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_CELL_AREA, GtkCellArea))
#define GTK_CELL_AREA_BOX(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_CELL_AREA_BOX, GtkCellAreaBox))
#define GTK_CELL_AREA_CONTEXT(obj) (G_TYPE_CHECK_INSTANCE_CAST
((obj), GTK_TYPE_CELL_AREA_CONTEXT, GtkCellAreaContext))
#define GTK_CELL_RENDERER_SPINNER(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_CELL_RENDERER_SPINNER,
GtkCellRendererSpinner))
#define GTK_COLOR_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_COLOR_CHOOSER, GtkColorChooser))
#define GTK_COLOR_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_COLOR_CHOOSER_DIALOG,
GtkColorChooserDialog))
#define GTK_COLOR_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_COLOR_CHOOSER_WIDGET,
GtkColorChooserWidget))
#define GTK_COMBO_BOX_TEXT(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_COMBO_BOX_TEXT, GtkComboBoxText))
#define GTK_ENTRY_BUFFER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_ENTRY_BUFFER, GtkEntryBuffer))
#define GTK_FILE_CHOOSER_BUTTON(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_FILE_CHOOSER_BUTTON,
GtkFileChooserButton))
#define GTK_FONT_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_FONT_CHOOSER, GtkFontChooser))

```

```

#define GTK_FONT_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_FONT_CHOOSER_DIALOG,
GtkFontChooserDialog))
#define GTK_FONT_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_CAST ((obj), GTK_TYPE_FONT_CHOOSER_WIDGET,
GtkFontChooserWidget))
#define GTK_GRID(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_GRID, GtkGrid))
#define GTK_LEVEL_BAR(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_LEVEL_BAR, GtkLevelBar))
#define GTK_MENU_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_MENU_BUTTON, GtkMenuButton))
#define GTK_NUMERABLE_ICON(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_NUMERABLE_ICON, GtkNumerableIcon))
#define GTK_ORIENTABLE(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_ORIENTABLE, GtkOrientable))
#define GTK_OVERLAY(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_OVERLAY, GtkOverlay))
#define GTK_RECENT_ACTION(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_RECENT_ACTION, GtkRecentAction))
#define GTK_SCALE_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_SCALE_BUTTON, GtkScaleButton))
#define GTK_SCROLLABLE(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_SCROLLABLE, GtkScrollable))
#define GTK_SEARCH_ENTRY(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_SEARCH_ENTRY, GtkSearchEntry))
#define GTK_SPINNER(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_SPINNER, GtkSpinner))
#define GTK_SWITCH(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_SWITCH, GtkSwitch))
#define GTK_TOOLTIP(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_TOOLTIP, GtkTooltip))
#define GTK_TOOL_SHELL(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_TOOL_SHELL, GtkToolShell))
#define GTK_VOLUME_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_CAST ((obj),
GTK_TYPE_VOLUME_BUTTON, GtkVolumeButton))
#define GTK_TOOL_ITEM_GROUP(obj) (G_TYPE_CHECK_INSTANCE_CAST
(obj, GTK_TYPE_TOOL_ITEM_GROUP, GtkToolItemGroup))
#define GTK_TOOL_PALETTE(obj) (G_TYPE_CHECK_INSTANCE_CAST (obj,
GTK_TYPE_TOOL_PALETTE, GtkToolPalette))
#define GTK_INFO_BAR(obj) (G_TYPE_CHECK_INSTANCE_CAST((obj),
GTK_TYPE_INFO_BAR, GtkInfoBar))
#define GTK_IS_ACTIONABLE(inst) (G_TYPE_CHECK_INSTANCE_TYPE
((inst), GTK_TYPE_ACTIONABLE))
#define GTK_IS_APPLICATION_WINDOW(inst)
(G_TYPE_CHECK_INSTANCE_TYPE ((inst), GTK_TYPE_APPLICATION_WINDOW))
#define GTK_IS_CSS_PROVIDER(o) (G_TYPE_CHECK_INSTANCE_TYPE ((o),
GTK_TYPE_CSS_PROVIDER))
#define GTK_IS_LOCK_BUTTON(o) (G_TYPE_CHECK_INSTANCE_TYPE ((o),
GTK_TYPE_LOCK_BUTTON))
#define GTK_IS_MOUNT_OPERATION(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GTK_TYPE_MOUNT_OPERATION))
#define GTK_IS_OFFSCREEN_WINDOW(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GTK_TYPE_OFFSCREEN_WINDOW))
#define GTK_IS_STYLE_CONTEXT(o) (G_TYPE_CHECK_INSTANCE_TYPE ((o),
GTK_TYPE_STYLE_CONTEXT))
#define GTK_IS_STYLE_PROPERTIES(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GTK_TYPE_STYLE_PROPERTIES))
#define GTK_IS_STYLE_PROVIDER(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GTK_TYPE_STYLE_PROVIDER))
#define GTK_IS_THEMING_ENGINE(o) (G_TYPE_CHECK_INSTANCE_TYPE
((o), GTK_TYPE_THEMING_ENGINE))
#define GTK_IS_ACTIVATABLE(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_ACTIVATABLE))
#define GTK_IS_APPLICATION(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_APPLICATION))

```

```

#define GTK_IS_APP_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_APP_CHOOSER))
#define
                                GTK_IS_APP_CHOOSER_BUTTON(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_APP_CHOOSER_BUTTON))
#define
                                GTK_IS_APP_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_APP_CHOOSER_DIALOG))
#define
                                GTK_IS_APP_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_APP_CHOOSER_WIDGET))
#define GTK_IS_BUILDABLE(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_BUILDABLE))
#define GTK_IS_BUILDER(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_BUILDER))
#define GTK_IS_CELL_AREA(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_CELL_AREA))
#define GTK_IS_CELL_AREA_BOX(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_CELL_AREA_BOX))
#define
                                GTK_IS_CELL_AREA_CONTEXT(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_CELL_AREA_CONTEXT))
#define
                                GTK_IS_CELL_RENDERER_SPINNER(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_CELL_RENDERER_SPINNER))
#define GTK_IS_COLOR_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_COLOR_CHOOSER))
#define
                                GTK_IS_COLOR_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_COLOR_CHOOSER_DIALOG))
#define
                                GTK_IS_COLOR_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_COLOR_CHOOSER_WIDGET))
#define GTK_IS_COMBO_BOX_TEXT(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_COMBO_BOX_TEXT))
#define GTK_IS_ENTRY_BUFFER(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_ENTRY_BUFFER))
#define
                                GTK_IS_FILE_CHOOSER_BUTTON(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_FILE_CHOOSER_BUTTON))
#define GTK_IS_FONT_CHOOSER(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_FONT_CHOOSER))
#define
                                GTK_IS_FONT_CHOOSER_DIALOG(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_FONT_CHOOSER_DIALOG))
#define
                                GTK_IS_FONT_CHOOSER_WIDGET(obj)
(G_TYPE_CHECK_INSTANCE_TYPE ((obj), GTK_TYPE_FONT_CHOOSER_WIDGET))
#define GTK_IS_GRID(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_GRID))
#define GTK_IS_LEVEL_BAR(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_LEVEL_BAR))
#define GTK_IS_MENU_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_MENU_BUTTON))
#define GTK_IS_NUMERABLE_ICON(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_NUMERABLE_ICON))
#define GTK_IS_ORIENTABLE(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_ORIENTABLE))
#define GTK_IS_OVERLAY(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_OVERLAY))
#define GTK_IS_RECENT_ACTION(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_RECENT_ACTION))
#define GTK_IS_SCALE_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_SCALE_BUTTON))
#define GTK_IS_SCROLLABLE(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_SCROLLABLE))
#define GTK_IS_SEARCH_ENTRY(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_SEARCH_ENTRY))
#define GTK_IS_SPINNER(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_SPINNER))
#define GTK_IS_SWITCH(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_SWITCH))
#define GTK_IS_TOOLTIP(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_TOOLTIP))

```



```

#define GTK_IS_TOOL_SHELL(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_TOOL_SHELL))
#define GTK_IS_VOLUME_BUTTON(obj) (G_TYPE_CHECK_INSTANCE_TYPE
((obj), GTK_TYPE_VOLUME_BUTTON))
#define GTK_IS_TOOL_ITEM_GROUP(obj)
(G_TYPE_CHECK_INSTANCE_TYPE (obj, GTK_TYPE_TOOL_ITEM_GROUP))
#define GTK_IS_TOOL_PALETTE(obj) (G_TYPE_CHECK_INSTANCE_TYPE
(obj, GTK_TYPE_TOOL_PALETTE))
#define GTK_IS_INFO_BAR(obj) (G_TYPE_CHECK_INSTANCE_TYPE ((obj),
GTK_TYPE_INFO_BAR))
#define GTK_APPLICATION_WINDOW_GET_CLASS(inst)
(G_TYPE_INSTANCE_GET_CLASS ((inst), GTK_TYPE_APPLICATION_WINDOW,
GtkApplicationWindowClass))
#define GTK_CSS_PROVIDER_GET_CLASS(o) (G_TYPE_INSTANCE_GET_CLASS
((o), GTK_TYPE_CSS_PROVIDER, GtkCssProviderClass))
#define GTK_LOCK_BUTTON_GET_CLASS(o) (G_TYPE_INSTANCE_GET_CLASS
((o), GTK_TYPE_LOCK_BUTTON, GtkLockButtonClass))
#define GTK_MOUNT_OPERATION_GET_CLASS(o)
(G_TYPE_INSTANCE_GET_CLASS ((o), GTK_TYPE_MOUNT_OPERATION,
GtkMountOperationClass))
#define GTK_OFFSCREEN_WINDOW_GET_CLASS(o)
(G_TYPE_INSTANCE_GET_CLASS ((o), GTK_TYPE_OFFSCREEN_WINDOW,
GtkOffscreenWindowClass))
#define GTK_STYLE_CONTEXT_GET_CLASS(o) (G_TYPE_INSTANCE_GET_CLASS
((o), GTK_TYPE_STYLE_CONTEXT, GtkStyleContextClass))
#define GTK_STYLE_PROPERTIES_GET_CLASS(o)
(G_TYPE_INSTANCE_GET_CLASS ((o), GTK_TYPE_STYLE_PROPERTIES,
GtkStylePropertiesClass))
#define GTK_THEMING_ENGINE_GET_CLASS(o) (G_TYPE_INSTANCE_GET_CLASS
((o), GTK_TYPE_THEMING_ENGINE, GtkThemingEngineClass))
#define GTK_APPLICATION_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_APPLICATION, GtkApplicationClass))
#define GTK_APP_CHOOSER_BUTTON_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_APP_CHOOSER_BUTTON,
GtkAppChooserButtonClass))
#define GTK_APP_CHOOSER_DIALOG_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_APP_CHOOSER_DIALOG,
GtkAppChooserDialogClass))
#define GTK_APP_CHOOSER_WIDGET_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_APP_CHOOSER_WIDGET,
GtkAppChooserWidgetClass))
#define GTK_BUILDER_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_BUILDER, GtkBuilderClass))
#define GTK_CELL_AREA_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_CELL_AREA, GtkCellAreaClass))
#define GTK_CELL_AREA_BOX_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_CELL_AREA_BOX,
GtkCellAreaBoxClass))
#define GTK_CELL_AREA_CONTEXT_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_CELL_AREA_CONTEXT,
GtkCellAreaContextClass))
#define GTK_CELL_RENDERER_SPINNER_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_CELL_RENDERER_SPINNER,
GtkCellRendererSpinnerClass))
#define GTK_COLOR_CHOOSER_DIALOG_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_COLOR_CHOOSER_DIALOG,
GtkColorChooserDialogClass))
#define GTK_COLOR_CHOOSER_WIDGET_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_COLOR_CHOOSER_WIDGET,
GtkColorChooserWidgetClass))
#define GTK_COMBO_BOX_TEXT_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_COMBO_BOX_TEXT,
GtkComboBoxTextClass))
#define GTK_ENTRY_BUFFER_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_ENTRY_BUFFER, GtkEntryBufferClass))

```

```

#define GTK_FILE_CHOOSER_BUTTON_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_FILE_CHOOSER_BUTTON,
GtkFileChooserButtonClass))
#define GTK_FONT_CHOOSER_DIALOG_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_FONT_CHOOSER_DIALOG,
GtkFontChooserDialogClass))
#define GTK_FONT_CHOOSER_WIDGET_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_FONT_CHOOSER_WIDGET,
GtkFontChooserWidgetClass))
#define GTK_GRID_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS ((obj),
GTK_TYPE_GRID, GtkGridClass))
#define GTK_LEVEL_BAR_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_LEVEL_BAR, GtkLevelBarClass))
#define GTK_MENU_BUTTON_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_MENU_BUTTON, GtkMenuButtonClass))
#define GTK_NUMERABLE_ICON_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_NUMERABLE_ICON,
GtkNumerableIconClass))
#define GTK_OVERLAY_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_OVERLAY, GtkOverlayClass))
#define GTK_RECENT_ACTION_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_RECENT_ACTION,
GtkRecentActionClass))
#define GTK_SCALE_BUTTON_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_SCALE_BUTTON, GtkScaleButtonClass))
#define GTK_SEARCH_ENTRY_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_SEARCH_ENTRY, GtkSearchEntryClass))
#define GTK_SPINNER_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_SPINNER, GtkSpinnerClass))
#define GTK_SWITCH_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_SWITCH, GtkSwitchClass))
#define GTK_TOOL_ITEM_GROUP_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_TOOL_ITEM_GROUP,
GtkToolItemGroupClass))
#define GTK_TOOL_PALETTE_GET_CLASS(obj) (G_TYPE_INSTANCE_GET_CLASS
((obj), GTK_TYPE_TOOL_PALETTE, GtkToolPaletteClass))
#define GTK_TREE_MODEL_FILTER_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_TREE_MODEL_FILTER,
GtkTreeModelFilterClass))
#define GTK_VOLUME_BUTTON_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_VOLUME_BUTTON,
GtkVolumeButtonClass))
#define GTK_INFO_BAR_GET_CLASS(obj)
(G_TYPE_INSTANCE_GET_CLASS ((obj), GTK_TYPE_INFO_BAR,
GtkInfoBarClass))
#define GTK_ACTIONABLE_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_ACTIONABLE,
GtkActionableInterface))
#define GTK_COLOR_CHOOSER_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_COLOR_CHOOSER,
GtkColorChooserInterface))
#define GTK_EDITABLE_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_EDITABLE,
GtkEditableInterface))
#define GTK_FONT_CHOOSER_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_FONT_CHOOSER,
GtkFontChooserIface))
#define GTK_ORIENTABLE_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_ORIENTABLE,
GtkOrientableIface))
#define GTK_SCROLLABLE_GET_IFACE(inst)
(G_TYPE_INSTANCE_GET_INTERFACE ((inst), GTK_TYPE_SCROLLABLE,
GtkScrollableInterface))
#define GTK_STYLE_PROVIDER_GET_IFACE(o)
(G_TYPE_INSTANCE_GET_INTERFACE ((o), GTK_TYPE_STYLE_PROVIDER,
GtkStyleProviderIface))

```

```

#define GTK_ACTIVATABLE_GET_IFACE(obj)
(G_TYPE_INSTANCE_GET_INTERFACE ((obj), GTK_TYPE_ACTIVATABLE,
GtkActivatableIface))
#define GTK_BUILDABLE_GET_IFACE(obj)
(G_TYPE_INSTANCE_GET_INTERFACE ((obj), GTK_TYPE_BUILDABLE,
GtkBuildableIface))
#define GTK_TOOL_SHELL_GET_IFACE(obj)
(G_TYPE_INSTANCE_GET_INTERFACE ((obj), GTK_TYPE_TOOL_SHELL,
GtkToolShellIface))
#define GTK_PATH_PRIO_MASK 0x0f
#define GTK_STYLE_PROVIDER_PRIORITY_FALLBACK 1
#define GTK_STYLE_PROVIDER_PRIORITY_THEME 200
#define GTK_STYLE_PROVIDER_PRIORITY_SETTINGS 400
#define GTK_STYLE_PROVIDER_PRIORITY_APPLICATION 600
#define GTK_STYLE_PROVIDER_PRIORITY_USER 800
#define GTK_STYLE_CLASS_ACCELERATOR "accelerator"
#define GTK_STYLE_CLASS_ARROW "arrow"
#define GTK_STYLE_CLASS_BACKGROUND "background"
#define GTK_STYLE_PROPERTY_BACKGROUND_COLOR "background-color"
#define GTK_STYLE_PROPERTY_BACKGROUND_IMAGE "background-image"
#define GTK_STYLE_PROPERTY_BORDER_COLOR "border-color"
#define GTK_STYLE_PROPERTY_BORDER_RADIUS "border-radius"
#define GTK_STYLE_PROPERTY_BORDER_STYLE "border-style"
#define GTK_STYLE_PROPERTY_BORDER_WIDTH "border-width"
#define GTK_STYLE_CLASS_BOTTOM "bottom"
#define GTK_STYLE_CLASS_BUTTON "button"
#define GTK_STYLE_CLASS_CALENDAR "calendar"
#define GTK_STYLE_CLASS_CELL "cell"
#define GTK_STYLE_CLASS_CHECK "check"
#define GTK_STYLE_PROPERTY_COLOR "color"
#define GTK_STYLE_REGION_COLUMN "column"
#define GTK_STYLE_REGION_COLUMN_HEADER "column-header"
#define GTK_STYLE_CLASS_COMBOBOX_ENTRY "combobox-entry"
#define GTK_STYLE_CLASS_CURSOR_HANDLE "cursor-handle"
#define GTK_STYLE_CLASS_DEFAULT "default"
#define GTK_STYLE_CLASS_DIM_LABEL "dim-label"
#define GTK_STYLE_CLASS_DND "dnd"
#define GTK_STYLE_CLASS_DOCK "dock"
#define GTK_STYLE_CLASS_ENTRY "entry"
#define GTK_STYLE_CLASS_ERROR "error"
#define GTK_STYLE_CLASS_EXPANDER "expander"
#define GTK_STYLE_PROPERTY_FONT "font"
#define GTK_STYLE_CLASS_FRAME "frame"
#define GTK_STYLE_CLASS_GRIP "grip"
#define GTK_STOCK_CAPS_LOCK_WARNING "gtk-caps-lock-warning"
#define GTK_STOCK_DISCARD "gtk-discard"
#define GTK_STOCK_ORIENTATION_LANDSCAPE "gtk-orientation-
landscape"
#define GTK_STOCK_ORIENTATION_PORTRAIT "gtk-orientation-portrait"
#define GTK_STOCK_ORIENTATION_REVERSE_LANDSCAPE "gtk-orientation-
reverse-landscape"
#define GTK_STOCK_ORIENTATION_REVERSE_PORTRAIT "gtk-orientation-
reverse-portrait"
#define GTK_STOCK_PAGE_SETUP "gtk-page-setup"
#define GTK_STOCK_PRINT_ERROR "gtk-print-error"
#define GTK_STOCK_PRINT_PAUSED "gtk-print-paused"
#define GTK_STOCK_PRINT_REPORT "gtk-print-report"
#define GTK_STOCK_PRINT_WARNING "gtk-print-warning"
#define gtk_binary_age gtk_get_binary_age ()
#define gtk_interface_age gtk_get_interface_age ()
#define gtk_major_version gtk_get_major_version ()
#define gtk_micro_version gtk_get_micro_version ()
#define gtk_minor_version gtk_get_minor_version ()
#define GTK_UNIT_PIXEL GTK_UNIT_NONE
#define GTK_ENTRY_BUFFER_MAX_SIZE G_MAXUSHORT

```



```

#define
GTK_CELL_AREA_WARN_INVALID_CELL_PROPERTY_ID(object,property_id,ps
pec)    G_OBJECT_WARN_INVALID_PSPEC ((object), "cell property id",
(property_id), (pspec))
#define          GTK_BUILDER_WARN_INVALID_CHILD_TYPE(object,type)
g_warning ("%s' is not a valid child type of '%s'", type,
g_type_name (G_OBJECT_TYPE (object)))
#define GTK_STYLE_CLASS_HEADER "header"
#define GTK_LEVEL_BAR_OFFSET_HIGH "high"
#define GTK_STYLE_CLASS_HIGHLIGHT "highlight"
#define GTK_STYLE_CLASS_HORIZONTAL "horizontal"
#define GTK_STYLE_CLASS_IMAGE "image"
#define GTK_STYLE_CLASS_INFO "info"
#define GTK_STYLE_CLASS_INLINE_TOOLBAR "inline-toolbar"
#define GTK_STYLE_CLASS_LEFT "left"
#define GTK_STYLE_CLASS_LEVEL_BAR "level-bar"
#define GTK_STYLE_CLASS_LINKED "linked"
#define GTK_LEVEL_BAR_OFFSET_LOW "low"
#define GTK_STYLE_PROPERTY_MARGIN "margin"
#define GTK_STYLE_CLASS_MARK "mark"
#define GTK_STYLE_CLASS_MENU "menu"
#define GTK_STYLE_CLASS_MENUBAR "menubar"
#define GTK_STYLE_CLASS_MENUITEM "menuitem"
#define GTK_STYLE_CLASS_NOTEBOOK "notebook"
#define GTK_STYLE_CLASS_OSD "osd"
#define GTK_PRINT_SETTINGS_OUTPUT_BASENAME "output-basename"
#define GTK_PRINT_SETTINGS_OUTPUT_DIR "output-dir"
#define GTK_STYLE_PROPERTY_PADDING "padding"
#define GTK_STYLE_CLASS_PANE_SEPARATOR "pane-separator"
#define GTK_STYLE_CLASS_PRIMARY_TOOLBAR "primary-toolbar"
#define GTK_STYLE_CLASS_PROGRESSBAR "progressbar"
#define GTK_STYLE_CLASS_PULSE "pulse"
#define GTK_STYLE_CLASS_QUESTION "question"
#define GTK_STYLE_CLASS_RADIO "radio"
#define GTK_STYLE_CLASS_RAISED "raised"
#define GTK_STYLE_CLASS_RIGHT "right"
#define GTK_STYLE_REGION_ROW "row"
#define GTK_STYLE_CLASS_RUBBERBAND "rubberband"
#define GTK_STYLE_CLASS_SCALE "scale"
#define GTK_STYLE_CLASS_SCALE_HAS_MARKS_ABOVE "scale-has-marks-
above"
#define GTK_STYLE_CLASS_SCALE_HAS_MARKS_BELOW "scale-has-marks-
below"
#define GTK_STYLE_CLASS_SCROLLBAR "scrollbar"
#define GTK_STYLE_CLASS_SCROLLBARS_JUNCTION "scrollbars-
junction"
#define GTK_STYLE_CLASS_SEPARATOR "separator"
#define GTK_STYLE_CLASS_SIDEBAR "sidebar"
#define GTK_STYLE_CLASS_SLIDER "slider"
#define GTK_STYLE_CLASS_SPINBUTTON "spinbutton"
#define GTK_STYLE_CLASS_SPINNER "spinner"
#define GTK_STYLE_REGION_TAB "tab"
#define GTK_STYLE_CLASS_TOOLBAR "toolbar"
#define GTK_STYLE_CLASS_TOOLTIP "tooltip"
#define GTK_STYLE_CLASS_TOP "top"
#define GTK_STYLE_CLASS_TROUGH "trough"
#define GTK_STYLE_CLASS_VERTICAL "vertical"
#define GTK_STYLE_CLASS_VIEW "view"
#define GTK_STYLE_CLASS_WARNING "warning"

typedef struct _GtkSettingsPrivate GtkSettingsPrivate;
typedef struct _GtkSettingsClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);

```

```

    void (*_gtk_reserved4) (void);
} GtkSettingsClass;
typedef struct _GtkSettingsValue {
    gchar *origin;
    GValue value;
} GtkSettingsValue;
typedef struct _GtkScaleButton {
    GtkButton parent;
    GtkScaleButtonPrivate *priv;
} GtkScaleButton;
typedef struct _GtkScaleButtonClass {
    GtkButtonClass parent_class;
    void (*value_changed) (GtkScaleButton * button, gdouble value);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkScaleButtonClass;
typedef struct _GtkScaleButtonPrivate GtkScaleButtonPrivate;
typedef struct _GtkComboBoxText {
    GtkComboBox parent_instance;
    GtkComboBoxTextPrivate *priv;
} GtkComboBoxText;
typedef struct _GtkComboBoxTextPrivate GtkComboBoxTextPrivate;
typedef struct _GtkComboBoxTextClass {
    GtkComboBoxClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkComboBoxTextClass;
typedef struct _GtkToolItemGroup {
    GtkContainer parent_instance;
    GtkToolItemGroupPrivate *priv;
} GtkToolItemGroup;
typedef struct _GtkToolItemGroupClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToolItemGroupClass;
typedef struct _GtkToolItemGroupPrivate GtkToolItemGroupPrivate;
typedef struct _GtkAdjustmentPrivate GtkAdjustmentPrivate;
typedef struct _GtkAdjustmentClass {
    GInitiallyUnownedClass parent_class;
    void (*changed) (GtkAdjustment * adjustment);
    void (*value_changed) (GtkAdjustment * adjustment);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAdjustmentClass;
typedef struct _GtkTextChildAnchor {
    GObject parent_instance;
    gpointer segment;
} GtkTextChildAnchor;
typedef struct _GtkTextChildAnchorClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTextChildAnchorClass;
typedef enum {
    GTK_TEXT_SEARCH_VISIBLE_ONLY,

```

```

    GTK_TEXT_SEARCH_TEXT_ONLY,
    GTK_TEXT_SEARCH_CASE_INSENSITIVE
} GtkTextSearchFlags;
typedef struct _GtkTextBuffer {
    GObject parent_instance;
    GtkTextBufferPrivate *priv;
} GtkTextBuffer;
typedef gboolean (*GtkTextCharPredicate) (void);
typedef struct _GtkSearchEntry {
    GtkEntry parent;
} GtkSearchEntry;
typedef struct _GtkSearchEntryClass {
    GtkEntryClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSearchEntryClass;
typedef struct _GtkPrintSettings GtkPrintSettings;
typedef void (*GtkPrintSettingsFunc) (const gchar * key,
                                       const gchar * value,
                                       gpointer user_data);

typedef struct _GtkPageRange {
    gint start;
    gint end;
} GtkPageRange;
typedef struct _GtkEventBox {
    GtkBin bin;
    GtkEventBoxPrivate *priv;
} GtkEventBox;
typedef struct _GtkEventBoxClass {
    GtkBinClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkEventBoxClass;
typedef struct _GtkEventBoxPrivate GtkEventBoxPrivate;
typedef struct _GtkActivatable GtkActivatable;
typedef struct _GtkActivatableIface {
    GTypeInterface g_iface;
    void (*update) (GtkActivatable * activatable, GtkAction * action,
                   const gchar * property_name);
    void (*sync_action_properties) (GtkActivatable * activatable,
                                    GtkAction * action);
} GtkActivatableIface;
typedef struct _GtkMisc {
    GtkWidget widget;
    GtkMiscPrivate *priv;
} GtkMisc;
typedef struct _GtkMiscPrivate GtkMiscPrivate;
typedef struct _GtkMiscClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMiscClass;
typedef struct _GtkColorChooserDialog {
    GtkDialog parent_instance;
    GtkColorChooserDialogPrivate *priv;
} GtkColorChooserDialog;
typedef struct _GtkColorChooserDialogPrivate
GtkColorChooserDialogPrivate;
typedef struct _GtkColorChooserDialogClass {
    GtkDialogClass parent_class;

```

```

        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkColorChooserDialogClass;
typedef struct _GtkBin {
    GtkContainer container;
    GtkBinPrivate *priv;
} GtkBin;
typedef struct _GtkBinPrivate GtkBinPrivate;
typedef struct _GtkBinClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkBinClass;
typedef struct _GtkPrintContext GtkPrintContext;
typedef enum {
    GTK_UPDATE_ALWAYS,
    GTK_UPDATE_IF_VALID
} GtkSpinButtonUpdatePolicy;
typedef enum {
    GTK_SPIN_STEP_FORWARD,
    GTK_SPIN_STEP_BACKWARD,
    GTK_SPIN_PAGE_FORWARD,
    GTK_SPIN_PAGE_BACKWARD,
    GTK_SPIN_HOME,
    GTK_SPIN_END,
    GTK_SPIN_USER_DEFINED
} GtkSpinType;
typedef struct _GtkSpinButton {
    GtkEntry entry;
    GtkSpinButtonPrivate *priv;
} GtkSpinButton;
typedef struct _GtkSpinButtonPrivate GtkSpinButtonPrivate;
typedef struct _GtkSpinButtonClass {
    GtkEntryClass parent_class;
    gint(*input) (GtkSpinButton * spin_button, gdouble * new_value);
    gint(*output) (GtkSpinButton * spin_button);
    void (*value_changed) (GtkSpinButton * spin_button);
    void (*change_value) (GtkSpinButton * spin_button,
        GtkScrollType scroll);
    void (*wrapped) (GtkSpinButton * spin_button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSpinButtonClass;
typedef struct _GtkToolButton {
    GtkToolItem parent;
    GtkToolButtonPrivate *priv;
} GtkToolButton;
typedef struct _GtkToolButtonClass {
    GtkToolItemClass parent_class;
    GType button_type;
    void (*clicked) (GtkToolButton * tool_item);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToolButtonClass;
typedef struct _GtkToolButtonPrivate GtkToolButtonPrivate;
typedef struct _GtkIMContext {
    GObject parent_instance;
} GtkIMContext;

```

```

typedef struct _GtkIMContextClass {
    GObjectClass parent_class;
    void (*preedit_start) (GtkIMContext * context);
    void (*preedit_end) (GtkIMContext * context);
    void (*preedit_changed) (GtkIMContext * context);
    void (*commit) (GtkIMContext * context, const gchar * str);
    gboolean(*retrieve_surrounding) (GtkIMContext * context);
    gboolean(*delete_surrounding) (GtkIMContext * context, gint
offset,
                                gint n_chars);
    void (*set_client_window) (GtkIMContext * context, GdkWindow *
window);
    void (*get_preedit_string) (GtkIMContext * context, gchar * *str,
                                PangoAttrList * *attrs, gint *
cursor_pos);
    gboolean(*filter_keypress) (GtkIMContext * context,
                                GdkEventKey * event);
    void (*focus_in) (GtkIMContext * context);
    void (*focus_out) (GtkIMContext * context);
    void (*reset) (GtkIMContext * context);
    void (*set_cursor_location) (GtkIMContext * context,
                                GdkRectangle * area);
    void (*set_use_preedit) (GtkIMContext * context, gboolean
use_preedit);
    void (*set_surrounding) (GtkIMContext * context, const gchar *
text,
                                gint len, gint cursor_index);
    gboolean(*get_surrounding) (GtkIMContext * context, gchar *
*text,
                                gint * cursor_index);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
} GtkIMContextClass;
typedef struct _GtkToolShell GtkToolShell;
typedef struct _GtkToolShellIface {
    GTypeInterface g_iface;
    GtkIconSize(*get_icon_size) (GtkToolShell * shell);
    GtkOrientation(*get_orientation) (GtkToolShell * shell);
    GtkToolbarStyle(*get_style) (GtkToolShell * shell);
    GtkReliefStyle(*get_relief_style) (GtkToolShell * shell);
    void (*rebuild_menu) (GtkToolShell * shell);
    GtkOrientation(*get_text_orientation) (GtkToolShell * shell);
    gfloat(*get_text_alignment) (GtkToolShell * shell);
    PangoEllipsizeMode(*get_ellipsize_mode) (GtkToolShell * shell);
    GtkSizeGroup *(*get_text_size_group) (GtkToolShell * shell);
} GtkToolShellIface;
typedef struct _GtkToggleButton {
    GtkButton button;
    GtkToggleButtonPrivate *priv;
} GtkToggleButton;
typedef struct _GtkToggleButtonPrivate GtkToggleButtonPrivate;
typedef struct _GtkToggleButtonClass {
    GtkButtonClass parent_class;
    void (*toggled) (GtkToggleButton * toggle_button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToggleButtonClass;
typedef struct _GtkMenuShell {
    GtkContainer container;
    GtkMenuShellPrivate *priv;
}

```

```

} GtkMenuShell;
typedef struct _GtkMenuShellClass {
    GtkContainerClass parent_class;
    guint submenu_placement:1;
    void (*deactivate) (GtkMenuShell * menu_shell);
    void (*selection_done) (GtkMenuShell * menu_shell);
    void (*move_current) (GtkMenuShell * menu_shell,
                          GtkMenuDirectionType direction);
    void (*activate_current) (GtkMenuShell * menu_shell,
                              gboolean force_hide);
    void (*cancel) (GtkMenuShell * menu_shell);
    void (*select_item) (GtkMenuShell * menu_shell, GtkWidget *
menu_item);
    void (*insert) (GtkMenuShell * menu_shell, GtkWidget * child,
                    gint position);
    gint(*get_popup_delay) (GtkMenuShell * menu_shell);
    gboolean(*move_selected) (GtkMenuShell * menu_shell, gint
distance);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMenuShellClass;
typedef struct _GtkMenuShellPrivate GtkMenuShellPrivate;
typedef struct _GtkFileChooserDialog {
    GtkDialog parent_instance;
    GtkFileChooserDialogPrivate *priv;
} GtkFileChooserDialog;
typedef struct _GtkFileChooserDialogPrivate
GtkFileChooserDialogPrivate;
typedef struct _GtkFileChooserDialogClass {
    GtkDialogClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFileChooserDialogClass;
typedef struct _GtkRecentInfo GtkRecentInfo;
typedef struct _GtkRecentData {
    gchar *display_name;
    gchar *description;
    gchar *mime_type;
    gchar *app_name;
    gchar *app_exec;
    gchar **groups;
    gboolean is_private;
} GtkRecentData;
typedef struct _GtkRecentManager {
    GObject parent;
    GtkRecentManagerPrivate *priv;
} GtkRecentManager;
typedef struct _GtkRecentManagerClass {
    GObjectClass parent_class;
    void (*changed) (GtkRecentManager * manager);
    void (*_gtk_recent1) (void);
    void (*_gtk_recent2) (void);
    void (*_gtk_recent3) (void);
    void (*_gtk_recent4) (void);
} GtkRecentManagerClass;
typedef struct _GtkRecentManagerPrivate GtkRecentManagerPrivate;
typedef struct _GtkFileChooserWidget {
    GtkBox parent_instance;
    GtkFileChooserWidgetPrivate *priv;
} GtkFileChooserWidget;
typedef struct _GtkFileChooserWidgetPrivate
GtkFileChooserWidgetPrivate;

```

```

typedef struct _GtkFileChooserWidgetClass {
    GtkBoxClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFileChooserWidgetClass;
typedef struct _GtkIMContextSimple {
    GtkIMContext object;
    GtkIMContextSimplePrivate *priv;
} GtkIMContextSimple;
typedef struct _GtkIMContextSimplePrivate
GtkIMContextSimplePrivate;
typedef struct _GtkIMContextSimpleClass {
    GtkIMContextClass parent_class;
} GtkIMContextSimpleClass;
typedef struct _GtkRadioButton {
    GtkCheckButton check_button;
    GtkRadioButtonPrivate *priv;
} GtkRadioButton;
typedef struct _GtkRadioButtonPrivate GtkRadioButtonPrivate;
typedef struct _GtkRadioButtonClass {
    GtkCheckButtonClass parent_class;
    void (*group_changed) (GtkRadioButton * radio_button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRadioButtonClass;
typedef struct _GtkSeparatorMenuItem {
    GtkMenuItem menu_item;
} GtkSeparatorMenuItem;
typedef struct _GtkSeparatorMenuItemClass {
    GtkMenuItemClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSeparatorMenuItemClass;
typedef struct _GtkPageSetup GtkPageSetup;
typedef gint(*GtkKeySnoopFunc) (GtkWidget * grab_widget,
                                GdkEventKey * event, gpointer func_data);
typedef struct _GtkLinkButton {
    GtkButton parent_instance;
    GtkLinkButtonPrivate *priv;
} GtkLinkButton;
typedef struct _GtkLinkButtonClass {
    GtkButtonClass parent_class;
    gboolean(*activate_link) (GtkLinkButton * button);
    void (*_gtk_padding1) (void);
    void (*_gtk_padding2) (void);
    void (*_gtk_padding3) (void);
    void (*_gtk_padding4) (void);
} GtkLinkButtonClass;
typedef struct _GtkLinkButtonPrivate GtkLinkButtonPrivate;
typedef struct _GtkAdjustment {
    GInitiallyUnowned parent_instance;
    GtkAdjustmentPrivate *priv;
} GtkAdjustment;
typedef struct _GtkClipboard GtkClipboard;
typedef struct _GtkIconSet GtkIconSet;
typedef struct _GtkIconSource GtkIconSource;
typedef struct _GtkRcStyle {
    GObject parent_instance;
    gchar *name;
    gchar *bg_pixmap_name;

```

```

    PangoFontDescription *font_desc;
    GtkRcFlags color_flags;
    GdkColor fg;
    GdkColor bg;
    GdkColor text;
    GdkColor base;
    gint xthickness;
    gint ythickness;
    GArray *rc_properties;
    GSList *rc_style_lists;
    GSList *icon_factories;
    quint engine_specified:1;
} GtkRcStyle;
typedef struct _GtkRequisition {
    gint width;
    gint height;
} GtkRequisition;
typedef struct _GtkSelectionData GtkSelectionData;
typedef struct _GtkSettings {
    GObject parent_instance;
    GtkSettingsPrivate *priv;
} GtkSettings;
typedef struct _GtkStyle {
    GObject parent_instance;
    GdkColor fg;
    GdkColor bg;
    GdkColor light;
    GdkColor dark;
    GdkColor mid;
    GdkColor text;
    GdkColor base;
    GdkColor text_aa;
    GdkColor black;
    GdkColor white;
    PangoFontDescription *font_desc;
    gint xthickness;
    gint ythickness;
    cairo_pattern_t *background;
    gint attach_count;
    GdkVisual *visual;
    PangoFontDescription *private_font_desc;
    GtkRcStyle *rc_style;
    GSList *styles;
    GArray *property_cache;
    GSList *icon_factories;
} GtkStyle;
typedef struct _GtkStyleContext {
    GObject parent_object;
    GtkStyleContextPrivate *priv;
} GtkStyleContext;
typedef struct _GtkTooltip GtkTooltip;
typedef struct _GtkWidget {
    GInitiallyUnowned parent_instance;
    GtkWidgetPrivate *priv;
} GtkWidget;
typedef struct _GtkWidgetPath GtkWidgetPath;
typedef struct _GtkWindow {
    GtkBin bin;
    GtkWindowPrivate *priv;
} GtkWindow;
typedef gboolean(*GtkRcPropertyParser) (const GParamSpec * pspec,
                                         const GString * rc_string,
                                         GValue * property_value);

typedef struct _GtkRange {
    GtkWidget widget;
    GtkRangePrivate *priv;
}

```



```

} GtkRange;
typedef struct _GtkRangePrivate GtkRangePrivate;
typedef struct _GtkRangeClass {
    GtkWidgetClass parent_class;
    gchar *slider_detail;
    gchar *stepper_detail;
    void (*value_changed) (GtkRange * range);
    void (*adjust_bounds) (GtkRange * range, gdouble new_value);
    void (*move_slider) (GtkRange * range, GtkScrollType scroll);
    void (*get_range_border) (GtkRange * range, GtkBorder * border);
    gboolean(*change_value) (GtkRange * range, GtkScrollType
scroll,
                                gdouble new_value);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRangeClass;
typedef struct _GtkTextMark {
    GObject parent_instance;
    gpointer segment;
} GtkTextMark;
typedef struct _GtkTextMarkClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTextMarkClass;
typedef struct _GtkTreeSelectionPrivate GtkTreeSelectionPrivate;
typedef gboolean(*GtkTreeSelectionFunc) (GtkTreeSelection *
selection,
                                GtkTreeModel * model,
                                GtkTreePath * path,
                                gboolean
path_currently_selected,
                                gpointer data);
typedef void (*GtkTreeSelectionForeachFunc) (GtkTreeModel * model,
                                GtkTreePath * path,
                                GtkTreeIter * iter,
                                gpointer data);

typedef struct _GtkSeparator {
    GtkWidget widget;
    GtkSeparatorPrivate *priv;
} GtkSeparator;
typedef struct _GtkSeparatorPrivate GtkSeparatorPrivate;
typedef struct _GtkSeparatorClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSeparatorClass;
typedef struct _GtkTreeViewColumn {
    GInitiallyUnowned parent_instance;
    GtkTreeViewColumnPrivate *priv;
} GtkTreeViewColumn;
typedef struct _GtkTreeViewColumnClass {
    GInitiallyUnownedClass parent_class;
    void (*clicked) (GtkTreeViewColumn * tree_column);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTreeViewColumnClass;
typedef struct _GtkTreeViewColumnPrivate GtkTreeViewColumnPrivate;

```

```

typedef enum {
    GTK_TREE_VIEW_COLUMN_GROW_ONLY,
    GTK_TREE_VIEW_COLUMN_AUTOSIZE,
    GTK_TREE_VIEW_COLUMN_FIXED
} GtkTreeViewColumnSizing;
typedef void (*GtkTreeCellDataFunc) (GtkTreeViewColumn *
tree_column,
                                GtkCellRenderer * cell,
                                GtkTreeModel * tree_model,
                                GtkTreeIter * iter, gpointer data);

typedef struct _GtkTreeModelSort {
    GObject parent;
    GtkTreeModelSortPrivate *priv;
} GtkTreeModelSort;
typedef struct _GtkTreeModelSortClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTreeModelSortClass;
typedef struct _GtkTreeModelSortPrivate GtkTreeModelSortPrivate;
typedef struct _GtkViewport {
    GtkBin bin;
    GtkViewportPrivate *priv;
} GtkViewport;
typedef struct _GtkViewportPrivate GtkViewportPrivate;
typedef struct _GtkViewportClass {
    GtkBinClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkViewportClass;
typedef struct _GtkStatusIcon {
    GtkDialog parent;
    GtkStatusIconPrivate *priv;
} GtkStatusIcon;
typedef struct _GtkStatusIconClass {
    GObjectClass parent_class;
    void (*activate) (GtkStatusIcon * status_icon);
    void (*popup_menu) (GtkStatusIcon * status_icon, guint button,
                        guint32 activate_time);
    gboolean(*size_changed) (GtkStatusIcon * status_icon, gint
size);
    gboolean(*button_press_event) (GtkStatusIcon * status_icon,
                                GdkEventKey * event);
    gboolean(*button_release_event) (GtkStatusIcon * status_icon,
                                GdkEventKey * event);
    gboolean(*scroll_event) (GtkStatusIcon * status_icon,
                            GdkEventScroll * event);
    gboolean(*query_tooltip) (GtkStatusIcon * status_icon, gint x,
gint y,
                                gboolean keyboard_mode,
                                GtkTooltip * tooltip);
    void *_gtk_reserved1;
    void *_gtk_reserved2;
    void *_gtk_reserved3;
    void *_gtk_reserved4;
} GtkStatusIconClass;
typedef struct _GtkStatusIconPrivate GtkStatusIconPrivate;
typedef struct _GtkBox {
    GtkContainer container;
    GtkBoxPrivate *priv;
} GtkBox;
typedef struct _GtkBoxPrivate GtkBoxPrivate;

```

```

typedef struct _GtkBoxClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkBoxClass;
typedef struct _GtkCalendar {
    GtkWidget widget;
    GtkCalendarPrivate *priv;
} GtkCalendar;
typedef struct _GtkCalendarClass {
    GtkWidgetClass parent_class;
    void (*month_changed) (GtkCalendar * calendar);
    void (*day_selected) (GtkCalendar * calendar);
    void (*day_selected_double_click) (GtkCalendar * calendar);
    void (*prev_month) (GtkCalendar * calendar);
    void (*next_month) (GtkCalendar * calendar);
    void (*prev_year) (GtkCalendar * calendar);
    void (*next_year) (GtkCalendar * calendar);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCalendarClass;
typedef struct _GtkCalendarPrivate GtkCalendarPrivate;
typedef enum {
    GTK_CALENDAR_SHOW_HEADING,
    GTK_CALENDAR_SHOW_DAY_NAMES,
    GTK_CALENDAR_NO_MONTH_CHANGE,
    GTK_CALENDAR_SHOW_WEEK_NUMBERS,
    GTK_CALENDAR_SHOW_DETAILS
} GtkCalendarDisplayOptions;
typedef gchar *(*GtkCalendarDetailFunc) (GtkCalendar * calendar,
                                           guint year, guint month,
                                           guint day, gpointer user_data);
typedef struct _GtkCellRendererProgress {
    GtkCellRenderer parent_instance;
    GtkCellRendererProgressPrivate *priv;
} GtkCellRendererProgress;
typedef struct _GtkCellRendererProgressClass {
    GtkCellRendererClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererProgressClass;
typedef struct _GtkCellRendererProgressPrivate
    GtkCellRendererProgressPrivate;
typedef enum {
    GTK_DIALOG_MODAL = 1 << 0,
    GTK_DIALOG_DESTROY_WITH_PARENT = 1 << 1
} GtkDialogFlags;
typedef enum {
    GTK_RESPONSE_NONE = -1,
    GTK_RESPONSE_REJECT = -2,
    GTK_RESPONSE_ACCEPT = -3,
    GTK_RESPONSE_DELETE_EVENT = -4,
    GTK_RESPONSE_OK = -5,
    GTK_RESPONSE_CANCEL = -6,
    GTK_RESPONSE_CLOSE = -7,
    GTK_RESPONSE_YES = -8,
    GTK_RESPONSE_NO = -9,
    GTK_RESPONSE_APPLY = -10,
    GTK_RESPONSE_HELP = -11
} GtkResponseType;

```

```

typedef struct _GtkDialog {
    GtkWindow window;
    GtkDialogPrivate *priv;
} GtkDialog;
typedef struct _GtkDialogPrivate GtkDialogPrivate;
typedef struct _GtkDialogClass {
    GtkWindowClass parent_class;
    void (*response) (GtkDialog, gint);
    void (*close) (GtkDialog);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkDialogClass;
typedef struct _GtkActionableInterface {
    GTypeInterface g_iface;
    const gchar *(*get_action_name) (GtkActionable * actionable);
    void (*set_action_name) (GtkActionable * actionable,
                             const gchar * action_name);
    GVariant *(*get_action_target_value) (GtkActionable * actionable);
    void (*set_action_target_value) (GtkActionable * actionable,
                                     GVariant * target_value);
} GtkActionableInterface;
typedef struct _GtkActionable GtkActionable;
typedef struct _GtkListStore {
    GObject parent;
    GtkListStorePrivate *priv;
} GtkListStore;
typedef struct _GtkListStorePrivate GtkListStorePrivate;
typedef struct _GtkListStoreClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkListStoreClass;
typedef enum {
    GTK_RECENT_SORT_NONE = 0,
    GTK_RECENT_SORT_MRU = 1,
    GTK_RECENT_SORT_LRU = 2,
    GTK_RECENT_SORT_CUSTOM = 3
} GtkRecentSortType;
typedef gint (*GtkRecentSortFunc) (GtkRecentInfo * a, GtkRecentInfo
* b,
                                gpointer user_data);
typedef struct _GtkRecentChooser GtkRecentChooser;
typedef struct _GtkRecentChooserIface {
    GTypeInterface base_iface;
    gboolean(*set_current_uri) (GtkRecentChooser * chooser,
                               const gchar * uri, GError * *error);
    gchar *(*get_current_uri) (GtkRecentChooser * chooser);
    gboolean(*select_uri) (GtkRecentChooser * chooser, const gchar
* uri,
                        GError * *error);
    void (*unselect_uri) (GtkRecentChooser * chooser, const gchar *
uri);
    void (*select_all) (GtkRecentChooser * chooser);
    void (*unselect_all) (GtkRecentChooser * chooser);
    GList *(*get_items) (GtkRecentChooser * chooser);
    GtkRecentManager *(*get_recent_manager) (GtkRecentChooser *
chooser);
    void (*add_filter) (GtkRecentChooser * chooser,
                       GtkRecentFilter * filter);
    void (*remove_filter) (GtkRecentChooser * chooser,
                          GtkRecentFilter * filter);

```

```

    GSList *(*list_filters) (GtkRecentChooser * chooser);
    void (*set_sort_func) (GtkRecentChooser * chooser,
                           GtkRecentSortFunc sort_func,
                           gpointer
sort_data,
                           GDestroyNotify data_destroy);
    void (*item_activated) (GtkRecentChooser * chooser);
    void (*selection_changed) (GtkRecentChooser * chooser);
} GtkRecentChooserInterface;
typedef enum {
    GTK_RECENT_CHOOSER_ERROR_NOT_FOUND,
    GTK_RECENT_CHOOSER_ERROR_INVALID_URI
} GtkRecentChooserError;
typedef struct _GtkScrollable GtkScrollable;
typedef struct _GtkScrollableInterface {
    GTypeInterface base_iface;
} GtkScrollableInterface;
typedef struct _GtkRadioAction {
    GtkToggleAction parent;
    GtkRadioActionPrivate *private_data;
} GtkRadioAction;
typedef struct _GtkRadioActionPrivate GtkRadioActionPrivate;
typedef struct _GtkRadioActionClass {
    GtkToggleActionClass parent_class;
    void (*changed) (GtkRadioAction * action, GtkRadioAction *
current);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRadioActionClass;
typedef struct _GtkRecentChooserWidget {
    GtkWidget parent_instance;
    GtkRecentChooserWidgetPrivate *priv;
} GtkRecentChooserWidget;
typedef struct _GtkRecentChooserWidgetClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRecentChooserWidgetClass;
typedef struct _GtkRecentChooserWidgetPrivate
GtkRecentChooserWidgetPrivate;
typedef struct _GtkPlug {
    GtkWidget window;
    GtkPlugPrivate *priv;
} GtkPlug;
typedef struct _GtkPlugPrivate GtkPlugPrivate;
typedef struct _GtkPlugClass {
    GtkWidgetClass parent_class;
    void (*embedded) (GtkPlug * plug);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkPlugClass;
typedef void (*GtkClipboardReceivedFunc) (GtkClipboard * clipboard,
                                           GtkSelectionData *
selection_data, gpointer data);
typedef void (*GtkClipboardTextReceivedFunc) (GtkClipboard *
clipboard,
                                              const gchar * text,
                                              gpointer data);
typedef void (*GtkClipboardRichTextReceivedFunc) (GtkClipboard *
clipboard,
                                                  GdkAtom format,

```

```

        const guint8 * text,
        gsize length,
        gpointer data);
typedef void (*GtkClipboardImageReceivedFunc) (GtkClipboard *
clipboard,
        GdkPixbuf * pixbuf,
        gpointer data);
typedef void (*GtkClipboardURIReturnedFunc) (GtkClipboard *
clipboard,
        gchar * *uris, gpointer data);
typedef void (*GtkClipboardTargetsReceivedFunc) (GtkClipboard *
clipboard,
        GdkAtom * atoms,
        gint n_atoms,
        gpointer data);
typedef void (*GtkClipboardGetFunc) (GtkClipboard * clipboard,
        GtkSelectionData * selection_data,
        guint info,
        gpointer user_data_or_owner);
typedef void (*GtkClipboardClearFunc) (GtkClipboard * clipboard,
        gpointer user_data_or_owner);
typedef enum {
    GTK_NOTEBOOK_TAB_FIRST,
    GTK_NOTEBOOK_TAB_LAST
} GtkNotebookTab;
typedef struct _GtkNotebook {
    GtkContainer widget;
    GtkNotebookPrivate *priv;
} GtkNotebook;
typedef struct _GtkNotebookPrivate GtkNotebookPrivate;
typedef struct _GtkNotebookClass {
    GtkContainerClass parent_class;
    void (*switch_page) (GtkNotebook * notebook, GtkWidget * page,
        guint page_num);
    gboolean(*select_page) (GtkNotebook * notebook, gboolean
move_focus);
    gboolean(*focus_tab) (GtkNotebook * notebook, GtkNotebookTab
type);
    gboolean(*change_current_page) (GtkNotebook * notebook, gint
offset);
    void (*move_focus_out) (GtkNotebook * notebook,
        GtkDirectionType direction);
    gboolean(*reorder_tab) (GtkNotebook * notebook,
        GtkDirectionType direction,
        gboolean move_to_last);
    gint(*insert_page) (GtkNotebook * notebook, GtkWidget * child,
        GtkWidget * tab_label, GtkWidget * menu_label,
        gint position);
    GtkNotebook *(*create_window) (GtkNotebook * notebook,
        GtkWidget * page, gint x, gint y);
    void (*page_reordered) (GtkNotebook * notebook, GtkWidget * page,
        guint page_num);
    void (*page_removed) (GtkNotebook * notebook, GtkWidget * page,
        guint page_num);
    void (*page_added) (GtkNotebook * notebook, GtkWidget * page,
        guint page_num);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkNotebookClass;
typedef struct _GtkLockButton {

```

```

    GtkButton parent;
    GtkLockButtonPrivate *priv;
} GtkLockButton;
typedef struct _GtkLockButtonClass {
    GtkButtonClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkLockButtonClass;
typedef struct _GtkLockButtonPrivate GtkLockButtonPrivate;
typedef struct _GtkTreeDragSource GtkTreeDragSource;
typedef struct _GtkTreeDragSourceIface {
    GTypeInterface g_iface;
    gboolean(*row_draggable) (GtkTreeDragSource * drag_source,
                               GtkTreePath * path);
    gboolean(*drag_data_get) (GtkTreeDragSource * drag_source,
                               GtkTreePath * path,
                               GtkSelectionData * selection_data);
    gboolean(*drag_data_delete) (GtkTreeDragSource * drag_source,
                                  GtkTreePath * path);
} GtkTreeDragSourceIface;
typedef struct _GtkTreeDragDest GtkTreeDragDest;
typedef struct _GtkTreeDragDestIface {
    GTypeInterface g_iface;
    gboolean(*drag_data_received) (GtkTreeDragDest * drag_dest,
                                    GtkTreePath * dest,
                                    GtkSelectionData * selection_data);
    gboolean(*row_drop_possible) (GtkTreeDragDest * drag_dest,
                                   GtkTreePath * dest,
                                   GtkSelectionData * selection_data);
} GtkTreeDragDestIface;
typedef struct _GtkLayout {
    GtkContainer container;
    GtkLayoutPrivate *priv;
} GtkLayout;
typedef struct _GtkLayoutPrivate GtkLayoutPrivate;
typedef struct _GtkLayoutClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkLayoutClass;
typedef struct _GtkGrid {
    GtkContainer container;
    GtkGridPrivate *priv;
} GtkGrid;
typedef struct _GtkGridPrivate GtkGridPrivate;
typedef struct _GtkGridClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkGridClass;
typedef struct _GtkEntryBuffer {
    GObject parent_instance;

```

```

    GtkEntryBufferPrivate *priv;
} GtkEntryBuffer;
typedef struct _GtkEntryBufferClass {
    GObjectClass parent_class;
    void (*inserted_text) (GtkEntryBuffer * buffer, guint position,
                           const gchar * chars, guint n_chars);
    void (*deleted_text) (GtkEntryBuffer * buffer, guint position,
                           guint n_chars);
    const gchar *(*get_text) (GtkEntryBuffer * buffer, gsize *
                               n_bytes);
    guint(*get_length) (GtkEntryBuffer * buffer);
    guint(*insert_text) (GtkEntryBuffer * buffer, guint position,
                           const gchar * chars, guint n_chars);
    guint(*delete_text) (GtkEntryBuffer * buffer, guint position,
                           guint n_chars);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkEntryBufferClass;
typedef struct _GtkEntryBufferPrivate GtkEntryBufferPrivate;
typedef struct _GtkRequestedSize {
    gint data;
    gpointer minimum_size;
    gpointer natural_size;
} GtkRequestedSize;
typedef struct _GtkFileFilter GtkFileFilter;
typedef struct _GtkFileFilterInfo {
    GtkFileFilterFlags contains;
    const gchar *filename;
    const gchar *uri;
    const gchar *display_name;
    const gchar *mime_type;
} GtkFileFilterInfo;
typedef enum {
    GTK_FILE_FILTER_FILENAME,
    GTK_FILE_FILTER_URI,
    GTK_FILE_FILTER_DISPLAY_NAME,
    GTK_FILE_FILTER_MIME_TYPE
} GtkFileFilterFlags;
typedef gboolean(*GtkFileFilterFunc) (const GtkFileFilterInfo *
                                       filter_info, gpointer data);
typedef struct _GtkThemingEngine {
    GObject parent_object;
    GtkThemingEnginePrivate *priv;
} GtkThemingEngine;
typedef struct GtkThemingEnginePrivate GtkThemingEnginePrivate;
typedef struct _GtkThemingEngineClass {
    GObjectClass parent_class;
    void (*render_line) (GtkThemingEngine * engine, cairo_t * cr,
                           gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_background) (GtkThemingEngine * engine, cairo_t *
cr,
                               gdouble x0, gdouble y0, gdouble x1,
                               gdouble y1);
    void (*render_frame) (GtkThemingEngine * engine, cairo_t * cr,
                           gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_frame_gap) (GtkThemingEngine * engine, cairo_t *
cr,
                               gdouble x, gdouble y, gdouble width,

```



```

        gdouble height, GtkPositionType gap_side,
        gdouble xy0_gap, gdouble xyl_gap);
    void (*render_extension) (GtkThemingEngine * engine, cairo_t *
cr,
        gdouble x, gdouble y, gdouble width,
        gdouble height, GtkPositionType gap_side);
    void (*render_check) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_option) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_arrow) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_expander) (GtkThemingEngine * engine, cairo_t *
cr,
        gdouble x0, gdouble y0, gdouble x1,
        gdouble y1);
    void (*render_focus) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_layout) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x, gdouble y, PangoLayout * layout);
    void (*render_slider) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x, gdouble y, gdouble width,
        gdouble height, GtkOrientation orientation);
    void (*render_handle) (GtkThemingEngine * engine, cairo_t * cr,
        gdouble x0, gdouble y0, gdouble x1, gdouble
y1);
    void (*render_activity) (GtkThemingEngine * engine, cairo_t *
cr,
        gdouble x0, gdouble y0, gdouble x1,
        gdouble y1);
    GdkPixbuf *(*render_icon_pixbuf) (GtkThemingEngine * engine,
        const GtkIconSource * source,
        GtkIconSize size);
    void (*render_icon) (GtkThemingEngine * engine, cairo_t * cr,
        GdkPixbuf * pixbuf, gdouble x, gdouble y);
    gpointer padding;
} GtkThemingEngineClass;
typedef gboolean(*GtkFontFilterFunc) (const PangoFontFamily *
family,
        const PangoFontFace * face,
        gpointer data);
typedef struct _GtkFontChooser GtkFontChooser;
typedef struct _GtkFontChooserIface {
    GTypeInterface base_iface;
    PangoFontFamily *(*get_font_family) (GtkFontChooser *
fontchooser);
    PangoFontFace *(*get_font_face) (GtkFontChooser * fontchooser);
    gint(*get_font_size) (GtkFontChooser * fontchooser);
    void(*set_filter_func) (GtkFontChooser * fontchooser,
        GtkFontFilterFunc filter, gpointer
user_data,
        GDestroyNotify destroy);
    void(*font_activated) (GtkFontChooser * chooser,
        const gchar * fontname);
    gpointer padding;
} GtkFontChooserIface;
typedef struct _GtkAppChooserButton {
    GtkComboBox parent;
    GtkAppChooserButtonPrivate *priv;
} GtkAppChooserButton;
typedef struct _GtkAppChooserButtonClass {
    GtkComboBoxClass parent_class;

```

```

        void (*custom_item_activated) (GtkAppChooserButton * self,
                                        const gchar * item_name);

        gpointer padding;
    } GtkAppChooserButtonClass;
typedef struct _GtkAppChooserButtonPrivate
GtkAppChooserButtonPrivate;
typedef struct _GtkCellRendererSpinner {
    GtkCellRenderer parent;
    GtkCellRendererSpinnerPrivate *priv;
} GtkCellRendererSpinner;
typedef struct _GtkCellRendererSpinnerClass {
    GtkCellRendererClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererSpinnerClass;
typedef struct _GtkCellRendererSpinnerPrivate
GtkCellRendererSpinnerPrivate;
typedef struct _GtkMountOperation {
    GMountOperation parent_instance;
    GtkMountOperationPrivate *priv;
} GtkMountOperation;
typedef struct _GtkMountOperationClass {
    GMountOperationClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMountOperationClass;
typedef struct _GtkMountOperationPrivate GtkMountOperationPrivate;
typedef enum {
    GTK_WIDGET_HELP_TOOLTIP,
    GTK_WIDGET_HELP_WHATS_THIS
} GtkWidgetHelpType;
typedef struct _GtkWidgetPrivate GtkWidgetPrivate;
typedef struct _GtkWidgetClass {
    GInitiallyUnownedClass parent_class;
    guint activate_signal;
    void (*dispatch_child_properties_changed) (GtkWidget * widget,
                                                guint n_specs,
                                                GParamSpec * *pspecs);

    void (*destroy) (GtkWidget * widget);
    void (*show) (GtkWidget * widget);
    void (*show_all) (GtkWidget * widget);
    void (*hide) (GtkWidget * widget);
    void (*map) (GtkWidget * widget);
    void (*unmap) (GtkWidget * widget);
    void (*realize) (GtkWidget * widget);
    void (*unrealize) (GtkWidget * widget);
    void (*size_allocate) (GtkWidget * widget, GtkAllocation *
allocation);
    void (*state_changed) (GtkWidget * widget,
                           GtkStateType previous_state);
    void (*state_flags_changed) (GtkWidget * widget,
                                 GtkStateFlags previous_state_flags);
    void (*parent_set) (GtkWidget * widget, GtkWidget *
previous_parent);
    void (*hierarchy_changed) (GtkWidget * widget,
                                GtkWidget * previous_parent);
    void (*style_set) (GtkWidget * widget, GtkStyle *
previous_style);
    void (*direction_changed) (GtkWidget * widget,
                               GtkTextDirection previous_direction);
    void (*grab_notify) (GtkWidget * widget, gboolean was_grabbed);

```

```

void (*child_notify) (GtkWidget * widget, GParamSpec *
child_property);
gboolean(*draw) (GtkWidget * widget, cairo_t * cr);
GtkSizeMode(*get_request_mode) (GtkWidget * widget);
void (*get_preferred_height) (GtkWidget * widget,
gint * minimum_height,
gint * natural_height);
void (*get_preferred_width_for_height) (GtkWidget * widget,
gint height,
gint * minimum_width,
gint * natural_width);
void (*get_preferred_width) (GtkWidget * widget, gint *
minimum_height,
gint * natural_height);
void (*get_preferred_height_for_width) (GtkWidget * widget,
gint height,
gint * minimum_width,
gint * natural_width);
gboolean(*mnemonic_activate) (GtkWidget * widget,
gboolean group_cycling);
void (*grab_focus) (GtkWidget * widget);
gboolean(*focus) (GtkWidget * widget, GtkDirectionType
direction);
void (*move_focus) (GtkWidget * widget, GtkDirectionType
direction);
gboolean(*keynav_failed) (GtkWidget * widget,
GtkDirectionType direction);
gboolean(*event) (GtkWidget * widget, GdkEvent * event);
gboolean(*button_press_event) (GtkWidget * widget,
GdkEventKey * event);
gboolean(*button_release_event) (GtkWidget * widget,
GdkEventKey * event);
gboolean(*scroll_event) (GtkWidget * widget, GdkEventScroll *
event);
gboolean(*motion_notify_event) (GtkWidget * widget,
GdkEventMotion * event);
gboolean(*delete_event) (GtkWidget * widget, GdkEventAny *
event);
gboolean(*destroy_event) (GtkWidget * widget, GdkEventAny *
event);
gboolean(*key_press_event) (GtkWidget * widget, GdkEventKey *
event);
gboolean(*key_release_event) (GtkWidget * widget,
GdkEventKey * event);
gboolean(*enter_notify_event) (GtkWidget * widget,
GdkEventCrossing * event);
gboolean(*leave_notify_event) (GtkWidget * widget,
GdkEventCrossing * event);
gboolean(*configure_event) (GtkWidget * widget,
GdkEventConfigure * event);
gboolean(*focus_in_event) (GtkWidget * widget, GdkEventFocus *
event);
gboolean(*focus_out_event) (GtkWidget * widget,
GdkEventFocus * event);
gboolean(*map_event) (GtkWidget * widget, GdkEventAny * event);
gboolean(*unmap_event) (GtkWidget * widget, GdkEventAny *
event);
gboolean(*property_notify_event) (GtkWidget * widget,
GdkEventProperty * event);
gboolean(*selection_clear_event) (GtkWidget * widget,
GdkEventSelection * event);
gboolean(*selection_request_event) (GtkWidget * widget,
GdkEventSelection * event);
gboolean(*selection_notify_event) (GtkWidget * widget,
GdkEventSelection * event);
gboolean(*proximity_in_event) (GtkWidget * widget,

```

```

        GdkEventProximity * event);
gboolean(*proximity_out_event) (GtkWidget * widget,
        GdkEventProximity * event);
gboolean(*visibility_notify_event) (GtkWidget * widget,
        GdkEventVisibility * event);
gboolean(*window_state_event) (GtkWidget * widget,
        GdkEventWindowState * event);
gboolean(*damage_event) (GtkWidget * widget, GdkEventExpose *
event);
gboolean(*grab_broken_event) (GtkWidget * widget,
        GdkEventGrabBroken * event);
void (*selection_get) (GtkWidget * widget,
        GtkSelectionData * selection_data, guint
info,
        guint time_);
void (*selection_received) (GtkWidget * widget,
        GtkSelectionData * selection_data,
        guint time_);
void (*drag_begin) (GtkWidget * widget, GdkDragContext *
context);
void (*drag_end) (GtkWidget * widget, GdkDragContext * context);
void (*drag_data_get) (GtkWidget * widget, GdkDragContext *
context,
        GtkSelectionData * selection_data, guint
info,
        guint time_);
void (*drag_data_delete) (GtkWidget * widget,
        GdkDragContext * context);
void (*drag_leave) (GtkWidget * widget, GdkDragContext * context,
        guint time_);
gboolean(*drag_motion) (GtkWidget * widget, GdkDragContext *
context,
        gint x, gint y, guint time_);
gboolean(*drag_drop) (GtkWidget * widget, GdkDragContext *
context,
        gint x, gint y, guint time_);
void (*drag_data_received) (GtkWidget * widget,
        GdkDragContext * context, gint x, gint
y,
        GtkSelectionData * selection_data,
        guint info, guint time_);
gboolean(*drag_failed) (GtkWidget * widget, GdkDragContext *
context,
        GtkDragResult result);
gboolean(*popup_menu) (GtkWidget * widget);
gboolean(*show_help) (GtkWidget * widget,
        GtkWidgetHelpType help_type);
AtkObject *(*get_accessible) (GtkWidget * widget);
void (*screen_changed) (GtkWidget * widget,
        GdkScreen * previous_screen);
gboolean(*can_activate_accel) (GtkWidget * widget, guint
signal_id);
void (*composited_changed) (GtkWidget * widget);
gboolean(*query_tooltip) (GtkWidget * widget, gint x, gint y,
        gboolean keyboard_tooltip,
        GtkTooltip * tooltip);
void (*compute_expand) (GtkWidget * widget, gboolean * hexpand_p,
        gboolean * vexpand_p);
void (*adjust_size_request) (GtkWidget * widget,
        GtkOrientation orientation,
        gint * minimum_size, gint *
natural_size);
void (*adjust_size_allocation) (GtkWidget * widget,
        GtkOrientation orientation,
        gint * minimum_size,
        gint * natural_size,

```

```

        gint * allocated_pos,
        gint * allocated_size);
void (*style_updated) (GtkWidget * widget);
gboolean(*touch_event) (GtkWidget * widget, GdkEventTouch *
event);
GtkWidgetClassPrivate *priv;
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
void (*_gtk_reserved5) (void);
void (*_gtk_reserved6) (void);
void (*_gtk_reserved7) (void);
} GtkWidgetClass;
typedef struct _GtkWidgetClassPrivate GtkWidgetClassPrivate;
typedef struct _GtkWidgetAuxInfo {
    gint width;
    gint height;
    guint halign:4;
    guint valign:4;
    GtkBorder margin;
} GtkWidgetAuxInfo;
typedef struct _cairo_rectangle_int GtkAllocation;
typedef void (*GtkCallback) (void);
typedef struct _GtkTreeStore {
    GObject parent;
    GtkTreeStorePrivate *priv;
} GtkTreeStore;
typedef struct _GtkTreeStoreClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTreeStoreClass;
typedef struct _GtkTreeStorePrivate GtkTreeStorePrivate;
typedef gchar *(*GtkTranslateFunc) (const char *path, gpointer
func_data);
typedef struct _GtkStockItem {
    gchar *stock_id;
    gchar *label;
    GdkModifierType modifier;
    guint keyval;
    gchar *translation_domain;
} GtkStockItem;
typedef enum {
    GTK_ENTRY_ICON_PRIMARY,
    GTK_ENTRY_ICON_SECONDARY
} GtkEntryIconPosition;
typedef struct _GtkEntry {
    GtkWidget parent_instance;
    GtkEntryPrivate *priv;
} GtkEntry;
typedef struct _GtkEntryPrivate GtkEntryPrivate;
typedef struct _GtkEntryClass {
    GtkWidgetClass parent_class;
    void (*populate_popup) (GtkEntry * entry, GtkMenu * menu);
    void (*activate) (GtkEntry * entry);
    void (*move_cursor) (GtkEntry * entry, GtkMovementStep step,
        gint count, gboolean extend_selection);
    void (*insert_at_cursor) (GtkEntry * entry, const gchar * str);
    void (*delete_from_cursor) (GtkEntry * entry, GtkDeleteType
type,
        gint count);
    void (*backspace) (GtkEntry * entry);
    void (*cut_clipboard) (GtkEntry * entry);
    void (*copy_clipboard) (GtkEntry * entry);

```

```

void (*paste_clipboard) (GtkEntry * entry);
void (*toggle_overwrite) (GtkEntry * entry);
void (*get_text_area_size) (GtkEntry * entry, gint * x, gint *
y,
                                gint * width, gint * height);
void (*get_frame_size) (GtkEntry * entry, gint * x, gint * y,
                                gint * width, gint * height);
void (*_gtk_reserved1) (void);
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
void (*_gtk_reserved5) (void);
void (*_gtk_reserved6) (void);
void (*_gtk_reserved7) (void);
} GtkEntryClass;
typedef struct _GtkColorChooser GtkColorChooser;
typedef struct _GtkColorChooserInterface {
    GTypeInterface base_interface;
    void (*get_rgba) (GtkColorChooser * chooser, GdkRGBA * color);
    void (*set_rgba) (GtkColorChooser * chooser, const GdkRGBA *
color);
    void (*add_palette) (GtkColorChooser * chooser,
                                GtkOrientation orientation, gint
colors_per_line,
                                gint n_colors, GdkRGBA * colors);
    void (*color_activated) (GtkColorChooser * chooser,
                                const GdkRGBA * color);
    gpointer padding;
} GtkColorChooserInterface;
typedef struct _GtkAccelLabel {
    GtkDialog label;
    GtkAccelLabelPrivate *priv;
} GtkAccelLabel;
typedef struct _GtkAccelLabelClass {
    GtkLabelClass parent_class;
    gchar *signal_quote1;
    gchar *signal_quote2;
    gchar *mod_name_shift;
    gchar *mod_name_control;
    gchar *mod_name_alt;
    gchar *mod_separator;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAccelLabelClass;
typedef struct _GtkAccelLabelPrivate GtkAccelLabelPrivate;
typedef struct _GtkMenuButton {
    GtkToggleButton parent;
    GtkMenuButtonPrivate *priv;
} GtkMenuButton;
typedef struct _GtkMenuButtonClass {
    GtkToggleButtonClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMenuButtonClass;
typedef struct _GtkMenuButtonPrivate GtkMenuButtonPrivate;
typedef struct _GtkAppChooserDialog {
    GtkDialog parent;
    GtkAppChooserDialogPrivate *priv;
} GtkAppChooserDialog;
typedef struct _GtkAppChooserDialogClass {
    GtkDialogClass parent_class;
    gpointer padding[16];

```

```

} GtkAppChooserDialogClass;
typedef struct _GtkAppChooserDialogPrivate
GtkAppChooserDialogPrivate;
typedef struct _GtkColorButton {
    GtkButton button;
    GtkColorButtonPrivate *priv;
} GtkColorButton;
typedef struct _GtkColorButtonClass {
    GtkButtonClass parent_class;
    void (*color_set) (GtkColorButton * cp);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkColorButtonClass;
typedef struct _GtkColorButtonPrivate GtkColorButtonPrivate;
typedef struct _GtkUIManager {
    GObject parent;
    GtkUIManagerPrivate *priv;
} GtkUIManager;
typedef struct _GtkUIManagerClass {
    GObjectClass parent_class;
    void (*add_widget) (GtkUIManager * manager, GtkWidget * widget);
    void (*actions_changed) (GtkUIManager * manager);
    void (*connect_proxy) (GtkUIManager * manager, GtkAction *
action,
                        GtkWidget * proxy);
    void (*disconnect_proxy) (GtkUIManager * manager, GtkAction *
action,
                        GtkWidget * proxy);
    void (*pre_activate) (GtkUIManager * manager, GtkAction *
action);
    void (*post_activate) (GtkUIManager * manager, GtkAction *
action);
    GtkWidget *(*get_widget) (GtkUIManager * manager, const gchar *
path);
    GtkAction *(*get_action) (GtkUIManager * manager, const gchar *
path);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkUIManagerClass;
typedef struct _GtkUIManagerPrivate GtkUIManagerPrivate;
typedef enum {
    GTK_UI_MANAGER_AUTO,
    GTK_UI_MANAGER_MENUBAR,
    GTK_UI_MANAGER_MENU,
    GTK_UI_MANAGER_TOOLBAR,
    GTK_UI_MANAGER_PLACEHOLDER,
    GTK_UI_MANAGER_POPUP,
    GTK_UI_MANAGER_MENUITEM,
    GTK_UI_MANAGER_TOOLITEM,
    GTK_UI_MANAGER_SEPARATOR,
    GTK_UI_MANAGER_ACCELERATOR,
    GTK_UI_MANAGER_POPUP_WITH_ACCELS
} GtkUIManagerItemType;
typedef struct _GtkToggleToolButton {
    GtkToolButton parent;
    GtkToggleToolButtonPrivate *priv;
} GtkToggleToolButton;
typedef struct _GtkToggleToolButtonClass {
    GtkToolButtonClass parent_class;
    void (*toggled) (GtkToggleToolButton * button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);

```

```

        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkToggleToolButtonClass;
typedef struct _GtkToggleToolButtonPrivate
GtkToggleToolButtonPrivate;
typedef struct _GtkCellRendererAccel {
    GtkCellRendererText parent;
    GtkCellRendererAccelPrivate *priv;
} GtkCellRendererAccel;
typedef struct _GtkCellRendererAccelPrivate
GtkCellRendererAccelPrivate;
typedef struct _GtkCellRendererAccelClass {
    GtkCellRendererTextClass parent_class;
    void (*accel_edited) (GtkCellRendererAccel * accel,
                          const gchar * path_string, guint accel_key,
                          GdkModifierType accel_mods,
                          guint hardware_keycode);
    void (*accel_cleared) (GtkCellRendererAccel * accel,
                          const gchar * path_string);
    void (*_gtk_reserved0) (void);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererAccelClass;
typedef enum {
    GTK_CELL_RENDERER_ACCEL_MODE_GTK,
    GTK_CELL_RENDERER_ACCEL_MODE_OTHER
} GtkCellRendererAccelMode;
typedef struct _GtkOverlay {
    GtkBin parent;
    GtkOverlayPrivate *priv;
} GtkOverlay;
typedef struct _GtkOverlayClass {
    GtkBinClass parent_class;
    gboolean(*get_child_position) (GtkOverlay * overlay,
                                   GtkWidget * widget,
                                   GtkAllocation * allocation);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkOverlayClass;
typedef struct _GtkOverlayPrivate GtkOverlayPrivate;
typedef enum {
    GTK_TEXT_WINDOW_PRIVATE,
    GTK_TEXT_WINDOW_WIDGET,
    GTK_TEXT_WINDOW_TEXT,
    GTK_TEXT_WINDOW_LEFT,
    GTK_TEXT_WINDOW_RIGHT,
    GTK_TEXT_WINDOW_TOP,
    GTK_TEXT_WINDOW_BOTTOM
} GtkTextWindowType;
typedef struct _GtkTextView {
    GtkContainer parent_instance;
    GtkTextViewPrivate *priv;
} GtkTextView;
typedef struct _GtkTextViewPrivate GtkTextViewPrivate;
typedef struct _GtkTextViewClass {
    GtkContainerClass parent_class;
    void (*populate_popup) (GtkTextView * text_view, GtkMenu * menu);

```



```

void (*move_cursor) (GtkTextView * text_view, GtkMovementStep
step,
                        gint count, gboolean extend_selection);
void (*set_anchor) (GtkTextView * text_view);
void (*insert_at_cursor) (GtkTextView * text_view, const gchar
* str);
void (*delete_from_cursor) (GtkTextView * text_view,
                        GtkDeleteType type, gint count);
void (*backspace) (GtkTextView * text_view);
void (*cut_clipboard) (GtkTextView * text_view);
void (*copy_clipboard) (GtkTextView * text_view);
void (*paste_clipboard) (GtkTextView * text_view);
void (*toggle_overwrite) (GtkTextView * text_view);
void (*_gtk_reserved1) (void);
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
void (*_gtk_reserved5) (void);
void (*_gtk_reserved6) (void);
void (*_gtk_reserved7) (void);
void (*_gtk_reserved8) (void);
} GtkTextViewClass;
typedef struct _GtkInfoBarPrivate GtkInfoBarPrivate;
typedef struct _GtkInfoBarClass {
    GtkBoxClass parent_class;
    void (*response) (GtkInfoBar * info_bar, gint response_id);
    void (*close) (GtkInfoBar * info_bar);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkInfoBarClass;
typedef struct _GtkInfoBar {
    GtkBox parent;
    GtkInfoBarPrivate *priv;
} GtkInfoBar;
typedef struct _GtkRecentAction {
    GtkAction parent_instance;
    GtkRecentActionPrivate *priv;
} GtkRecentAction;
typedef struct _GtkRecentActionPrivate GtkRecentActionPrivate;
typedef struct _GtkRecentActionClass {
    GtkActionClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRecentActionClass;
typedef struct _GtkNumerableIcon {
    GEmblemedIcon parent;
    GtkNumerableIconPrivate *priv;
} GtkNumerableIcon;
typedef struct _GtkNumerableIconClass {
    GEmblemedIconClass parent_class;
    gpointer padding[16];
} GtkNumerableIconClass;
typedef struct _GtkNumerableIconPrivate GtkNumerableIconPrivate;
typedef void (*GtkTextTagTableForeach) (void);
typedef struct _GtkTextTagTablePrivate GtkTextTagTablePrivate;
typedef struct _GtkTextTagTableClass {
    GObjectClass parent_class;
    void (*tag_changed) (GtkTextTagTable * table, GtkTextTag * tag,
                        gboolean size_changed);
    void (*tag_added) (GtkTextTagTable * table, GtkTextTag * tag);
    void (*tag_removed) (GtkTextTagTable * table, GtkTextTag * tag);
    void (*_gtk_reserved1) (void);

```

```

        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkTextTagTableClass;
typedef struct _GtkRecentFilter GtkRecentFilter;
typedef struct _GtkRecentFilterInfo {
    GtkRecentFilterFlags contains;
    const gchar *uri;
    const gchar *display_name;
    const gchar *mime_type;
    const gchar **applications;
    const gchar **groups;
    gint age;
} GtkRecentFilterInfo;
typedef enum {
    GTK_RECENT_FILTER_URI = 1 << 0,
    GTK_RECENT_FILTER_DISPLAY_NAME = 1 << 1,
    GTK_RECENT_FILTER_MIME_TYPE = 1 << 2,
    GTK_RECENT_FILTER_APPLICATION = 1 << 3,
    GTK_RECENT_FILTER_GROUP = 1 << 4,
    GTK_RECENT_FILTER_AGE = 1 << 5
} GtkRecentFilterFlags;
typedef gboolean(*GtkRecentFilterFunc) (const GtkRecentFilterInfo
*,
                                     filter_info, gpointer user_data);

typedef struct _GtkSwitch {
    GtkWidget parent_instance;
    GtkSwitchPrivate *priv;
} GtkSwitch;
typedef struct _GtkSwitchPrivate GtkSwitchPrivate;
typedef struct _GtkSwitchClass {
    GtkWidgetClass parent_class;
    void (*activate) (GtkSwitch *sw);
    void (*_switch_padding_1) (void);
    void (*_switch_padding_2) (void);
    void (*_switch_padding_3) (void);
    void (*_switch_padding_4) (void);
    void (*_switch_padding_5) (void);
    void (*_switch_padding_6) (void);
} GtkSwitchClass;
typedef enum {
    GTK_CELL_RENDERER_SELECTED = 1 << 0,
    GTK_CELL_RENDERER_PRELIT = 1 << 1,
    GTK_CELL_RENDERER_INSENSITIVE = 1 << 2,
    GTK_CELL_RENDERER_SORTED = 1 << 3,
    GTK_CELL_RENDERER_FOCUSED = 1 << 4,
    GTK_CELL_RENDERER_EXPANDABLE = 1 << 5,
    GTK_CELL_RENDERER_EXPANDED = 1 << 6
} GtkCellRendererState;
typedef enum {
    GTK_CELL_RENDERER_MODE_INERT,
    GTK_CELL_RENDERER_MODE_ACTIVATABLE,
    GTK_CELL_RENDERER_MODE_EDITABLE
} GtkCellRendererMode;
typedef struct _GtkCellRenderer {
    GInitiallyUnowned parent_instance;
    GtkCellRendererPrivate *priv;
} GtkCellRenderer;
typedef struct _GtkCellRendererPrivate GtkCellRendererPrivate;
typedef struct _GtkCellRendererClass {
    GInitiallyUnownedClass parent_class;
    GtkSizeRequestMode(*get_request_mode) (GtkCellRenderer * cell);
    void (*get_preferred_width) (GtkCellRenderer * cell,
                                GtkWidget * widget, gint * minimum_size,
                                gint * natural_size);
    void (*get_preferred_height_for_width) (GtkCellRenderer * cell,

```

```

        GtkWidget * widget, gint width,
        gint * minimum_height,
        gint * natural_height);
void (*get_preferred_height) (GtkCellRenderer * cell,
        GtkWidget * widget, gint *
minimum_size,
        gint * natural_size);
void (*get_preferred_width_for_height) (GtkCellRenderer * cell,
        GtkWidget * widget, gint width,
        gint * minimum_height,
        gint * natural_height);
void (*get_aligned_area) (GtkCellRenderer * cell, GtkWidget *
widget,
        GtkCellRendererState flags,
        const GdkRectangle * cell_area,
        GdkRectangle * aligned_area);
void (*get_size) (GtkCellRenderer * cell, GtkWidget * widget,
        const GdkRectangle * cell_area, gint * x_offset,
        gint * y_offset, gint * width, gint * height);
void (*render) (GtkCellRenderer * cell, cairo_t * cr,
        GtkWidget * widget,
        const GdkRectangle * background_area,
        const GdkRectangle * cell_area,
        GtkCellRendererState flags);
gboolean(*activate) (GtkCellRenderer * cell, GdkEvent * event,
        GtkWidget * widget, const gchar * path,
        const GdkRectangle * background_area,
        const GdkRectangle * cell_area,
        GtkCellRendererState flags);
GtkCellEditable *(*start_editing) (GtkCellRenderer * cell,
        GdkEvent * event,
        GtkWidget * widget,
        const gchar * path,
        const GdkRectangle *
        background_area,
        const GdkRectangle * cell_area,
        GtkCellRendererState flags);
void (*editing_canceled) (GtkCellRenderer * cell);
void (*editing_started) (GtkCellRenderer * cell,
        GtkCellEditable * editable,
        const gchar * path);
GtkCellRendererClassPrivate *priv;
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
} GtkCellRendererClass;
typedef struct _GtkCellRendererClassPrivate _GtkCellRendererClassPrivate
GtkCellRendererClassPrivate;
typedef struct _GtkTargetList GtkTargetList;
typedef struct _GtkTargetEntry {
    gchar *target;
    guint flags;
    guint info;
} GtkTargetEntry;
typedef struct _GtkMenuBar {
    GtkMenuShell menu_shell;
    GtkMenuBarPrivate *priv;
} GtkMenuBar;
typedef struct _GtkMenuBarPrivate GtkMenuBarPrivate;
typedef struct _GtkMenuBarClass {
    GtkMenuShellClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMenuBarClass;

```

```

typedef struct _GtkFrame {
    GtkBin bin;
    GtkFramePrivate *priv;
} GtkFrame;
typedef struct _GtkFramePrivate GtkFramePrivate;
typedef struct _GtkFrameClass {
    GtkBinClass parent_class;
    void (*compute_child_allocation) (GtkFrame * frame,
                                      GtkAllocation * allocation);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFrameClass;
typedef enum {
    GTK_ASSISTANT_PAGE_CONTENT = 0,
    GTK_ASSISTANT_PAGE_INTRO = 1,
    GTK_ASSISTANT_PAGE_CONFIRM = 2,
    GTK_ASSISTANT_PAGE_SUMMARY = 3,
    GTK_ASSISTANT_PAGE_PROGRESS = 4,
    GTK_ASSISTANT_PAGE_CUSTOM = 5
} GtkAssistantPageType;
typedef struct _GtkAssistant {
    GtkWindow parent;
    GtkAssistantPrivate *priv;
} GtkAssistant;
typedef struct _GtkAssistantPrivate GtkAssistantPrivate;
typedef struct _GtkAssistantClass {
    GtkWindowClass parent_class;
    void (*prepare) (GtkAssistant * assistant, GtkWidget * page);
    void (*apply) (GtkAssistant * assistant);
    void (*close) (GtkAssistant * assistant);
    void (*cancel) (GtkAssistant * assistant);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
} GtkAssistantClass;
typedef gint (*GtkAssistantPageFunc) (gint current_page, gpointer
data);
typedef struct _GtkBorder {
    gint16 left;
    gint16 right;
    gint16 top;
    gint16 bottom;
} GtkBorder;
typedef struct _GtkToolItem {
    GtkBin parent;
    GtkToolItemPrivate *priv;
} GtkToolItem;
typedef struct _GtkToolItemClass {
    GtkBinClass parent_class;
    gboolean (*create_menu_proxy) (GtkToolItem * tool_item);
    void (*toolbar_reconfigured) (GtkToolItem * tool_item);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToolItemClass;
typedef struct _GtkToolItemPrivate GtkToolItemPrivate;
typedef struct _GtkTextAttributes {
    guint refcount;
    GtkTextAppearance appearance;
    GtkJustification justification;
    GtkTextDirection direction;

```

```

PangoFontDescription *font;
gdouble font_scale;
gint left_margin;
gint right_margin;
gint indent;
gint pixels_above_lines;
gint pixels_below_lines;
gint pixels_inside_wrap;
PangoTabArray *tabs;
GtkWrapMode *wrap_mode;
PangoLanguage *language;
GdkColor *pg_bg_color;
guint invisible:1;
guint bg_full_height:1;
guint editable:1;
GdkRGBA *pg_bg_rgba;
guint padding[3];
} GtkTextAttributes;
typedef struct _GtkTextAppearance {
    GdkColor bg_color;
    GdkColor fg_color;
    gint rise;
    guint underline:4;
    guint strikethrough:1;
    guint draw_bg:1;
    guint inside_selection:1;
    guint is_text:1;
    GdkRGBA *rgba[2];
} GtkTextAppearance;
typedef struct _GtkCellRendererSpin {
    GtkCellRendererText parent;
    GtkCellRendererSpinPrivate *priv;
} GtkCellRendererSpin;
typedef struct _GtkCellRendererSpinClass {
    GtkCellRendererTextClass parent;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererSpinClass;
typedef struct _GtkCellRendererSpinPrivate
GtkCellRendererSpinPrivate;
typedef struct _GtkIMMulticontext {
    GtkIMContext object;
    GtkIMMulticontextPrivate *priv;
} GtkIMMulticontext;
typedef struct _GtkIMMulticontextClass {
    GtkIMContextClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkIMMulticontextClass;
typedef struct _GtkIMMulticontextPrivate GtkIMMulticontextPrivate;
typedef struct _GtkCellAreaBox {
    GtkCellArea parent_instance;
    GtkCellAreaBoxPrivate *priv;
} GtkCellAreaBox;
typedef struct _GtkCellAreaBoxClass {
    GtkCellAreaClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellAreaBoxClass;
typedef struct _GtkCellAreaBoxPrivate GtkCellAreaBoxPrivate;

```

```

typedef struct _GtkFontChooserDialog {
    GtkDialog parent_instance;
    GtkFontChooserDialogPrivate *priv;
} GtkFontChooserDialog;
typedef struct _GtkFontChooserDialogPrivate
GtkFontChooserDialogPrivate;
typedef struct _GtkFontChooserDialogClass {
    GtkDialogClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFontChooserDialogClass;
typedef struct _GtkPrintOperationPreview GtkPrintOperationPreview;
typedef struct _GtkPrintOperationPreviewIface {
    GTypeInterface g_iface;
    void (*ready) (GtkPrintOperationPreview * preview,
                  GtkPrintContext * context);
    void (*got_page_size) (GtkPrintOperationPreview * preview,
                          GtkPrintContext * context,
                          GtkPageSetup * page_setup);
    void (*render_page) (GtkPrintOperationPreview * preview, gint
page_nr);
    gboolean(*is_selected) (GtkPrintOperationPreview * preview,
                           gint page_nr);
    void (*end_preview) (GtkPrintOperationPreview * preview);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkPrintOperationPreviewIface;
typedef struct _GtkToggleAction {
    GtkAction parent;
    GtkToggleActionPrivate *priv;
} GtkToggleAction;
typedef struct _GtkToggleActionPrivate GtkToggleActionPrivate;
typedef struct _GtkToggleActionClass {
    GtkActionClass parent_class;
    void (*toggled) (GtkToggleAction * action);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToggleActionClass;
typedef enum {
    GTK_CSS_PROVIDER_ERROR_FAILED,
    GTK_CSS_PROVIDER_ERROR_SYNTAX,
    GTK_CSS_PROVIDER_ERROR_IMPORT,
    GTK_CSS_PROVIDER_ERROR_NAME,
    GTK_CSS_PROVIDER_ERROR_DEPRECATED,
    GTK_CSS_PROVIDER_ERROR_UNKNOWN_VALUE
} GtkCssProviderError;
typedef struct _GtkCssProvider {
    GObject parent_instance;
    GtkCssProviderPrivate *priv;
} GtkCssProvider;
typedef struct _GtkCssProviderClass {
    GObjectClass parent_class;
    void (*parsing_error) (GtkCssProvider * provider,
                          GtkCssSection * section, const GError *
error);
    void (*_gtk_reserved2) (void);

```

```

    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCssProviderClass;
typedef struct _GtkCssProviderPrivate GtkCssProviderPrivate;
typedef gboolean(*GtkTreeModelFilterVisibleFunc) (GtkTreeModel *
model,
                                                    GtkTreeIter * iter,
                                                    gpointer data);
typedef void (*GtkTreeModelFilterModifyFunc) (GtkTreeModel * model,
                                                GtkTreeIter * iter,
                                                GValue * value, gint column,
                                                gpointer data);

typedef struct _GtkTreeModelFilter {
    GObject parent;
    GtkTreeModelFilterPrivate *priv;
} GtkTreeModelFilter;
typedef struct _GtkTreeModelFilterClass {
    GObjectClass parent_class;
    gboolean(*visible) (GtkTreeModelFilter * self,
                        GtkTreeModel * child_model, GtkTreeIter *
iter);
    void (*modify) (GtkTreeModelFilter * self, GtkTreeModel *
child_model,
                    GtkTreeIter * iter, GValue * value, gint column);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTreeModelFilterClass;
typedef struct _GtkTreeModelFilterPrivate
GtkTreeModelFilterPrivate;
typedef struct _GtkSizeGroup {
    GObject parent_instance;
    GtkSizeGroupPrivate *priv;
} GtkSizeGroup;
typedef struct _GtkSizeGroupPrivate GtkSizeGroupPrivate;
typedef struct _GtkSizeGroupClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSizeGroupClass;
typedef enum {
    GTK_SIZE_GROUP_NONE,
    GTK_SIZE_GROUP_HORIZONTAL,
    GTK_SIZE_GROUP_VERTICAL,
    GTK_SIZE_GROUP_BOTH
} GtkSizeGroupMode;
typedef struct _GtkScrollbar {
    GtkRange range;
} GtkScrollbar;
typedef struct _GtkScrollbarClass {
    GtkRangeClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkScrollbarClass;
typedef struct _GtkSocket {
    GtkContainer container;
    GtkSocketPrivate *priv;
} GtkSocket;
typedef struct _GtkSocketClass {
    GtkContainerClass parent_class;
    void (*plug_added) (GtkSocket * socket_);

```

```

        gboolean(*plug_removed) (GtkSocket * socket_);
        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkSocketClass;
typedef struct _GtkSocketPrivate GtkSocketPrivate;
typedef struct _GtkOffscreenWindow {
    GtkWidget parent_object;
} GtkOffscreenWindow;
typedef struct _GtkOffscreenWindowClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkOffscreenWindowClass;
typedef enum {
    GTK_CSS_SECTION_DOCUMENT,
    GTK_CSS_SECTION_IMPORT,
    GTK_CSS_SECTION_COLOR_DEFINITION,
    GTK_CSS_SECTION_BINDING_SET,
    GTK_CSS_SECTION_RULESET,
    GTK_CSS_SECTION_SELECTOR,
    GTK_CSS_SECTION_DECLARATION,
    GTK_CSS_SECTION_VALUE,
    GTK_CSS_SECTION_KEYFRAMES
} GtkCssSectionType;
typedef struct _GtkCssSection GtkCssSection;
typedef struct _GtkCellView {
    GtkWidget widget;
    GtkCellViewPrivate *priv;
} GtkCellView;
typedef struct _GtkCellViewClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellViewClass;
typedef struct _GtkCellViewPrivate GtkCellViewPrivate;
typedef enum {
    GTK_DEST_DEFAULT_MOTION,
    GTK_DEST_DEFAULT_HIGHLIGHT,
    GTK_DEST_DEFAULT_DROP,
    GTK_DEST_DEFAULT_ALL
} GtkDestDefaults;
typedef struct _GtkMenuItem {
    GtkBin bin;
    GtkMenuItemPrivate *priv;
} GtkMenuItem;
typedef struct _GtkMenuItemClass {
    GtkBinClass parent_class;
    guint hide_on_activate:1;
    void (*activate) (GtkMenuItem * menu_item);
    void (*activate_item) (GtkMenuItem * menu_item);
    void (*toggle_size_request) (GtkMenuItem * menu_item,
                                gint * requisition);
    void (*toggle_size_allocate) (GtkMenuItem * menu_item,
                                gint allocation);
    void (*set_label) (GtkMenuItem * menu_item, const gchar * label);
    const gchar *(*get_label) (GtkMenuItem * menu_item);
    void (*select) (GtkMenuItem * menu_item);
    void (*deselect) (GtkMenuItem * menu_item);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);

```



```

        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkMenuItemClass;
typedef struct _GtkMenuItemPrivate GtkMenuItemPrivate;
typedef struct _GtkIMContextInfo {
    const gchar *context_id;
    const gchar *context_name;
    const gchar *domain;
    const gchar *domain_dirname;
    const gchar *default_locales;
} GtkIMContextInfo;
typedef struct _GtkActionGroup {
    GObject parent;
    GtkActionGroupPrivate *priv;
} GtkActionGroup;
typedef struct _GtkActionGroupPrivate GtkActionGroupPrivate;
typedef struct _GtkActionGroupClass {
    GObjectClass parent_class;
    GtkAction *(*get_action) (GtkActionGroup * action_group,
                              const gchar * action_name);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkActionGroupClass;
typedef struct _GtkActionEntry {
    const gchar *name;
    const gchar *stock_id;
    const gchar *label;
    const gchar *accelerator;
    const gchar *tooltip;
    GCallback callback;
} GtkActionEntry;
typedef struct _GtkToggleActionEntry {
    const gchar *name;
    const gchar *stock_id;
    const gchar *label;
    const gchar *accelerator;
    const gchar *tooltip;
    GCallback callback;
    gboolean is_active;
} GtkToggleActionEntry;
typedef struct _GtkRadioActionEntry {
    const gchar *name;
    const gchar *stock_id;
    const gchar *label;
    const gchar *accelerator;
    const gchar *tooltip;
    gint value;
} GtkRadioActionEntry;
typedef struct _GtkFileChooser GtkFileChooser;
typedef enum {
    GTK_FILE_CHOOSER_ACTION_OPEN,
    GTK_FILE_CHOOSER_ACTION_SAVE,
    GTK_FILE_CHOOSER_ACTION_SELECT_FOLDER,
    GTK_FILE_CHOOSER_ACTION_CREATE_FOLDER
} GtkFileChooserAction;
typedef enum {
    GTK_FILE_CHOOSER_CONFIRMATION_CONFIRM,
    GTK_FILE_CHOOSER_CONFIRMATION_ACCEPT_FILENAME,
    GTK_FILE_CHOOSER_CONFIRMATION_SELECT_AGAIN
} GtkFileChooserConfirmation;
typedef enum {
    GTK_FILE_CHOOSER_ERROR_NONEXISTENT,
    GTK_FILE_CHOOSER_ERROR_BAD_FILENAME,
    GTK_FILE_CHOOSER_ERROR_ALREADY_EXISTS,

```

```

    GTK_FILE_CHOOSER_ERROR_INCOMPLETE_HOSTNAME
} GtkFileChooserError;
typedef struct _GtkPaned {
    GtkContainer container;
    GtkPanedPrivate *priv;
} GtkPaned;
typedef struct _GtkPanedClass {
    GtkContainerClass parent_class;
    gboolean(*cycle_child_focus) (GtkPaned * paned, gboolean
reverse);
    gboolean(*toggle_handle_focus) (GtkPaned * paned);
    gboolean(*move_handle) (GtkPaned * paned, GtkScrollType
scroll);
    gboolean(*cycle_handle_focus) (GtkPaned * paned, gboolean
reverse);
    gboolean(*accept_position) (GtkPaned * paned);
    gboolean(*cancel_position) (GtkPaned * paned);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkPanedClass;
typedef struct _GtkPanedPrivate GtkPanedPrivate;
typedef struct _GtkEditable GtkEditable;
typedef struct _GtkEditableInterface {
    GTypeInterface base_iface;
    void (*insert_text) (GtkEditable * editable, const gchar *
new_text,
                        gint new_text_length, gint * position);
    void (*delete_text) (GtkEditable * editable, gint start_pos,
                        gint end_pos);
    void (*changed) (GtkEditable * editable);
    void (*do_insert_text) (GtkEditable * editable, const gchar *
new_text,
                        gint new_text_length, gint * position);
    void (*do_delete_text) (GtkEditable * editable, gint start_pos,
                        gint end_pos);
    gchar *(*get_chars) (GtkEditable * editable, gint start_pos,
                        gint end_pos);
    void (*set_selection_bounds) (GtkEditable * editable, gint
start_pos,
                        gint end_pos);
    gboolean(*get_selection_bounds) (GtkEditable * editable,
                        gint * start_pos, gint * end_pos);
    void (*set_position) (GtkEditable * editable, gint position);
    gint(*get_position) (GtkEditable * editable);
} GtkEditableInterface;
typedef struct _GtkAction {
    GObject object;
    GtkActionPrivate *private_data;
} GtkAction;
typedef struct _GtkActionClass {
    GObjectClass parent_class;
    void (*activate) (GtkAction * action);
    GType menu_item_type;
    GType toolbar_item_type;
    GtkWidget *(*create_menu_item) (GtkAction * action);
    GtkWidget *(*create_tool_item) (GtkAction * action);
    void (*connect_proxy) (GtkAction * action, GtkWidget * proxy);
    void (*disconnect_proxy) (GtkAction * action, GtkWidget * proxy);
    GtkWidget *(*create_menu) (GtkAction * action);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkActionClass;

```

```

typedef struct _GtkActionPrivate GtkActionPrivate;
typedef struct _GtkScrolledWindow {
    GtkBin container;
    GtkScrolledWindowPrivate *priv;
} GtkScrolledWindow;
typedef struct _GtkScrolledWindowPrivate GtkScrolledWindowPrivate;
typedef struct _GtkScrolledWindowClass {
    GtkBinClass parent_class;
    gint scrollbar_spacing;
    gboolean(*scroll_child) (GtkScrolledWindow * scrolled_window,
                             GtkScrollType scroll,
                             gboolean
horizontal);
    void (*move_focus_out) (GtkScrolledWindow * scrolled_window,
                             GtkDirectionType direction);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkScrolledWindowClass;
typedef struct _GtkScale {
    GtkRange parent;
    GtkScalePrivate *priv;
} GtkScale;
typedef struct _GtkScalePrivate GtkScalePrivate;
typedef struct _GtkScaleClass {
    GtkRangeClass parent_class;
    gchar *(*format_value) (GtkScale * scale, gdouble value);
    void (*draw_value) (GtkScale * scale);
    void (*get_layout_offsets) (GtkScale * scale, gint * x, gint *
y);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkScaleClass;
typedef struct _GtkImage {
    GtkMisc misc;
    GtkImagePrivate *priv;
} GtkImage;
typedef struct _GtkImagePrivate GtkImagePrivate;
typedef struct _GtkImageClass {
    GtkMiscClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkImageClass;
typedef enum {
    GTK_IMAGE_EMPTY,
    GTK_IMAGE_PIXBUF,
    GTK_IMAGE_STOCK,
    GTK_IMAGE_ICON_SET,
    GTK_IMAGE_ANIMATION,
    GTK_IMAGE_ICON_NAME,
    GTK_IMAGE_GICON
} GtkImageType;
typedef struct _GtkCellRendererText {
    GtkCellRenderer parent;
    GtkCellRendererTextPrivate *priv;
} GtkCellRendererText;
typedef struct _GtkCellRendererTextPrivate
GtkCellRendererTextPrivate;
typedef struct _GtkCellRendererTextClass {
    GtkCellRendererClass parent_class;
    void (*edited) (GtkCellRendererText * cell_renderer_text,
                    const gchar * path, const gchar * new_text);

```

```

        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkCellRendererTextClass;
    typedef struct _GtkRadioToolButton {
        GtkToggleToolButton parent;
    } GtkRadioToolButton;
    typedef struct _GtkRadioToolButtonClass {
        GtkToggleToolButtonClass parent_class;
        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkRadioToolButtonClass;
    typedef struct _GtkTextIter {
        gpointer dummy1;
        gpointer dummy2;
        gint dummy3;
        gint dummy4;
        gint dummy5;
        gint dummy6;
        gint dummy7;
        gint dummy8;
        gpointer dummy9;
        gpointer dummy10;
        gint dummy11;
        gint dummy12;
        gint dummy13;
        gpointer dummy14;
    } GtkTextIter;
    typedef struct _GtkTextTagTable {
        GObject parent_instance;
        GtkTextTagTablePrivate *priv;
    } GtkTextTagTable;
    typedef struct _GtkTextTag {
        GObject parent_instance;
        GtkTextTagPrivate *priv;
    } GtkTextTag;
    typedef struct _GtkTextTagPrivate GtkTextTagPrivate;
    typedef struct _GtkTextTagClass {
        GObjectClass parent_class;
        gboolean(*event) (GtkTextTag * tag, GObject * event_object,
            GdkEvent * event, const GtkTextIter * iter);
        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkTextTagClass;
    typedef struct _GtkApplication {
        GApplication parent;
        GtkApplicationPrivate *priv;
    } GtkApplication;
    typedef struct _GtkApplicationClass {
        GApplicationClass parent_class;
        void (*window_added) (GtkApplication * application,
            GtkWidget * window);
        void (*window_removed) (GtkApplication * application,
            GtkWidget * window);
        gpointer padding;
    } GtkApplicationClass;
    typedef struct _GtkApplicationPrivate GtkApplicationPrivate;
    typedef enum {
        GTK_APPLICATION_INHIBIT_LOGOUT,
        GTK_APPLICATION_INHIBIT_SWITCH,
        GTK_APPLICATION_INHIBIT_SUSPEND,

```

```

    GTK_APPLICATION_INHIBIT_IDLE
} GtkApplicationInhibitFlags;
typedef struct _GtkAccessible {
    GObject parent;
    GtkAccessiblePrivate *priv;
} GtkAccessible;
typedef struct _GtkAccessiblePrivate GtkAccessiblePrivate;
typedef struct _GtkAccessibleClass {
    GObjectClass parent_class;
    void (*connect_widget_destroyed) (GtkAccessible * accessible);
    void (*widget_set) (GtkAccessible * accessible);
    void (*widget_unset) (GtkAccessible * accessible);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAccessibleClass;
typedef struct _GtkMessageDialog {
    GtkDialog parent_instance;
    GtkMessageDialogPrivate *priv;
} GtkMessageDialog;
typedef struct _GtkMessageDialogPrivate GtkMessageDialogPrivate;
typedef struct _GtkMessageDialogClass {
    GtkDialogClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMessageDialogClass;
typedef enum {
    GTK_BUTTONS_NONE,
    GTK_BUTTONS_OK,
    GTK_BUTTONS_CLOSE,
    GTK_BUTTONS_CANCEL,
    GTK_BUTTONS_YES_NO,
    GTK_BUTTONS_OK_CANCEL
} GtkButtonsType;
typedef struct _GtkRadioMenuItem {
    GtkCheckMenuItem check_menu_item;
    GtkRadioMenuItemPrivate *priv;
} GtkRadioMenuItem;
typedef struct _GtkRadioMenuItemPrivate GtkRadioMenuItemPrivate;
typedef struct _GtkRadioMenuItemClass {
    GtkCheckMenuItemClass parent_class;
    void (*group_changed) (GtkRadioMenuItem * radio_menu_item);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRadioMenuItemClass;
typedef struct _GtkStyleProperties {
    GObject parent_object;
    GtkStylePropertiesPrivate *priv;
} GtkStyleProperties;
typedef struct _GtkStylePropertiesClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkStylePropertiesClass;
typedef struct _GtkStylePropertiesPrivate
GtkStylePropertiesPrivate;
typedef struct _GtkSymbolicColor GtkSymbolicColor;
typedef struct _GtkGradient GtkGradient;
typedef gboolean(*GtkStylePropertyParser) (void);
typedef struct _GtkArrow {
    GtkMisc misc;

```

```

    GtkArrowPrivate *priv;
} GtkArrow;
typedef struct _GtkArrowPrivate GtkArrowPrivate;
typedef struct _GtkArrowClass {
    GtkMiscClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkArrowClass;
typedef struct _GtkVolumeButton {
    GtkScaleButton parent;
} GtkVolumeButton;
typedef struct _GtkVolumeButtonClass {
    GtkScaleButtonClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkVolumeButtonClass;
typedef struct _GtkStyleProviderIface {
    GTypeInterface g_iface;
    GtkStyleProperties *(*get_style) (GtkStyleProvider * provider,
                                      GtkWidgetPath * path);
    gboolean(*get_style_property) (GtkStyleProvider * provider,
                                   GtkWidgetPath * path,
                                   GtkStateFlags state,
                                   GParamSpec * pspec, GValue * value);
    GtkIconFactory *(*get_icon_factory) (GtkStyleProvider *
provider,
                                      GtkWidgetPath * path);
} GtkStyleProviderIface;
typedef struct _GtkStyleProvider GtkStyleProvider;
typedef struct _GtkApplicationWindowPrivate
GtkApplicationWindowPrivate;
typedef struct _GtkApplicationWindowClass {
    GtkWindowClass parent_class;
    gpointer padding[14];
} GtkApplicationWindowClass;
typedef struct _GtkApplicationWindow {
    GtkWindow parent_instance;
    GtkApplicationWindowPrivate *priv;
} GtkApplicationWindow;
typedef struct _GtkEntryCompletion {
    GObject parent_instance;
    GtkEntryCompletionPrivate *priv;
} GtkEntryCompletion;
typedef struct _GtkEntryCompletionClass {
    GObjectClass parent_class;
    gboolean(*match_selected) (GtkEntryCompletion * completion,
                              GtkTreeModel * model, GtkTreeIter *
iter);
    void (*action_activated) (GtkEntryCompletion * completion,
                              gint index_);
    gboolean(*insert_prefix) (GtkEntryCompletion * completion,
                              const gchar * prefix);
    gboolean(*cursor_on_match) (GtkEntryCompletion * completion,
                              GtkTreeModel * model, GtkTreeIter *
iter);
    void (*_gtk_reserved0) (void);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
} GtkEntryCompletionClass;
typedef struct _GtkEntryCompletionPrivate
GtkEntryCompletionPrivate;

```

```

typedef gboolean(*GtkEntryCompletionMatchFunc) (GtkEntryCompletion
*
completion,
const gchar * key,
GtkTreeIter * iter,
gpointer user_data);

typedef struct _GtkCellEditable GtkCellEditable;
typedef struct _GtkCellEditableIface {
    GTypeInterface g_iface;
    void (*editing_done) (GtkCellEditable * cell_editable);
    void (*remove_widget) (GtkCellEditable * cell_editable);
    void (*start_editing) (GtkCellEditable * cell_editable,
        GdkEvent * event);
} GtkCellEditableIface;
typedef struct _GtkCheckMenuItem {
    GtkMenuItem menu_item;
    GtkCheckMenuItemPrivate *priv;
} GtkCheckMenuItem;
typedef struct _GtkCheckMenuItemPrivate GtkCheckMenuItemPrivate;
typedef struct _GtkCheckMenuItemClass {
    GtkMenuItemClass parent_class;
    void (*toggled) (GtkCheckMenuItem * check_menu_item);
    void (*draw_indicator) (GtkCheckMenuItem * check_menu_item,
        cairo_t * cr);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCheckMenuItemClass;
typedef struct _GtkFontChooserWidget {
    GtkBox parent_instance;
    GtkFontChooserWidgetPrivate *priv;
} GtkFontChooserWidget;
typedef struct _GtkFontChooserWidgetPrivate
GtkFontChooserWidgetPrivate;
typedef struct _GtkFontChooserWidgetClass {
    GtkBoxClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkFontChooserWidgetClass;
typedef struct _GtkFontButton {
    GtkButton button;
    GtkFontButtonPrivate *priv;
} GtkFontButton;
typedef struct _GtkFontButtonClass {
    GtkButtonClass parent_class;
    void (*font_set) (GtkFontButton * gfp);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFontButtonClass;
typedef struct _GtkFontButtonPrivate GtkFontButtonPrivate;
typedef struct _GtkImageMenuItem {
    GtkMenuItem menu_item;
    GtkImageMenuItemPrivate *priv;
} GtkImageMenuItem;
typedef struct _GtkImageMenuItemPrivate GtkImageMenuItemPrivate;
typedef struct _GtkImageMenuItemClass {
    GtkMenuItemClass parent_class;

```

```

        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkImageMenuItemClass;
typedef struct _GtkAppChooser GtkAppChooser;
typedef struct _GtkDrawingArea {
    GtkWidget widget;
    gpointer dummy;
} GtkDrawingArea;
typedef struct _GtkDrawingAreaClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkDrawingAreaClass;
typedef struct _GtkFixed {
    GtkContainer container;
    GtkFixedPrivate *priv;
} GtkFixed;
typedef struct _GtkFixedPrivate GtkFixedPrivate;
typedef struct _GtkFixedClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkFixedClass;
typedef struct _GtkFixedChild {
    GtkWidget *widget;
    gint x;
    gint y;
} GtkFixedChild;
typedef struct _GtkBuildable GtkBuildable;
typedef struct _GtkBuildableInterface {
    GTypeInterface g_iface;
    void (*set_name) (GtkBuildable * buildable, const gchar * name);
    const gchar *(*get_name) (GtkBuildable * buildable);
    void (*add_child) (GtkBuildable * buildable, GtkBuilder *
builder,
                        GObject * child, const gchar * type);
    void (*set_buildable_property) (GtkBuildable * buildable,
                                    GtkBuilder * builder,
                                    const gchar * name,
                                    const GValue * value);
    GObject *(*construct_child) (GtkBuildable * buildable,
                                GtkBuilder * builder, const gchar *
name);
    gboolean(*custom_tag_start) (GtkBuildable * buildable,
                                GtkBuilder * builder, GObject * child,
                                const gchar * tagname,
                                GMarkupParser * parser, gpointer *
data);
    void (*custom_tag_end) (GtkBuildable * buildable, GtkBuilder *
builder,
                           GObject * child, const gchar * tagname,
                           gpointer * data);
    void (*custom_finished) (GtkBuildable * buildable,
                             GtkBuilder * builder, GObject * child,
                             const gchar * tagname, gpointer data);
    void (*parser_finished) (GtkBuildable * buildable,
                             GtkBuilder * builder);
    GObject *(*get_internal_child) (GtkBuildable * buildable,
                                    GtkBuilder * builder,
                                    const gchar * name);

```



```

} GtkBuildableIface;
typedef struct _GtkButton {
    GtkBin bin;
    GtkButtonPrivate *priv;
} GtkButton;
typedef struct _GtkButtonPrivate GtkButtonPrivate;
typedef struct _GtkButtonClass {
    GtkBinClass parent_class;
    void (*pressed) (GtkButton * button);
    void (*released) (GtkButton * button);
    void (*clicked) (GtkButton * button);
    void (*enter) (GtkButton * button);
    void (*leave) (GtkButton * button);
    void (*activate) (GtkButton * button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkButtonClass;
typedef enum {
    GTK_ALIGN_FILL = 0,
    GTK_ALIGN_START = 1,
    GTK_ALIGN_END = 2,
    GTK_ALIGN_CENTER = 3
} GtkAlign;
typedef enum {
    GTK_ARROWS_BOTH,
    GTK_ARROWS_START,
    GTK_ARROWS_END
} GtkArrowPlacement;
typedef enum {
    GTK_ARROW_UP = 0,
    GTK_ARROW_DOWN = 1,
    GTK_ARROW_LEFT = 2,
    GTK_ARROW_RIGHT = 3
} GtkArrowType;
typedef enum {
    GTK_EXPAND = 1 << 0,
    GTK_SHRINK = 1 << 1,
    GTK_FILL = 1 << 2
} GtkAttachOptions;
typedef enum {
    GTK_BUTTONBOX_DEFAULT_STYLE = 0,
    GTK_BUTTONBOX_SPREAD = 1,
    GTK_BUTTONBOX_EDGE = 2,
    GTK_BUTTONBOX_START = 3,
    GTK_BUTTONBOX_END = 4
} GtkButtonBoxStyle;
typedef enum {
    GTK_DELETE_CHARS,
    GTK_DELETE_WORD_ENDS,
    GTK_DELETE_WORDS,
    GTK_DELETE_DISPLAY_LINES,
    GTK_DELETE_DISPLAY_LINE_ENDS,
    GTK_DELETE_PARAGRAPH_ENDS,
    GTK_DELETE_PARAGRAPHS,
    GTK_DELETE_WHITESPACE
} GtkDeleteType;
typedef enum {
    GTK_DIR_TAB_FORWARD = 0,
    GTK_DIR_TAB_BACKWARD = 1,
    GTK_DIR_UP = 2,
    GTK_DIR_DOWN = 3,
    GTK_DIR_LEFT = 4,
    GTK_DIR_RIGHT = 5
} GtkDirectionType;

```

```

typedef enum {
    GTK_EXPANDER_COLLAPSED,
    GTK_EXPANDER_SEMI_COLLAPSED,
    GTK_EXPANDER_SEMI_EXPANDED,
    GTK_EXPANDER_EXPANDED
} GtkExpanderStyle;
typedef enum {
    GTK_ICON_SIZE_INVALID = 0,
    GTK_ICON_SIZE_MENU = 1,
    GTK_ICON_SIZE_SMALL_TOOLBAR = 2,
    GTK_ICON_SIZE_LARGE_TOOLBAR = 3,
    GTK_ICON_SIZE_BUTTON = 4,
    GTK_ICON_SIZE_DND = 5,
    GTK_ICON_SIZE_DIALOG = 6
} GtkIconSize;
typedef enum {
    GTK_SENSITIVITY_AUTO,
    GTK_SENSITIVITY_ON,
    GTK_SENSITIVITY_OFF
} GtkSensitivityType;
typedef enum {
    GTK_TEXT_DIR_NONE,
    GTK_TEXT_DIR_LTR,
    GTK_TEXT_DIR_RTL
} GtkTextDirection;
typedef enum {
    GTK_JUSTIFY_LEFT,
    GTK_JUSTIFY_RIGHT,
    GTK_JUSTIFY_CENTER,
    GTK_JUSTIFY_FILL
} GtkJustification;
typedef enum {
    GTK_MENU_DIR_PARENT,
    GTK_MENU_DIR_CHILD,
    GTK_MENU_DIR_NEXT,
    GTK_MENU_DIR_PREV
} GtkMenuDirectionType;
typedef enum {
    GTK_MESSAGE_INFO,
    GTK_MESSAGE_WARNING,
    GTK_MESSAGE_QUESTION,
    GTK_MESSAGE_ERROR,
    GTK_MESSAGE_OTHER
} GtkMessageType;
typedef enum {
    GTK_MOVEMENT_LOGICAL_POSITIONS,
    GTK_MOVEMENT_VISUAL_POSITIONS,
    GTK_MOVEMENT_WORDS,
    GTK_MOVEMENT_DISPLAY_LINES,
    GTK_MOVEMENT_DISPLAY_LINE_ENDS,
    GTK_MOVEMENT_PARAGRAPHS,
    GTK_MOVEMENT_PARAGRAPH_ENDS,
    GTK_MOVEMENT_PAGES,
    GTK_MOVEMENT_BUFFER_ENDS,
    GTK_MOVEMENT_HORIZONTAL_PAGES
} GtkMovementStep;
typedef enum {
    GTK_SCROLL_STEPS,
    GTK_SCROLL_PAGES,
    GTK_SCROLL_ENDS,
    GTK_SCROLL_HORIZONTAL_STEPS,
    GTK_SCROLL_HORIZONTAL_PAGES,
    GTK_SCROLL_HORIZONTAL_ENDS
} GtkScrollStep;
typedef enum {
    GTK_ORIENTATION_HORIZONTAL,

```

```

    GTK_ORIENTATION_VERTICAL
} GtkOrientation;
typedef enum {
    GTK_CORNER_TOP_LEFT,
    GTK_CORNER_BOTTOM_LEFT,
    GTK_CORNER_TOP_RIGHT,
    GTK_CORNER_BOTTOM_RIGHT
} GtkCornerType;
typedef enum {
    GTK_PACK_START,
    GTK_PACK_END
} GtkPackType;
typedef enum {
    GTK_PATH_PRIO_LOWEST = 0,
    GTK_PATH_PRIO_GTK = 4,
    GTK_PATH_PRIO_APPLICATION = 8,
    GTK_PATH_PRIO_THEME = 10,
    GTK_PATH_PRIO_RC = 12,
    GTK_PATH_PRIO_HIGHEST = 15
} GtkPathPriorityType;
typedef enum {
    GTK_PATH_WIDGET,
    GTK_PATH_WIDGET_CLASS,
    GTK_PATH_CLASS
} GtkPathType;
typedef enum {
    GTK_POLICY_ALWAYS,
    GTK_POLICY_AUTOMATIC,
    GTK_POLICY_NEVER
} GtkPolicyType;
typedef enum {
    GTK_POS_LEFT,
    GTK_POS_RIGHT,
    GTK_POS_TOP,
    GTK_POS_BOTTOM
} GtkPositionType;
typedef enum {
    GTK_RELIEF_NORMAL,
    GTK_RELIEF_HALF,
    GTK_RELIEF_NONE
} GtkReliefStyle;
typedef enum {
    GTK_RESIZE_PARENT,
    GTK_RESIZE_QUEUE,
    GTK_RESIZE_IMMEDIATE
} GtkResizeMode;
typedef enum {
    GTK_SCROLL_NONE,
    GTK_SCROLL_JUMP,
    GTK_SCROLL_STEP_BACKWARD,
    GTK_SCROLL_STEP_FORWARD,
    GTK_SCROLL_PAGE_BACKWARD,
    GTK_SCROLL_PAGE_FORWARD,
    GTK_SCROLL_STEP_UP,
    GTK_SCROLL_STEP_DOWN,
    GTK_SCROLL_PAGE_UP,
    GTK_SCROLL_PAGE_DOWN,
    GTK_SCROLL_STEP_LEFT,
    GTK_SCROLL_STEP_RIGHT,
    GTK_SCROLL_PAGE_LEFT,
    GTK_SCROLL_PAGE_RIGHT,
    GTK_SCROLL_START,
    GTK_SCROLL_END
} GtkScrollType;
typedef enum {
    GTK_SELECTION_NONE,

```

```

        GTK_SELECTION_SINGLE,
        GTK_SELECTION_BROWSE,
        GTK_SELECTION_MULTIPLE
    } GtkSelectionMode;
typedef enum {
    GTK_SHADOW_NONE,
    GTK_SHADOW_IN,
    GTK_SHADOW_OUT,
    GTK_SHADOW_ETCHED_IN,
    GTK_SHADOW_ETCHED_OUT
} GtkShadowType;
typedef enum {
    GTK_STATE_NORMAL,
    GTK_STATE_ACTIVE,
    GTK_STATE_PRELIGHT,
    GTK_STATE_SELECTED,
    GTK_STATE_INSENSITIVE,
    GTK_STATE_INCONSISTENT,
    GTK_STATE_FOCUSED
} GtkStateType;
typedef enum {
    GTK_TOOLBAR_ICONS,
    GTK_TOOLBAR_TEXT,
    GTK_TOOLBAR_BOTH,
    GTK_TOOLBAR_BOTH_HORIZ
} GtkToolbarStyle;
typedef enum {
    GTK_WIN_POS_NONE,
    GTK_WIN_POS_CENTER,
    GTK_WIN_POS_MOUSE,
    GTK_WIN_POS_CENTER_ALWAYS,
    GTK_WIN_POS_CENTER_ON_PARENT
} GtkWindowPosition;
typedef enum {
    GTK_WINDOW_TOPLEVEL,
    GTK_WINDOW_POPUP
} GtkWindowType;
typedef enum {
    GTK_WRAP_NONE,
    GTK_WRAP_CHAR,
    GTK_WRAP_WORD,
    GTK_WRAP_WORD_CHAR
} GtkWrapMode;
typedef enum {
    GTK_SORT_ASCENDING,
    GTK_SORT_DESCENDING
} GtkSortType;
typedef enum {
    GTK_IM_PREEDIT_NOTHING,
    GTK_IM_PREEDIT_CALLBACK,
    GTK_IM_PREEDIT_NONE
} GtkIMPreeditStyle;
typedef enum {
    GTK_IM_STATUS_NOTHING,
    GTK_IM_STATUS_CALLBACK,
    GTK_IM_STATUS_NONE
} GtkIMStatusStyle;
typedef enum {
    GTK_PACK_DIRECTION_LTR,
    GTK_PACK_DIRECTION_RTL,
    GTK_PACK_DIRECTION_TTB,
    GTK_PACK_DIRECTION_BTT
} GtkPackDirection;
typedef enum {
    GTK_PRINT_PAGES_ALL,
    GTK_PRINT_PAGES_CURRENT,

```

```

        GTK_PRINT_PAGES_RANGES,
        GTK_PRINT_PAGES_SELECTION
    } GtkPrintPages;
typedef enum {
    GTK_PAGE_SET_ALL,
    GTK_PAGE_SET_EVEN,
    GTK_PAGE_SET_ODD
} GtkPageSet;
typedef enum {
    GTK_NUMBER_UP_LAYOUT_LEFT_TO_RIGHT_TOP_TO_BOTTOM,
    GTK_NUMBER_UP_LAYOUT_LEFT_TO_RIGHT_BOTTOM_TO_TOP,
    GTK_NUMBER_UP_LAYOUT_RIGHT_TO_LEFT_TOP_TO_BOTTOM,
    GTK_NUMBER_UP_LAYOUT_RIGHT_TO_LEFT_BOTTOM_TO_TOP,
    GTK_NUMBER_UP_LAYOUT_TOP_TO_BOTTOM_LEFT_TO_RIGHT,
    GTK_NUMBER_UP_LAYOUT_TOP_TO_BOTTOM_RIGHT_TO_LEFT,
    GTK_NUMBER_UP_LAYOUT_BOTTOM_TO_TOP_LEFT_TO_RIGHT,
    GTK_NUMBER_UP_LAYOUT_BOTTOM_TO_TOP_RIGHT_TO_LEFT
} GtkNumberUpLayout;
typedef enum {
    GTK_PAGE_ORIENTATION_PORTRAIT,
    GTK_PAGE_ORIENTATION_LANDSCAPE,
    GTK_PAGE_ORIENTATION_REVERSE_PORTRAIT,
    GTK_PAGE_ORIENTATION_REVERSE_LANDSCAPE
} GtkPageOrientation;
typedef enum {
    GTK_PRINT_QUALITY_LOW,
    GTK_PRINT_QUALITY_NORMAL,
    GTK_PRINT_QUALITY_HIGH,
    GTK_PRINT_QUALITY_DRAFT
} GtkPrintQuality;
typedef enum {
    GTK_PRINT_DUPLEX_SIMPLEX,
    GTK_PRINT_DUPLEX_HORIZONTAL,
    GTK_PRINT_DUPLEX_VERTICAL
} GtkPrintDuplex;
typedef enum {
    GTK_UNIT_NONE,
    GTK_UNIT_POINTS,
    GTK_UNIT_INCH,
    GTK_UNIT_MM
} GtkUnit;
typedef enum {
    GTK_TREE_VIEW_GRID_LINES_NONE,
    GTK_TREE_VIEW_GRID_LINES_HORIZONTAL,
    GTK_TREE_VIEW_GRID_LINES_VERTICAL,
    GTK_TREE_VIEW_GRID_LINES_BOTH
} GtkTreeViewGridLines;
typedef enum {
    GTK_DRAG_RESULT_SUCCESS,
    GTK_DRAG_RESULT_NO_TARGET,
    GTK_DRAG_RESULT_USER_CANCELLED,
    GTK_DRAG_RESULT_TIMEOUT_EXPIRED,
    GTK_DRAG_RESULT_GRAB_BROKEN,
    GTK_DRAG_RESULT_ERROR
} GtkDragResult;
typedef enum {
    GTK_SIZE_REQUEST_HEIGHT_FOR_WIDTH,
    GTK_SIZE_REQUEST_WIDTH_FOR_HEIGHT,
    GTK_SIZE_REQUEST_CONSTANT_SIZE
} GtkSizeRequestMode;
typedef enum {
    GTK_SCROLL_MINIMUM,
    GTK_SCROLL_NATURAL
} GtkScrollablePolicy;
typedef enum {
    GTK_STATE_FLAG_NORMAL = 0,

```

```

        GTK_STATE_FLAG_ACTIVE = 1 << 0,
        GTK_STATE_FLAG_PRELIGHT = 1 << 1,
        GTK_STATE_FLAG_SELECTED = 1 << 2,
        GTK_STATE_FLAG_INSENSITIVE = 1 << 3,
        GTK_STATE_FLAG_INCONSISTENT = 1 << 4,
        GTK_STATE_FLAG_FOCUSED = 1 << 5,
        GTK_STATE_FLAG_BACKDROP = 1 << 6
    } GtkStateFlags;
typedef enum {
    GTK_REGION_EVEN = 1 << 0,
    GTK_REGION_ODD = 1 << 1,
    GTK_REGION_FIRST = 1 << 2,
    GTK_REGION_LAST = 1 << 3,
    GTK_REGION_ONLY = 1 << 4,
    GTK_REGION_SORTED = 1 << 5
} GtkRegionFlags;
typedef enum {
    GTK_JUNCTION_NONE = 0,
    GTK_JUNCTION_CORNER_TOPLEFT = 1 << 0,
    GTK_JUNCTION_CORNER_TOPRIGHT = 1 << 1,
    GTK_JUNCTION_CORNER_BOTTOMLEFT = 1 << 2,
    GTK_JUNCTION_CORNER_BOTTOMRIGHT = 1 << 3,
    GTK_JUNCTION_TOP =
        (GTK_JUNCTION_CORNER_TOPLEFT
GTK_JUNCTION_CORNER_TOPRIGHT),
    GTK_JUNCTION_BOTTOM =
        (GTK_JUNCTION_CORNER_BOTTOMLEFT
GTK_JUNCTION_CORNER_BOTTOMRIGHT),
    GTK_JUNCTION_LEFT =
        (GTK_JUNCTION_CORNER_TOPLEFT
GTK_JUNCTION_CORNER_BOTTOMLEFT),
    GTK_JUNCTION_RIGHT =
        (GTK_JUNCTION_CORNER_TOPRIGHT
GTK_JUNCTION_CORNER_BOTTOMRIGHT)
} GtkJunctionSides;
typedef enum {
    GTK_BORDER_STYLE_NONE,
    GTK_BORDER_STYLE_SOLID,
    GTK_BORDER_STYLE_INSET,
    GTK_BORDER_STYLE_OUTSET,
    GTK_BORDER_STYLE_HIDDEN,
    GTK_BORDER_STYLE_DOTTED,
    GTK_BORDER_STYLE_DASHED,
    GTK_BORDER_STYLE_DOUBLE,
    GTK_BORDER_STYLE_GROOVE,
    GTK_BORDER_STYLE RIDGE
} GtkBorderStyle;
typedef enum {
    GTK_LEVEL_BAR_MODE_CONTINUOUS,
    GTK_LEVEL_BAR_MODE_DISCRETE
} GtkLevelBarMode;
typedef enum {
    GTK_INPUT_PURPOSE_FREE_FORM,
    GTK_INPUT_PURPOSE_ALPHA,
    GTK_INPUT_PURPOSE_DIGITS,
    GTK_INPUT_PURPOSE_NUMBER,
    GTK_INPUT_PURPOSE_PHONE,
    GTK_INPUT_PURPOSE_URL,
    GTK_INPUT_PURPOSE_EMAIL,
    GTK_INPUT_PURPOSE_NAME,
    GTK_INPUT_PURPOSE_PASSWORD,
    GTK_INPUT_PURPOSE_PIN
} GtkInputPurpose;
typedef enum {
    GTK_INPUT_HINT_NONE = 0,
    GTK_INPUT_HINT_SPELLCHECK = 1 << 0,

```

```

GTK_INPUT_HINT_NO_SPELLCHECK = 1 << 1,
GTK_INPUT_HINT_WORD_COMPLETION = 1 << 2,
GTK_INPUT_HINT_LOWERCASE = 1 << 3,
GTK_INPUT_HINT_UPPERCASE_CHARS = 1 << 4,
GTK_INPUT_HINT_UPPERCASE_WORDS = 1 << 4,
GTK_INPUT_HINT_UPPERCASE_SENTENCES = 1 << 6,
GTK_INPUT_HINT_INHIBIT_OSK = 1 << 7
} GtkInputHints;
typedef enum {
    GTK_ACCEL_VISIBLE,
    GTK_ACCEL_LOCKED,
    GTK_ACCEL_MASK
} GtkAccelFlags;
typedef struct _GtkAccelGroup {
    GObject parent;
    GtkAccelGroupPrivate *priv;
} GtkAccelGroup;
typedef struct _GtkAccelGroupClass {
    GObjectClass parent_class;
    void (*accel_changed) (GtkAccelGroup * accel_group, guint
keyval,
                                GdkModifierType modifier,
                                GClosure * accel_closure);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAccelGroupClass;
typedef struct _GtkAccelGroupPrivate GtkAccelGroupPrivate;
typedef struct _GtkAccelKey {
    guint accel_key;
    GdkModifierType accel_mods;
    guint accel_flags:16;
} GtkAccelKey;
typedef struct _GtkAccelGroupEntry {
    GtkAccelKey key;
    GClosure *closure;
    GQuark accel_path_quark;
} GtkAccelGroupEntry;
typedef gboolean(*GtkAccelGroupFindFunc) (GtkAccelKey * key,
                                GClosure * closure,
                                gpointer data);

typedef struct _GtkIconFactory {
    GObject parent_instance;
    GtkIconFactoryPrivate *priv;
} GtkIconFactory;
typedef struct _GtkIconFactoryPrivate GtkIconFactoryPrivate;
typedef struct _GtkIconFactoryClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkIconFactoryClass;
typedef struct _GtkAppChooserWidget {
    GtkBox parent;
    GtkAppChooserWidgetPrivate *priv;
} GtkAppChooserWidget;
typedef struct _GtkAppChooserWidgetClass {
    GtkBoxClass parent_class;
    void (*application_selected) (GtkAppChooserWidget * self,
                                GAppInfo * app_info);
    void (*application_activated) (GtkAppChooserWidget * self,
                                GAppInfo * app_info);
    void (*populate_popup) (GtkAppChooserWidget * self, GtkMenu *
menu,

```

```

        GAppInfo * app_info);

    gpointer padding;
} GtkAppChooserWidgetClass;
typedef struct _GtkAppChooserWidgetPrivate
GtkAppChooserWidgetPrivate;
typedef struct _GtkButtonBox {
    GtkWidget box;
    GtkButtonBoxPrivate *priv;
} GtkButtonBox;
typedef struct _GtkButtonBoxPrivate GtkButtonBoxPrivate;
typedef struct _GtkButtonBoxClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkButtonBoxClass;
typedef struct _GtkCellArea {
    GInitiallyUnowned parent_instance;
    GtkCellAreaPrivate *priv;
} GtkCellArea;
typedef struct _GtkCellAreaClass {
    GInitiallyUnownedClass parent_class;
    void (*add) (GtkCellArea * area, GtkCellRenderer * renderer);
    void (*remove) (GtkCellArea * area, GtkCellRenderer * renderer);
    void (*foreach) (GtkCellArea * area, GtkCellCallback callback,
        gpointer callback_data);
    void (*foreach_alloc) (GtkCellArea * area,
        GtkCellAreaContext * context,
        GtkWidget * widget,
        const GdkRectangle * cell_area,
        const GdkRectangle * background_area,
        GtkCellAllocCallback callback,
        gpointer callback_data);
    gint(*event) (GtkCellArea * area, GtkCellAreaContext * context,
        GtkWidget * widget, GdkEvent * event,
        const GdkRectangle * cell_area,
        GtkCellRendererState flags);
    void (*render) (GtkCellArea * area, GtkCellAreaContext * context,
        GtkWidget * widget, cairo_t * cr,
        const GdkRectangle * background_area,
        const GdkRectangle * cell_area,
        GtkCellRendererState flags, gboolean paint_focus);
    void (*apply_attributes) (GtkCellArea * area,
        GtkTreeModel * tree_model,
        GtkTreeIter * iter, gboolean is_expander,
        gboolean is_expanded);
    GtkCellAreaContext *(*create_context) (GtkCellArea * area);
    GtkCellAreaContext *(*copy_context) (GtkCellArea * area,
        GtkCellAreaContext * context);
    GtkSizeRequestMode(*get_request_mode) (GtkCellArea * area);
    void (*get_preferred_width) (GtkCellArea * area,
        GtkCellAreaContext * context,
        GtkWidget * widget, gint *
        minimum_width,
        gint * natural_width);
    void (*get_preferred_height_for_width) (GtkCellArea * area,
        GtkCellAreaContext * context,
        GtkWidget * widget, gint width,
        gint * minimum_height,
        gint * natural_height);
    void (*get_preferred_height) (GtkCellArea * area,
        GtkCellAreaContext * context,
        GtkWidget * widget, gint *
        minimum_width,
        gint * natural_width);

```



```

void (*get_preferred_width_for_height) (GtkCellArea * area,
                                         GtkCellAreaContext * context,
                                         GtkWidget * widget, gint width,
                                         gint * minimum_height,
                                         gint * natural_height);

void (*set_cell_property) (GtkCellArea * area,
                           GtkCellRenderer * renderer,
                           guint property_id, const GValue * value,
                           GParamSpec * pspec);

void (*get_cell_property) (GtkCellArea * area,
                           GtkCellRenderer * renderer,
                           guint property_id, GValue * value,
                           GParamSpec * pspec);

gboolean(*focus) (GtkCellArea * area, GtkDirectionType
direction);
gboolean(*is_activatable) (GtkCellArea * area);
gboolean(*activate) (GtkCellArea * area, GtkCellAreaContext *
context,
                    GtkWidget * widget,
                    const GdkRectangle * cell_area,
                    GtkCellRendererState flags, gboolean
edit_only);
void (*_gtk_reserved1) (void);
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
void (*_gtk_reserved5) (void);
void (*_gtk_reserved6) (void);
void (*_gtk_reserved7) (void);
void (*_gtk_reserved8) (void);
} GtkCellAreaClass;
typedef struct _GtkCellAreaPrivate GtkCellAreaPrivate;
typedef struct _GtkCellAreaContext {
    GObject parent_instance;
    GtkCellAreaContextPrivate *priv;
} GtkCellAreaContext;
typedef gboolean(*GtkCellCallback) (void);
typedef gboolean(*GtkCellAllocCallback) (GtkCellRenderer *
renderer,
                                         const GdkRectangle * cell_area,
                                         const GdkRectangle *
                                         cell_background, gpointer data);
typedef struct _GtkCellAreaContextPrivate
GtkCellAreaContextPrivate;
typedef struct _GtkCellAreaContextClass {
    GObjectClass parent_class;
    void (*allocate) (GtkCellAreaContext * context, gint width,
                     gint height);
    void (*reset) (GtkCellAreaContext * context);
    void (*get_preferred_height_for_width) (GtkCellAreaContext *
context,
                                         gint width,
                                         gint * minimum_height,
                                         gint * natural_height);
    void (*get_preferred_width_for_height) (GtkCellAreaContext *
context,
                                         gint width,
                                         gint * minimum_height,
                                         gint * natural_height);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
} GtkCellAreaContextClass;

```

```

typedef struct _GtkPrintOperationClass {
    GObjectClass parent_class;
    void (*done) (GtkPrintOperation * operation,
                  GtkPrintOperationResult result);
    void (*begin_print) (GtkPrintOperation * operation,
                         GtkPrintContext * context);
    gboolean(*paginate) (GtkPrintOperation * operation,
                          GtkPrintContext * context);
    void (*request_page_setup) (GtkPrintOperation * operation,
                                GtkPrintContext * context, gint page_nr,
                                GtkPageSetup * setup);
    void (*draw_page) (GtkPrintOperation * operation,
                       GtkPrintContext * context, gint page_nr);
    void (*end_print) (GtkPrintOperation * operation,
                       GtkPrintContext * context);
    void (*status_changed) (GtkPrintOperation * operation);
    GtkWidget *(*create_custom_widget) (GtkPrintOperation *
operation);
    void (*custom_widget_apply) (GtkPrintOperation * operation,
                                 GtkWidget * widget);
    gboolean(*preview) (GtkPrintOperation * operation,
                        GtkPrintOperationPreview * preview,
                        GtkPrintContext * context, GtkWindow * parent);
    void (*update_custom_widget) (GtkPrintOperation * operation,
                                  GtkWidget * widget, GtkPageSetup *
setup,
                                  GtkPrintSettings * settings);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkPrintOperationClass;
typedef struct _GtkPrintOperationPrivate GtkPrintOperationPrivate;
typedef struct _GtkPrintOperation {
    GObject parent_instance;
    GtkPrintOperationPrivate *priv;
} GtkPrintOperation;
typedef enum {
    GTK_PRINT_STATUS_INITIAL,
    GTK_PRINT_STATUS_PREPARING,
    GTK_PRINT_STATUS_GENERATING_DATA,
    GTK_PRINT_STATUS_SENDING_DATA,
    GTK_PRINT_STATUS_PENDING,
    GTK_PRINT_STATUS_PENDING_ISSUE,
    GTK_PRINT_STATUS_PRINTING,
    GTK_PRINT_STATUS_FINISHED,
    GTK_PRINT_STATUS_FINISHED_ABORTED
} GtkPrintStatus;
typedef enum {
    GTK_PRINT_OPERATION_RESULT_ERROR,
    GTK_PRINT_OPERATION_RESULT_APPLY,
    GTK_PRINT_OPERATION_RESULT_CANCEL,
    GTK_PRINT_OPERATION_RESULT_IN_PROGRESS
} GtkPrintOperationResult;
typedef enum {
    GTK_PRINT_OPERATION_ACTION_PRINT_DIALOG,
    GTK_PRINT_OPERATION_ACTION_PRINT,
    GTK_PRINT_OPERATION_ACTION_PREVIEW,
    GTK_PRINT_OPERATION_ACTION_EXPORT
} GtkPrintOperationAction;
typedef enum {
    GTK_PRINT_ERROR_GENERAL,

```

```

GTK_PRINT_ERROR_INTERNAL_ERROR,
GTK_PRINT_ERROR_NOMEM,
GTK_PRINT_ERROR_INVALID_FILE
} GtkPrintError;
typedef void (*GtkPageSetupDoneFunc) (GtkPageSetup * page_setup,
                                       gpointer data);
typedef guint8 (*GtkTextBufferSerializeFunc) (GtkTextBuffer *
                                              register_buffer,
                                              GtkTextBuffer *
                                              content_buffer,
                                              GtkTextIter * start,
                                              GtkTextIter * end,
                                              gsize * length,
                                              gpointer user_data);
typedef gboolean(*GtkTextBufferDeserializeFunc) (GtkTextBuffer *
                                                  register_buffer,
                                                  GtkTextBuffer *
                                                  content_buffer,
                                                  GtkTextIter * iter,
                                                  const guint8 * data,
                                                  gsize length,
                                                  gboolean create_tags,
                                                  gpointer user_data,
                                                  GError * *error);

typedef struct _GtkCellRendererPixbuf {
    GtkCellRenderer parent;
    GtkCellRendererPixbufPrivate *priv;
} GtkCellRendererPixbuf;
typedef struct _GtkCellRendererPixbufPrivate
GtkCellRendererPixbufPrivate;
typedef struct _GtkCellRendererPixbufClass {
    GtkCellRendererClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererPixbufClass;
typedef struct _GtkAlignment {
    GtkBin parent_instance;
    GtkAlignmentPrivate *priv;
} GtkAlignment;
typedef struct _GtkAlignmentPrivate GtkAlignmentPrivate;
typedef struct _GtkAlignmentClass {
    GtkBinClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAlignmentClass;
typedef enum {
    GTK_TREE_VIEW_DROP_BEFORE,
    GTK_TREE_VIEW_DROP_AFTER,
    GTK_TREE_VIEW_DROP_INTRO_OR_BEFORE,
    GTK_TREE_VIEW_DROP_INTRO_OR_AFTER
} GtkTreeViewDropPosition;
typedef struct _GtkTreeView {
    GtkContainer parent;
    GtkTreeViewPrivate *priv;
} GtkTreeView;
typedef struct _GtkTreeViewClass {
    GtkContainerClass parent_class;
    void (*row_activated) (GtkTreeView * tree_view, GtkTreePath *
path,
                        GtkTreeViewColumn * column);
    gboolean(*test_expand_row) (GtkTreeView * tree_view,
                                GtkTreeIter * iter, GtkTreePath * path);

```

```

        gboolean(*test_collapse_row) (GtkTreeView * tree_view,
                                       GtkTreeIter * iter, GtkTreePath *
path);
        void (*row_expanded) (GtkTreeView * tree_view, GtkTreeIter *
iter,
                             GtkTreePath * path);
        void (*row_collapsed) (GtkTreeView * tree_view, GtkTreeIter *
iter,
                              GtkTreePath * path);
        void (*columns_changed) (GtkTreeView * tree_view);
        void (*cursor_changed) (GtkTreeView * tree_view);
        gboolean(*move_cursor) (GtkTreeView * tree_view,
GtkMovementStep step,
                              gint count);
        gboolean(*select_all) (GtkTreeView * tree_view);
        gboolean(*unselect_all) (GtkTreeView * tree_view);
        gboolean(*select_cursor_row) (GtkTreeView * tree_view,
                                     gboolean start_editing);
        gboolean(*toggle_cursor_row) (GtkTreeView * tree_view);
        gboolean(*expand_collapse_cursor_row) (GtkTreeView * tree_view,
                                               gboolean logical,
                                               gboolean expand,
                                               gboolean open_all);
        gboolean(*select_cursor_parent) (GtkTreeView * tree_view);
        gboolean(*start_interactive_search) (GtkTreeView * tree_view);
        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
        void (*_gtk_reserved5) (void);
        void (*_gtk_reserved6) (void);
        void (*_gtk_reserved7) (void);
        void (*_gtk_reserved8) (void);
    } GtkTreeViewClass;
    typedef struct _GtkTreeViewPrivate GtkTreeViewPrivate;
    typedef struct _GtkTreeSelection {
        GObject parent;
        GtkTreeSelectionPrivate *priv;
    } GtkTreeSelection;
    typedef struct _GtkTreeSelectionClass {
        GObjectClass parent_class;
        void (*changed) (GtkTreeSelection * selection);
        void (*_gtk_reserved1) (void);
        void (*_gtk_reserved2) (void);
        void (*_gtk_reserved3) (void);
        void (*_gtk_reserved4) (void);
    } GtkTreeSelectionClass;
    typedef gboolean(*GtkTreeViewColumnDropFunc) (GtkTreeView *
tree_view,
                                                  GtkTreeViewColumn * column,
                                                  GtkTreeViewColumn *
prev_column,
                                                  GtkTreeViewColumn *
next_column, gpointer data);
    typedef void (*GtkTreeViewMappingFunc) (GtkTreeView * tree_view,
                                           GtkTreePath * path,
                                           gpointer user_data);
    typedef gboolean(*GtkTreeViewSearchEqualFunc) (GtkTreeModel *
model,
                                                  gint column,
                                                  const gchar * key,
                                                  GtkTreeIter * iter,
                                                  gpointer search_data);
    typedef gboolean(*GtkTreeViewRowSeparatorFunc) (GtkTreeModel *
model,
                                                  GtkTreeIter * iter,

```

```

                                gpointer data);
typedef void (*GtkTreeViewSearchPositionFunc) (GtkTreeView *
tree_view,
                                GtkWidget * search_dialog,
                                gpointer user_data);
typedef void (*GtkTreeDestroyCountFunc) (GtkTreeView * tree_view,
                                GtkTreePath * path, gint
children,
                                gpointer user_data);
typedef struct _GtkPaperSize GtkPaperSize;
typedef struct _GtkOrientable GtkOrientable;
typedef struct _GtkOrientableIface {
    GTypeInterface base_iface;
} GtkOrientableIface;
typedef struct _GtkCellRendererToggle {
    GtkCellRenderer parent;
    GtkCellRendererTogglePrivate *priv;
} GtkCellRendererToggle;
typedef struct _GtkCellRendererTogglePrivate
GtkCellRendererTogglePrivate;
typedef struct _GtkCellRendererToggleClass {
    GtkCellRendererClass parent_class;
    void (*toggled) (GtkCellRendererToggle * cell_renderer_toggle,
                    const gchar * path);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererToggleClass;
typedef struct _GtkAboutDialog {
    GtkDialog parent_instance;
    GtkAboutDialogPrivate *priv;
} GtkAboutDialog;
typedef struct _GtkAboutDialogClass {
    GtkDialogClass parent_class;
    gboolean(*activate_link) (GtkAboutDialog * dialog, const gchar
* uri);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAboutDialogClass;
typedef struct _GtkAboutDialogPrivate GtkAboutDialogPrivate;
typedef enum {
    GTK_LICENSE_UNKNOWN,
    GTK_LICENSE_CUSTOM,
    GTK_LICENSE_GPL_2_0,
    GTK_LICENSE_GPL_3_0,
    GTK_LICENSE_LGPL_2_1,
    GTK_LICENSE_LGPL_3_0,
    GTK_LICENSE_BSD,
    GTK_LICENSE_MIT_X11,
    GTK_LICENSE_ARTISTIC
} GtkLicense;
typedef struct _GtkTreeIter {
    gint stamp;
    gpointer user_data;
    gpointer user_data2;
    gpointer user_data3;
} GtkTreeIter;
typedef struct _GtkTreePath GtkTreePath;
typedef struct _GtkTreeRowReference GtkTreeRowReference;
typedef struct _GtkTreeModel GtkTreeModel;
typedef struct _GtkTreeModelIface {
    GTypeInterface g_iface;

```

```

    void (*row_changed) (GtkTreeModel * tree_model, GtkTreePath *
    path,
                        GtkTreeIter * iter);
    void (*row_inserted) (GtkTreeModel * tree_model, GtkTreePath *
    path,
                        GtkTreeIter * iter);
    void (*row_has_child_toggled) (GtkTreeModel * tree_model,
                                GtkTreePath * path, GtkTreeIter *
    iter);
    void (*row_deleted) (GtkTreeModel * tree_model, GtkTreePath *
    path);
    void (*rows_reordered) (GtkTreeModel * tree_model, GtkTreePath
    * path,
                        GtkTreeIter * iter, gint * new_order);
    GtkTreeModelFlags (*get_flags) (GtkTreeModel * tree_model);
    gint (*get_n_columns) (GtkTreeModel * tree_model);
    GType (*get_column_type) (GtkTreeModel * tree_model, gint
    index_);
    gboolean (*get_iter) (GtkTreeModel * tree_model, GtkTreeIter *
    iter,
                        GtkTreePath * path);
    GtkTreePath * (*get_path) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter);
    void (*get_value) (GtkTreeModel * tree_model, GtkTreeIter * iter,
                        gint column, GValue * value);
    gboolean (*iter_next) (GtkTreeModel * tree_model, GtkTreeIter *
    iter);
    gboolean (*iter_previous) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter);
    gboolean (*iter_children) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter, GtkTreeIter * parent);
    gboolean (*iter_has_child) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter);
    gint (*iter_n_children) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter);
    gboolean (*iter_nth_child) (GtkTreeModel * tree_model,
                                GtkTreeIter * iter, GtkTreeIter * parent,
                                gint n);
    gboolean (*iter_parent) (GtkTreeModel * tree_model, GtkTreeIter
    * iter,
                        GtkTreeIter * parent);
    void (*ref_node) (GtkTreeModel * tree_model, GtkTreeIter * iter);
    void (*unref_node) (GtkTreeModel * tree_model, GtkTreeIter *
    iter);
} GtkTreeModelIface;
typedef gboolean (*GtkTreeModelForeachFunc) (GtkTreeModel * model,
                                GtkTreePath * path,
                                GtkTreeIter * iter,
                                gpointer data);
typedef enum {
    GTK_TREE_MODEL_ITERS_PERSIST,
    GTK_TREE_MODEL_LIST_ONLY
} GtkTreeModelFlags;
typedef struct _GtkCheckButton {
    GtkToggleButton toggle_button;
} GtkCheckButton;
typedef struct _GtkCheckButtonClass {
    GtkToggleButtonClass parent_class;
    void (*draw_indicator) (GtkCheckButton * check_button, cairo_t
    * cr);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCheckButtonClass;
typedef struct _GtkWindowPrivate GtkWindowPrivate;

```

```

typedef struct _GtkWindowClass {
    GtkBinClass parent_class;
    void (*set_focus) (GtkWindow * window, GtkWidget * focus);
    void (*activate_focus) (GtkWindow * window);
    void (*activate_default) (GtkWindow * window);
    void (*keys_changed) (GtkWindow * window);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkWindowClass;
typedef struct _GtkWindowGeometryInfo GtkWindowGeometryInfo;
typedef struct _GtkWindowGroup {
    GObject parent_instance;
    GtkWindowPrivate *priv;
} GtkWindowGroup;
typedef struct _GtkWindowGroupClass {
    GObjectClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkWindowGroupClass;
typedef struct _GtkWindowGroupPrivate GtkWindowGroupPrivate;
typedef struct _GtkMenu {
    GtkMenuShell menu_shell;
    GtkMenuPrivate *priv;
} GtkMenu;
typedef struct _GtkMenuClass {
    GtkMenuShellClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkMenuClass;
typedef struct _GtkMenuPrivate GtkMenuPrivate;
typedef void (*GtkMenuPositionFunc) (GtkMenu * menu, gint * x, gint
* y,
                                gboolean * push_in,
                                gpointer user_data);
typedef void (*GtkMenuDetachFunc) (GtkWidget * attach_widget,
                                GtkMenu * menu);
typedef struct _GtkLevelBarClass {
    GtkWidgetClass parent_class;
    void (*offset_changed) (GtkLevelBar * self, const gchar * name);
    gpointer padding;
} GtkLevelBarClass;
typedef struct _GtkLevelBar {
    GtkWidget parent;
    GtkLevelBarPrivate *priv;
} GtkLevelBar;
typedef struct _GtkLevelBarPrivate GtkLevelBarPrivate;
typedef struct _GtkComboBox {
    GtkBin parent_instance;
    GtkComboBoxPrivate *priv;
} GtkComboBox;
typedef struct _GtkComboBoxClass {
    GtkBinClass parent_class;
    void (*changed) (GtkComboBox * combo_box);
    gchar *(*format_entry_text) (GtkComboBox * combo_box,
                                const gchar * path);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
} GtkComboBoxClass;
typedef struct _GtkComboBoxPrivate GtkComboBoxPrivate;

```

```

typedef struct _GtkSpinner {
    GtkWidget parent;
    GtkSpinnerPrivate *priv;
} GtkSpinner;
typedef struct _GtkSpinnerClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSpinnerClass;
typedef struct _GtkSpinnerPrivate GtkSpinnerPrivate;
typedef struct _GtkToolPalette {
    GtkContainer parent_instance;
    GtkToolPalettePrivate *priv;
} GtkToolPalette;
typedef struct _GtkToolPaletteClass {
    GtkContainerClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkToolPaletteClass;
typedef struct _GtkToolPalettePrivate GtkToolPalettePrivate;
typedef enum {
    GTK_TOOL_PALETTE_DRAG_ITEMS = (1 << 0),
    GTK_TOOL_PALETTE_DRAG_GROUPS = (1 << 1)
} GtkToolPaletteDragTargets;
typedef struct _GtkRecentChooserDialog {
    GtkDialog parent_instance;
    GtkRecentChooserDialogPrivate *priv;
} GtkRecentChooserDialog;
typedef struct _GtkRecentChooserDialogClass {
    GtkDialogClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRecentChooserDialogClass;
typedef struct _GtkRecentChooserDialogPrivate
    GtkRecentChooserDialogPrivate;
typedef struct _GtkCellRendererCombo {
    GtkCellRendererText parent_instance;
    GtkCellRendererComboPrivate *priv;
} GtkCellRendererCombo;
typedef struct _GtkCellRendererComboPrivate
    _GtkCellRendererComboPrivate;
typedef struct _GtkCellRendererComboClass {
    GtkCellRendererTextClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkCellRendererComboClass;
typedef struct _GtkBuilder {
    GObject parent_instance;
    GtkBuilderPrivate *priv;
} GtkBuilder;
typedef struct _GtkBuilderClass {
    GObjectClass parent_class;
    GType(*get_type_from_name) (GtkBuilder * builder,
                                const char *type_name);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
}

```



```

void (*_gtk_reserved5) (void);
void (*_gtk_reserved6) (void);
void (*_gtk_reserved7) (void);
void (*_gtk_reserved8) (void);
} GtkBuilderClass;
typedef struct _GtkBuilderPrivate GtkBuilderPrivate;
typedef void (*GtkBuilderConnectFunc) (GtkBuilder * builder,
                                       GObject * object,
                                       const gchar * signal_name,
                                       const gchar * handler_name,
                                       GObject * connect_object,
                                       GConnectFlags flags,
                                       gpointer user_data);

typedef struct _GtkTextBTree GtkTextBTree;
typedef struct _GtkTextBufferPrivate GtkTextBufferPrivate;
typedef struct _GtkTextBufferClass {
    GObjectClass parent_class;
    void (*insert_text) (GtkTextBuffer * buffer, GtkTextIter * pos,
                        const gchar * new_text, gint new_text_length);
    void (*insert_pixbuf) (GtkTextBuffer * buffer, GtkTextIter *
iter,
                        GdkPixbuf * pixbuf);
    void (*insert_child_anchor) (GtkTextBuffer * buffer,
                                GtkTextIter * iter,
                                GtkTextChildAnchor * anchor);
    void (*delete_range) (GtkTextBuffer * buffer, GtkTextIter *
start,
                        GtkTextIter * end);
    void (*changed) (GtkTextBuffer * buffer);
    void (*modified_changed) (GtkTextBuffer * buffer);
    void (*mark_set) (GtkTextBuffer * buffer, const GtkTextIter *
location,
                        GtkTextMark * mark);
    void (*mark_deleted) (GtkTextBuffer * buffer, GtkTextMark *
mark);
    void (*apply_tag) (GtkTextBuffer * buffer, GtkTextTag * tag,
                        const GtkTextIter * start, const GtkTextIter *
end);
    void (*remove_tag) (GtkTextBuffer * buffer, GtkTextTag * tag,
                        const GtkTextIter * start,
                        const GtkTextIter * end);
    void (*begin_user_action) (GtkTextBuffer * buffer);
    void (*end_user_action) (GtkTextBuffer * buffer);
    void (*paste_done) (GtkTextBuffer * buffer, GtkClipboard *
clipboard);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkTextBufferClass;
typedef struct _GtkIconInfo GtkIconInfo;
typedef struct _GtkIconTheme {
    GObject parent_instance;
    GtkIconThemePrivate *priv;
} GtkIconTheme;
typedef struct _GtkIconThemeClass {
    GObjectClass parent_class;
    void (*changed) (GtkIconTheme * icon_theme);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkIconThemeClass;
typedef struct _GtkIconThemePrivate GtkIconThemePrivate;
typedef enum {
    GTK_ICON_LOOKUP_NO_SVG = 1 << 0,

```

```

    GTK_ICON_LOOKUP_FORCE_SVG = 1 << 1,
    GTK_ICON_LOOKUP_USE_BUILTIN = 1 << 2,
    GTK_ICON_LOOKUP_GENERIC_FALLBACK = 1 << 3,
    GTK_ICON_LOOKUP_FORCE_SIZE = 1 << 4
} GtkIconLookupFlags;
typedef enum {
    GTK_ICON_THEME_NOT_FOUND,
    GTK_ICON_THEME_FAILED
} GtkIconThemeError;
typedef struct _GtkBindingSet {
    gchar *set_name;
    gint priority;
    GSList *widget_path_pspecs;
    GSList *widget_class_pspecs;
    GSList *class_branch_pspecs;
    GtkBindingEntry *entries;
    GtkBindingEntry *current;
    guint parsed:1;
} GtkBindingSet;
typedef struct _GtkBindingEntry {
    guint keyval;
    GdkModifierType modifiers;
    GtkBindingSet *binding_set;
    guint destroyed:1;
    guint in_emission:1;
    guint marks_unbound:1;
    GtkBindingEntry *set_next;
    GtkBindingEntry *hash_next;
    GtkBindingSignal *signals;
} GtkBindingEntry;
typedef struct _GtkBindingSignal {
    GtkBindingSignal *next;
    gchar *signal_name;
    guint n_args;
    GtkBindingArg *args;
} GtkBindingSignal;
typedef struct _GtkBindingArg {
    GType arg_type;
    union {
        glong long_data;
        gdouble double_data;
        gchar *string_data;
    } d;
} GtkBindingArg;
typedef struct _GtkCellLayout GtkCellLayout;
typedef struct _GtkCellLayoutIface {
    GTypeInterface g_iface;
    void (*pack_start) (GtkCellLayout * cell_layout,
                        GtkCellRenderer * cell, gboolean expand);
    void (*pack_end) (GtkCellLayout * cell_layout, GtkCellRenderer
* cell,
                        gboolean expand);
    void (*clear) (GtkCellLayout * cell_layout);
    void (*add_attribute) (GtkCellLayout * cell_layout,
                        GtkCellRenderer * cell, const gchar *
attribute,
                        gint column);
    void (*set_cell_data_func) (GtkCellLayout * cell_layout,
                        GtkCellRenderer * cell,
                        GtkCellLayoutDataFunc func,
                        gpointer func_data,
                        GDestroyNotify destroy);
    void (*clear_attributes) (GtkCellLayout * cell_layout,
                        GtkCellRenderer * cell);
    void (*reorder) (GtkCellLayout * cell_layout, GtkCellRenderer *
cell,

```

```

        gint position);
    GList *(*get_cells) (GtkCellLayout * cell_layout);
    GtkWidgetArea *(*get_area) (GtkCellLayout * cell_layout);
} GtkWidgetCellLayoutIface;
typedef void (*GtkCellLayoutDataFunc) (GtkClipboard * cell_layout,
                                       GtkWidgetCellRenderer * cell,
                                       GtkWidgetTreeModel * tree_model,
                                       GtkWidgetTreeIter * iter, gpointer data);

typedef struct _GtkWidgetToolbar {
    GtkWidgetContainer container;
    GtkWidgetToolbarPrivate *priv;
} GtkWidgetToolbar;
typedef struct _GtkWidgetToolbarPrivate GtkWidgetToolbarPrivate;
typedef struct _GtkWidgetToolbarClass {
    GtkWidgetContainerClass parent_class;
    void (*orientation_changed) (GtkWidgetToolbar * toolbar,
                                GtkWidgetOrientation orientation);
    void (*style_changed) (GtkWidgetToolbar * toolbar, GtkWidgetToolbarStyle
style);
    gboolean (*popup_context_menu) (GtkWidgetToolbar * toolbar, gint x,
gint y,
                                gint button_number);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkWidgetToolbarClass;
typedef struct _GtkWidgetMenuToolButtonClass {
    GtkWidgetToolButtonClass parent_class;
    void (*show_menu) (GtkWidgetMenuToolButton * button);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkWidgetMenuToolButtonClass;
typedef struct _GtkWidgetMenuToolButton {
    GtkWidgetToolButton parent;
    GtkWidgetMenuToolButtonPrivate *priv;
} GtkWidgetMenuToolButton;
typedef struct _GtkWidgetMenuToolButtonPrivate GtkWidgetMenuToolButtonPrivate;
typedef struct _GtkWidgetColorChooserWidget {
    GtkWidgetBox parent_instance;
    GtkWidgetColorChooserWidgetPrivate *priv;
} GtkWidgetColorChooserWidget;
typedef struct _GtkWidgetColorChooserWidgetPrivate
GtkWidgetColorChooserWidgetPrivate;
typedef struct _GtkWidgetColorChooserWidgetClass {
    GtkWidgetBoxClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkWidgetColorChooserWidgetClass;
typedef struct _GtkWidgetStatusbar {
    GtkWidgetBox parent_widget;
    GtkWidgetStatusbarPrivate *priv;
} GtkWidgetStatusbar;
typedef struct _GtkWidgetStatusbarPrivate GtkWidgetStatusbarPrivate;
typedef struct _GtkWidgetStatusbarClass {
    GtkWidgetBoxClass parent_class;
    gpointer reserved;
    void (*text_pushed) (GtkWidgetStatusbar * statusbar, guint context_id,

```

```

        const gchar * text);
void (*text_popped) (GtkStatusbar * statusbar, guint context_id,
        const gchar * text);
void (*_gtk_reserved1) (void);
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
} GtkStatusbarClass;
typedef struct _GtkInvisible {
    GtkWidget widget;
    GtkInvisiblePrivate *priv;
} GtkInvisible;
typedef struct _GtkInvisiblePrivate GtkInvisiblePrivate;
typedef struct _GtkInvisibleClass {
    GtkWidgetClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkInvisibleClass;
typedef struct _GtkExpander {
    GtkBin bin;
    GtkExpanderPrivate *priv;
} GtkExpander;
typedef struct _GtkExpanderClass {
    GtkBinClass parent_class;
    void (*activate) (GtkExpander * expander);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkExpanderClass;
typedef struct _GtkExpanderPrivate GtkExpanderPrivate;
typedef struct _GtkRecentChooserMenu {
    GtkMenu parent_instance;
    GtkRecentChooserMenuPrivate *priv;
} GtkRecentChooserMenu;
typedef struct _GtkRecentChooserMenuClass {
    GtkMenuClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkRecentChooserMenuClass;
typedef struct _GtkRecentChooserMenuPrivate
GtkRecentChooserMenuPrivate;
typedef struct _GtkStyleContextClass {
    GObjectClass parent_class;
    void (*changed) (GtkStyleContext * context);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkStyleContextClass;
typedef struct _GtkStyleContextPrivate GtkStyleContextPrivate;
typedef struct _GtkSeparatorMenuItem {
    GtkMenuItem parent;
    GtkSeparatorMenuItemPrivate *priv;
} GtkSeparatorMenuItem;
typedef struct _GtkSeparatorMenuItemClass {
    GtkMenuItemClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkSeparatorMenuItemClass;

```

```

typedef struct _GtkSeparatorToolItemPrivate
GtkSeparatorToolItemPrivate;
typedef struct _GtkAspectFrame {
    GtkFrame frame;
    GtkAspectFramePrivate *priv;
} GtkAspectFrame;
typedef struct _GtkAspectFramePrivate GtkAspectFramePrivate;
typedef struct _GtkAspectFrameClass {
    GtkFrameClass parent_class;
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
} GtkAspectFrameClass;
typedef struct _GtkLabel {
    GtkMisc misc;
    GtkLabelPrivate *priv;
} GtkLabel;
typedef struct _GtkLabelPrivate GtkLabelPrivate;
typedef struct _GtkLabelClass {
    GtkMiscClass parent_class;
    void (*move_cursor) (GtkLabel * label, GtkMovementStep step,
        gint count, gboolean extend_selection);
    void (*copy_clipboard) (GtkLabel * label);
    void (*populate_popup) (GtkLabel * label, GtkMenu * menu);
    gboolean(*activate_link) (GtkLabel * label, const gchar * uri);
    void (*_gtk_reserved1) (void);
    void (*_gtk_reserved2) (void);
    void (*_gtk_reserved3) (void);
    void (*_gtk_reserved4) (void);
    void (*_gtk_reserved5) (void);
    void (*_gtk_reserved6) (void);
    void (*_gtk_reserved7) (void);
    void (*_gtk_reserved8) (void);
} GtkLabelClass;
typedef struct _GtkLabelSelectionInfo GtkLabelSelectionInfo;
typedef struct _GtkTreeSortable GtkTreeSortable;
typedef struct _GtkTreeSortableIface {
    GTypeInterface g_iface;
    void (*sort_column_changed) (GtkTreeSortable * sortable);
    gboolean(*get_sort_column_id) (GtkTreeSortable * sortable,
        gint * sort_column_id,
        GtkSortType * order);
    void (*set_sort_column_id) (GtkTreeSortable * sortable,
        gint sort_column_id, GtkSortType order);
    void (*set_sort_func) (GtkTreeSortable * sortable, gint
        sort_column_id,
        GtkTreeIterCompareFunc sort_func,
        gpointer user_data, GDestroyNotify destroy);
    void (*set_default_sort_func) (GtkTreeSortable * sortable,
        GtkTreeIterCompareFunc sort_func,
        gpointer user_data,
        GDestroyNotify destroy);
    gboolean(*has_default_sort_func) (GtkTreeSortable * sortable);
} GtkTreeSortableIface;
typedef gint(*GtkTreeIterCompareFunc) (GtkTreeModel * model,
        GtkTreeIter * a, GtkTreeIter * b,
        gpointer user_data);
typedef struct _GtkFileChooserButton {
    GtkBox parent;
    GtkFileChooserButtonPrivate *priv;
} GtkFileChooserButton;
typedef struct _GtkFileChooserButtonPrivate
GtkFileChooserButtonPrivate;
typedef struct _GtkFileChooserButtonClass {
    GtkBoxClass parent_class;

```

```

    void (*file_set) (GtkFileChooserButton * fc);
    void *__gtk_reserved1;
    void *__gtk_reserved2;
    void *__gtk_reserved3;
    void *__gtk_reserved4;
} GtkFileChooserButtonClass;
typedef struct _GtkAccelMap GtkAccelMap;
typedef struct _GtkAccelMapClass GtkAccelMapClass;
typedef void (*GtkAccelMapForeach) (void);
typedef struct _GtkProgressBar {
    GtkWidget parent;
    GtkProgressBarPrivate *priv;
} GtkProgressBar;
typedef struct _GtkProgressBarPrivate GtkProgressBarPrivate;
typedef struct _GtkProgressBarClass {
    GtkWidgetClass parent_class;
    void (*__gtk_reserved1) (void);
    void (*__gtk_reserved2) (void);
    void (*__gtk_reserved3) (void);
    void (*__gtk_reserved4) (void);
} GtkProgressBarClass;
typedef struct _GtkContainer {
    GtkWidget widget;
    GtkContainerPrivate *priv;
} GtkContainer;
typedef struct _GtkContainerPrivate GtkContainerPrivate;
typedef struct _GtkContainerClass {
    GtkWidgetClass parent_class;
    void (*add) (GtkContainer * container, GtkWidget * widget);
    void (*remove) (GtkContainer * container, GtkWidget * widget);
    void (*check_resize) (GtkContainer * container);
    void (*forall) (GtkContainer * container, gboolean
include_internals,
    GtkCallback callback, gpointer callback_data);
    void (*set_focus_child) (GtkContainer * container, GtkWidget *
widget);
    GType(*child_type) (GtkContainer * container);
    gchar *(*composite_name) (GtkContainer * container, GtkWidget *
child);
    void (*set_child_property) (GtkContainer * container,
    GtkWidget * child, guint property_id,
    const GValue * value, GParamSpec *
pspec);
    void (*get_child_property) (GtkContainer * container,
    GtkWidget * child, guint property_id,
    GValue * value, GParamSpec * pspec);
    GtkWidgetPath *(*get_path_for_child) (GtkContainer * container,
    GtkWidget * child);
    unsigned int _handle_border_width;
    void (*__gtk_reserved1) (void);
    void (*__gtk_reserved2) (void);
    void (*__gtk_reserved3) (void);
    void (*__gtk_reserved4) (void);
    void (*__gtk_reserved5) (void);
    void (*__gtk_reserved6) (void);
    void (*__gtk_reserved7) (void);
    void (*__gtk_reserved8) (void);
} GtkContainerClass;
typedef struct _GtkIconView {
    GtkContainer parent;
    GtkIconViewPrivate *priv;
} GtkIconView;
typedef struct _GtkIconViewClass {
    GtkContainerClass parent_class;
    void (*item_activated) (GtkIconView * icon_view, GtkTreePath *
path);

```

```

void (*selection_changed) (GtkIconView * icon_view);
void (*select_all) (GtkIconView * icon_view);
void (*unselect_all) (GtkIconView * icon_view);
void (*select_cursor_item) (GtkIconView * icon_view);
void (*toggle_cursor_item) (GtkIconView * icon_view);
gboolean(*move_cursor) (GtkIconView * icon_view,
GtkMovementStep step,
gint count);
gboolean(*activate_cursor_item) (GtkIconView * icon_view);
void (*_gtk_reserved1) (void);
void (*_gtk_reserved2) (void);
void (*_gtk_reserved3) (void);
void (*_gtk_reserved4) (void);
} GtkIconViewClass;
typedef struct _GtkIconViewPrivate GtkIconViewPrivate;
typedef void (*GtkIconViewForeachFunc) (GtkIconView * icon_view,
GtkTreePath * path, gpointer
data);
typedef enum {
GTK_ICON_VIEW_NO_DROP,
GTK_ICON_VIEW_DROP_INTRO,
GTK_ICON_VIEW_DROP_LEFT,
GTK_ICON_VIEW_DROP_RIGHT,
GTK_ICON_VIEW_DROP_ABOVE,
GTK_ICON_VIEW_DROP_BELOW
} GtkIconViewDropPosition;
typedef enum {
GTK_RC_FG = 1 << 0,
GTK_RC_BG = 1 << 1,
GTK_RC_TEXT = 1 << 2,
GTK_RC_BASE = 1 << 3
} GtkRcFlags;
extern void gtk_about_dialog_add_credit_section(GtkAboutDialog *
about,
const char *,
const char **);
extern const char *gtk_about_dialog_get_artists(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_authors(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_comments(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_copyright(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_documenters(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_license(GtkAboutDialog *
about);
extern GtkLicense gtk_about_dialog_get_license_type(GtkAboutDialog
*
about);
extern GdkPixbuf *gtk_about_dialog_get_logo(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_logo_icon_name(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_program_name(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_translator_credits(GtkAboutDialog *
about);
extern GType gtk_about_dialog_get_type(void);

```

```

extern const char *gtk_about_dialog_get_version(GtkAboutDialog *
about);
extern const char *gtk_about_dialog_get_website(GtkAboutDialog *
about);
extern          const          char
*gtk_about_dialog_get_website_label(GtkAboutDialog *
          about);
extern gboolean gtk_about_dialog_get_wrap_license(GtkAboutDialog *
about);
extern GtkWidget *gtk_about_dialog_new(void);
extern void gtk_about_dialog_set_artists(GtkAboutDialog * about,
          const char **);
extern void gtk_about_dialog_set_authors(GtkAboutDialog * about,
          const char **);
extern void gtk_about_dialog_set_comments(GtkAboutDialog * about,
          const char *);
extern void gtk_about_dialog_set_copyright(GtkAboutDialog * about,
          const char *);
extern void gtk_about_dialog_set_documenters(GtkAboutDialog *
about,
          const char **);
extern void gtk_about_dialog_set_license(GtkAboutDialog * about,
          const char *);
extern void gtk_about_dialog_set_license_type(GtkAboutDialog *
about,
          GtkLicense);
extern void gtk_about_dialog_set_logo(GtkAboutDialog * about,
GdkPixbuf *);
extern void gtk_about_dialog_set_logo_icon_name(GtkAboutDialog *
about,
          const char *);
extern void gtk_about_dialog_set_program_name(GtkAboutDialog *
about,
          const char *);
extern void gtk_about_dialog_set_translator_credits(GtkAboutDialog
* about,
          const char *);
extern void gtk_about_dialog_set_version(GtkAboutDialog * about,
          const char *);
extern void gtk_about_dialog_set_website(GtkAboutDialog * about,
          const char *);
extern void gtk_about_dialog_set_website_label(GtkAboutDialog *
about,
          const char *);
extern void gtk_about_dialog_set_wrap_license(GtkAboutDialog *
about,
          gboolean);
extern GType gtk_accel_flags_get_type(void);
extern gboolean gtk_accel_group_activate(GtkAccelGroup *
accel_group,
          GQuark accel_quark,
          GObject * accelratable,
          guint accel_key,
          GdkModifierType accel_mods);
extern void gtk_accel_group_connect(GtkAccelGroup * accel_group,
          guint accel_key,
          GdkModifierType accel_mods,
          GtkAccelFlags accel_flags,
          GClosure * closure);
extern void gtk_accel_group_connect_by_path(GtkAccelGroup *
accel_group,
          const char *, GClosure *);
extern gboolean gtk_accel_group_disconnect(GtkAccelGroup *
accel_group,
          GClosure *);

```



```

extern gboolean gtk_accel_group_disconnect_key(GtkAccelGroup *
accel_group,
                                guint accel_key,
                                GdkModifierType
accel_mods);
extern GtkAccelKey *gtk_accel_group_find(GtkAccelGroup *
accel_group,
                                GtkAccelGroupFindFunc,
gpointer);
extern GtkAccelGroup *gtk_accel_group_from_accel_closure(GClosure
*
                                closure);
extern gboolean gtk_accel_group_get_is_locked(GtkAccelGroup *
accel_group);
extern GdkModifierType
gtk_accel_group_get_modifier_mask(GtkAccelGroup *
                                accel_group);
extern GType gtk_accel_group_get_type(void);
extern void gtk_accel_group_lock(GtkAccelGroup * accel_group);
extern GtkAccelGroup *gtk_accel_group_new(void);
extern GtkAccelGroupEntry *gtk_accel_group_query(GtkAccelGroup *
                                accel_group,
                                guint accel_key,
                                GdkModifierType
                                accel_mods,
                                guint n_entries);
extern void gtk_accel_group_unlock(GtkAccelGroup * accel_group);
extern gboolean gtk_accel_groups_activate(GObject * object,
                                guint accel_key,
                                GdkModifierType accel_mods);
extern GSList *gtk_accel_groups_from_object(GObject * object);
extern GtkWidget *gtk_accel_label_get_accel_widget(GtkAccelLabel *
                                accel_label);
extern guint gtk_accel_label_get_accel_width(GtkAccelLabel *
accel_label);
extern GType gtk_accel_label_get_type(void);
extern GtkWidget *gtk_accel_label_new(const char *string);
extern gboolean gtk_accel_label_refetch(GtkAccelLabel *
accel_label);
extern void gtk_accel_label_set_accel(GtkAccelLabel * accel_label,
                                guint accelerator_key,
                                GdkModifierType accelerator_mods);
extern void gtk_accel_label_set_accel_closure(GtkAccelLabel *
accel_label,
                                GClosure *);
extern void gtk_accel_label_set_accel_widget(GtkAccelLabel *
accel_label,
                                GtkWidget *);
extern void gtk_accel_map_add_entry(const gchar * accel_path,
                                guint accel_key,
                                GdkModifierType accel_mods);
extern void gtk_accel_map_add_filter(const char *filter_pattern);
extern gboolean gtk_accel_map_change_entry(const gchar * accel_path,
                                guint accel_key,
                                GdkModifierType accel_mods,
                                gboolean replace);
extern void gtk_accel_map_foreach(gpointer data,
GtkAccelMapForeach);
extern void gtk_accel_map_foreach_unfiltered(gpointer data,
                                GtkAccelMapForeach);
extern GtkAccelMap *gtk_accel_map_get(void);
extern GType gtk_accel_map_get_type(void);
extern void gtk_accel_map_load(const char *file_name);
extern void gtk_accel_map_load_fd(gint fd);
extern void gtk_accel_map_load_scanner(GScanner * scanner);
extern void gtk_accel_map_lock_path(const char *accel_path);

```

```

extern gboolean gtk_accel_map_lookup_entry(const char *accel_path,
                                           GtkAccelKey *);
extern void gtk_accel_map_save(const char *file_name);
extern void gtk_accel_map_save_fd(gint fd);
extern void gtk_accel_map_unlock_path(const char *accel_path);
extern GdkModifierType gtk_accelerator_get_default_mod_mask(void);
extern gchar *gtk_accelerator_get_label(guint accelerator_key,
                                       GdkModifierType
                                       accelerator_mods);
extern gchar *gtk_accelerator_get_label_with_keycode(GdkDisplay *
display,
                                                    guint accel_key,
                                                    guint keycode,
                                                    GdkModifierType
                                                    accel_mods);
extern gchar *gtk_accelerator_name(guint accelerator_key,
                                   GdkModifierType accel_key);
extern gchar *gtk_accelerator_name_with_keycode(GdkDisplay *
display,
                                                    guint accel_key,
                                                    guint keycode,
                                                    GdkModifierType
                                                    accel_mods);
extern void gtk_accelerator_parse(const char *accelerator, guint *,
                                   GdkModifierType *);
extern void gtk_accelerator_parse_with_keycode(const char
*accelerator,
                                                    guint *, guint * *,
                                                    GdkModifierType *);
extern void gtk_accelerator_set_default_mod_mask(GdkModifierType
default_mod_mask);
extern gboolean gtk_accelerator_valid(guint keyval,
                                       GdkModifierType modifiers);
extern void gtk_accessible_connect_widget_destroyed(GtkAccessible
*
accessible);
extern GType gtk_accessible_get_type(void);
extern GtkWidget *gtk_accessible_get_widget(GtkAccessible *
accessible);
extern void gtk_accessible_set_widget(GtkAccessible * accessible,
                                       GtkWidget *);
extern void gtk_action_activate(GtkAction * action);
extern void gtk_action_block_activate(GtkAction * action);
extern void gtk_action_connect_accelerator(GtkAction * action);
extern GtkWidget *gtk_action_create_icon(GtkAction * action,
GtkIconSize);
extern GtkWidget *gtk_action_create_menu(GtkAction * action);
extern GtkWidget *gtk_action_create_menu_item(GtkAction * action);
extern GtkWidget *gtk_action_create_tool_item(GtkAction * action);
extern void gtk_action_disconnect_accelerator(GtkAction * action);
extern GClosure *gtk_action_get_accel_closure(GtkAction * action);
extern const char *gtk_action_get_accel_path(GtkAction * action);
extern gboolean gtk_action_get_always_show_image(GtkAction *
action);
extern GIcon *gtk_action_get_gicon(GtkAction * action);
extern const char *gtk_action_get_icon_name(GtkAction * action);
extern gboolean gtk_action_get_is_important(GtkAction * action);
extern const char *gtk_action_get_label(GtkAction * action);
extern const char *gtk_action_get_name(GtkAction * action);
extern GSList *gtk_action_get_proxies(GtkAction * action);
extern gboolean gtk_action_get_sensitive(GtkAction * action);
extern const char *gtk_action_get_short_label(GtkAction * action);
extern const char *gtk_action_get_stock_id(GtkAction * action);
extern const char *gtk_action_get_tooltip(GtkAction * action);
extern GType gtk_action_get_type(void);
extern gboolean gtk_action_get_visible(GtkAction * action);

```

```

extern gboolean gtk_action_get_visible_horizontal(GtkAction *
action);
extern gboolean gtk_action_get_visible_vertical(GtkAction *
action);
extern void gtk_action_group_add_action(GtkActionGroup *
action_group,
                                   GtkAction *);
extern void gtk_action_group_add_action_with_accel(GtkActionGroup *
action_group,
                                   GtkAction *,
                                   const char *);
extern void gtk_action_group_add_actions(GtkActionGroup *
action_group,
                                   const GtkActionEntry *, guint,
                                   gpointer);
extern void gtk_action_group_add_actions_full(GtkActionGroup *
action_group,
                                   const GtkActionEntry *,
                                   guint, gpointer,
                                   GDestroyNotify);
extern void gtk_action_group_add_radio_actions(GtkActionGroup *
action_group,
                                   const GtkRadioActionEntry *,
                                   guint, gint, GCallback,
                                   gpointer);
extern void gtk_action_group_add_radio_actions_full(GtkActionGroup *
action_group,
                                   const
                                   GtkRadioActionEntry *,
                                   guint, gint, GCallback,
                                   gpointer,
                                   GDestroyNotify);
extern void gtk_action_group_add_toggle_actions(GtkActionGroup *
action_group,
                                   const GtkToggleActionEntry
                                   *, guint, gpointer);
extern void gtk_action_group_add_toggle_actions_full(GtkActionGroup *
action_group,
                                   const
                                   GtkToggleActionEntry
                                   *, guint, gpointer,
                                   GDestroyNotify);
extern GtkAccelGroup *gtk_action_group_get_accel_group(GtkActionGroup *
action_group);
extern GtkAction *gtk_action_group_get_action(GtkActionGroup *
action_group, const char *);
extern const char *gtk_action_group_get_name(GtkActionGroup *
action_group);
extern gboolean gtk_action_group_get_sensitive(GtkActionGroup *
action_group);
extern GType gtk_action_group_get_type(void);
extern gboolean gtk_action_group_get_visible(GtkActionGroup *
action_group);
extern GList *gtk_action_group_list_actions(GtkActionGroup *
action_group);
extern GtkActionGroup *gtk_action_group_new(const char *name);
extern void gtk_action_group_remove_action(GtkActionGroup *
action_group,
                                   GtkAction *);
extern void gtk_action_group_set_accel_group(GtkActionGroup *
action_group,
                                   GtkAccelGroup *);

```

```

extern void gtk_action_group_set_sensitive(GtkActionGroup *
action_group,
                                           gboolean);
extern void gtk_action_group_set_translate_func(GtkActionGroup *
action_group,
                                           GtkTranslateFunc, gpointer,
                                           GDestroyNotify);
extern void gtk_action_group_set_translation_domain(GtkActionGroup
*
                                           action_group,
                                           const char *);
extern void gtk_action_group_set_visible(GtkActionGroup *
action_group,
                                           gboolean);
extern const char
*gtk_action_group_translate_string(GtkActionGroup *
action_group,
const char *);
extern gboolean gtk_action_is_sensitive(GtkAction * action);
extern gboolean gtk_action_is_visible(GtkAction * action);
extern GtkAction *gtk_action_new(const char *name, const char *,
const char *, const char *);
extern void gtk_action_set_accel_group(GtkAction * action,
GtkAccelGroup *);
extern void gtk_action_set_accel_path(GtkAction * action, const
char *);
extern void gtk_action_set_always_show_image(GtkAction * action,
gboolean);
extern void gtk_action_set_gicon(GtkAction * action, GIcon *);
extern void gtk_action_set_icon_name(GtkAction * action, const char
*);
extern void gtk_action_set_is_important(GtkAction * action,
gboolean);
extern void gtk_action_set_label(GtkAction * action, const char *);
extern void gtk_action_set_sensitive(GtkAction * action, gboolean);
extern void gtk_action_set_short_label(GtkAction * action, const
char *);
extern void gtk_action_set_stock_id(GtkAction * action, const char
*);
extern void gtk_action_set_tooltip(GtkAction * action, const char
*);
extern void gtk_action_set_visible(GtkAction * action, gboolean);
extern void gtk_action_set_visible_horizontal(GtkAction * action,
gboolean);
extern void gtk_action_set_visible_vertical(GtkAction * action,
gboolean);
extern void gtk_action_unblock_activate(GtkAction * action);
extern const char *gtk_actionable_get_action_name(GtkActionable *
actionable);
extern GVariant
*gtk_actionable_get_action_target_value(GtkActionable *
actionable);
extern GType gtk_actionable_get_type(void);
extern void gtk_actionable_set_action_name(GtkActionable *
actionable,
const char *);
extern void gtk_actionable_set_action_target(GtkActionable *
actionable,
const char *, ...);
extern void gtk_actionable_set_action_target_value(GtkActionable *
actionable, GVariant *);
extern void gtk_actionable_set_detailed_action_name(GtkActionable
*
actionable,
const char *);
extern void gtk_activatable_do_set_related_action(GtkActivatable *

```

```

        activatable,
        GtkAction *);

extern          GtkAction
*gtk_activatable_get_related_action(GtkActivatable *
        activatable);

extern GType gtk_activatable_get_type(void);
extern          gboolean
gtk_activatable_get_use_action_appearance(GtkActivatable *
        activatable);

extern void gtk_activatable_set_related_action(GtkActivatable *
        activatable, GtkAction *);

extern          void
gtk_activatable_set_use_action_appearance(GtkActivatable *
        activatable,
        gboolean);

extern void gtk_activatable_sync_action_properties(GtkActivatable
*
        activatable,
        GtkAction *);

extern void gtk_adjustment_changed(GtkAdjustment * adjustment);
extern void gtk_adjustment_clamp_page(GtkAdjustment * adjustment,
        gdouble,
        gdouble);

extern void gtk_adjustment_configure(GtkAdjustment * adjustment,
        gdouble,
        gdouble, gdouble, gdouble, gdouble,
        gdouble);

extern gdouble gtk_adjustment_get_lower(GtkAdjustment *
        adjustment);
extern gdouble gtk_adjustment_get_minimum_increment(GtkAdjustment
*
        adjustment);

extern gdouble gtk_adjustment_get_page_increment(GtkAdjustment *
        adjustment);
extern gdouble gtk_adjustment_get_page_size(GtkAdjustment *
        adjustment);
extern gdouble gtk_adjustment_get_step_increment(GtkAdjustment *
        adjustment);

extern GType gtk_adjustment_get_type(void);
extern gdouble gtk_adjustment_get_upper(GtkAdjustment *
        adjustment);
extern gdouble gtk_adjustment_get_value(GtkAdjustment *
        adjustment);
extern GtkAdjustment *gtk_adjustment_new(gdouble value, gdouble,
        gdouble,
        gdouble, gdouble, gdouble);

extern void gtk_adjustment_set_lower(GtkAdjustment * adjustment,
        gdouble);
extern void gtk_adjustment_set_page_increment(GtkAdjustment *
        adjustment,
        gdouble);
extern void gtk_adjustment_set_page_size(GtkAdjustment *
        adjustment,
        gdouble);
extern void gtk_adjustment_set_step_increment(GtkAdjustment *
        adjustment,
        gdouble);
extern void gtk_adjustment_set_upper(GtkAdjustment * adjustment,
        gdouble);
extern void gtk_adjustment_set_value(GtkAdjustment * adjustment,
        gdouble);
extern void gtk_adjustment_value_changed(GtkAdjustment *
        adjustment);
extern GType gtk_align_get_type(void);
extern void gtk_alignment_get_padding(GtkAlignment * alignment,
        guint *,

```

```

                                guint *, guint *, guint *);
extern GType gtk_alignment_get_type(void);
extern GtkWidget *gtk_alignment_new(gfloat xalign, gfloat, gfloat,
gfloat);
extern void gtk_alignment_set(GtkAlignment * alignment, gfloat,
gfloat,
                                gfloat, gfloat);
extern void gtk_alignment_set_padding(GtkAlignment * alignment,
guint,
                                guint, guint, guint);
extern gboolean gtk_alternative_dialog_button_order(GdkScreen *
screen);
extern
                                void
gtk_app_chooser_button_append_custom_item(GtkAppChooserButton *
                                self, const char *,
                                const char *,
                                GIcon *);

extern
                                void
gtk_app_chooser_button_append_separator(GtkAppChooserButton *
                                self);

extern
                                const
                                char
*gtk_app_chooser_button_get_heading(GtkAppChooserButton *
                                self);

extern gboolean
gtk_app_chooser_button_get_show_default_item(GtkAppChooserButton *
self);
extern gboolean
gtk_app_chooser_button_get_show_dialog_item(GtkAppChooserButton *
self);
extern GType gtk_app_chooser_button_get_type(void);
extern GtkWidget *gtk_app_chooser_button_new(const char
*content_type);
extern void
gtk_app_chooser_button_set_active_custom_item(GtkAppChooserButton
* self,
                                const char *);
extern void gtk_app_chooser_button_set_heading(GtkAppChooserButton
* self,
                                const char *);

extern void
gtk_app_chooser_button_set_show_default_item(GtkAppChooserButton *
self,
                                gboolean);

extern
                                void
gtk_app_chooser_button_set_show_dialog_item(GtkAppChooserButton
* self, gboolean);

extern
                                const
                                char
*gtk_app_chooser_dialog_get_heading(GtkAppChooserDialog *
                                self);
extern GType gtk_app_chooser_dialog_get_type(void);
extern
                                GtkWidget
*gtk_app_chooser_dialog_get_widget(GtkAppChooserDialog *
                                self);
extern GtkWidget *gtk_app_chooser_dialog_new(GtkWindow * parent,
GtkDialogFlags, GFile *);
extern
                                GtkWidget
*gtk_app_chooser_dialog_new_for_content_type(GtkWindow *
                                parent,
                                GtkDialogFlags,
                                const char
                                *);
extern void gtk_app_chooser_dialog_set_heading(GtkAppChooserDialog
* self,
                                const char *);
extern GAppInfo *gtk_app_chooser_get_app_info(GtkAppChooser *
self);

```

```

extern gchar *gtk_app_chooser_get_content_type(GtkAppChooser *
self);
extern GType gtk_app_chooser_get_type(void);
extern void gtk_app_chooser_refresh(GtkAppChooser * self);
extern const char
*gtk_app_chooser_widget_get_default_text(GtkAppChooserWidget *
self);
extern
gtk_app_chooser_widget_get_show_all(GtkAppChooserWidget *
self);
extern
gtk_app_chooser_widget_get_show_default(GtkAppChooserWidget
* self);
extern gboolean
gtk_app_chooser_widget_get_show_fallback(GtkAppChooserWidget *
self);
extern
gtk_app_chooser_widget_get_show_other(GtkAppChooserWidget *
self);
extern gboolean
gtk_app_chooser_widget_get_show_recommended(GtkAppChooserWidget *
self);
extern GType gtk_app_chooser_widget_get_type(void);
extern GtkWidget *gtk_app_chooser_widget_new(const char
*content_type);
extern
gtk_app_chooser_widget_set_default_text(GtkAppChooserWidget *
self, const char *);
extern
gtk_app_chooser_widget_set_show_all(GtkAppChooserWidget * self,
gboolean);
extern
gtk_app_chooser_widget_set_show_default(GtkAppChooserWidget *
self, gboolean);
extern
gtk_app_chooser_widget_set_show_fallback(GtkAppChooserWidget *
self, gboolean);
extern
gtk_app_chooser_widget_set_show_other(GtkAppChooserWidget *
self, gboolean);
extern
gtk_app_chooser_widget_set_show_recommended(GtkAppChooserWidget
* self, gboolean);
extern void gtk_application_add_accelerator(GtkApplication *
application,
const char *, const char *,
GVariant *);
extern void gtk_application_add_window(GtkApplication *
application,
GtkWindow *);
extern GtkWindow *gtk_application_get_active_window(GtkApplication
*
application);
extern GMenuModel *gtk_application_get_app_menu(GtkApplication *
application);
extern GMenuModel *gtk_application_get_menubar(GtkApplication *
application);
extern GType gtk_application_get_type(void);
extern GtkWindow *gtk_application_get_window_by_id(GtkApplication
*
application, guint);
extern GList *gtk_application_get_windows(GtkApplication *
application);
extern guint gtk_application_inhibit(GtkApplication * application,
GtkWindow *,
GtkApplicationInhibitFlags,

```

```

        const char *);
extern GType gtk_application_inhibit_flags_get_type(void);
extern gboolean gtk_application_is_inhibited(GtkApplication *
application,
        GtkApplicationInhibitFlags);
extern GtkApplication *gtk_application_new(const char
*application_id,
        GApplicationFlags);
extern void gtk_application_remove_accelerator(GtkApplication *
application, const char *,
        GVariant *);
extern void gtk_application_remove_window(GtkApplication *
application,
        GtkWindow *);
extern void gtk_application_set_app_menu(GtkApplication *
application,
        GMenuModel *);
extern void gtk_application_set_menubar(GtkApplication *
application,
        GMenuModel *);
extern void gtk_application_uninhibit(GtkApplication * application,
guint);
extern guint gtk_application_window_get_id(GtkApplicationWindow *
window);
extern gboolean
gtk_application_window_get_show_menubar(GtkApplicationWindow *
window);
extern GType gtk_application_window_get_type(void);
extern GtkWidget *gtk_application_window_new(GtkApplication *
application);
extern void
gtk_application_window_set_show_menubar(GtkApplicationWindow *
window, gboolean);
extern GType gtk_arrow_get_type(void);
extern GtkWidget *gtk_arrow_new(GtkArrowType arrow_type,
GtkShadowType);
extern GType gtk_arrow_placement_get_type(void);
extern void gtk_arrow_set(GtkArrow * arrow, GtkArrowType,
GtkShadowType);
extern GType gtk_arrow_type_get_type(void);
extern GType gtk_aspect_frame_get_type(void);
extern GtkWidget *gtk_aspect_frame_new(const char *label, gfloat,
gfloat,
        gfloat, gboolean);
extern void gtk_aspect_frame_set(GtkAspectRatioFrame * aspect_frame,
gfloat,
        gfloat, gfloat, gboolean);
extern void gtk_assistant_add_action_widget(GtkAssistant *
assistant,
        GtkWidget *);
extern gint gtk_assistant_append_page(GtkAssistant * assistant,
        GtkWidget *);
extern void gtk_assistant_commit(GtkAssistant * assistant);
extern gint gtk_assistant_get_current_page(GtkAssistant *
assistant);
extern gint gtk_assistant_get_n_pages(GtkAssistant * assistant);
extern GtkWidget *gtk_assistant_get_nth_page(GtkAssistant *
assistant,
        gint);
extern gboolean gtk_assistant_get_page_complete(GtkAssistant *
assistant,
        GtkWidget *);
extern GdkPixbuf *gtk_assistant_get_page_header_image(GtkAssistant
*
        assistant,
        GtkWidget *);

```



```

extern GdkPixbuf *gtk_assistant_get_page_side_image(GtkAssistant *
                                                    assistant,
                                                    GtkWidget *);
extern const char *gtk_assistant_get_page_title(GtkAssistant *
assistant,
                                                    GtkWidget *);
extern
                                                    GtkWidget *);
gtk_assistant_get_page_type(GtkAssistant *
                                                    GtkAssistantPageType
                                                    assistant,
                                                    GtkWidget *);

extern GType gtk_assistant_get_type(void);
extern gint gtk_assistant_insert_page(GtkAssistant * assistant,
                                      GtkWidget *, gint);
extern GtkWidget *gtk_assistant_new(void);
extern void gtk_assistant_next_page(GtkAssistant * assistant);
extern GType gtk_assistant_page_type_get_type(void);
extern gint gtk_assistant_prepend_page(GtkAssistant * assistant,
                                       GtkWidget *);
extern void gtk_assistant_previous_page(GtkAssistant * assistant);
extern void gtk_assistant_remove_action_widget(GtkAssistant *
assistant,
                                                GtkWidget *);
extern void gtk_assistant_remove_page(GtkAssistant * assistant,
gint);
extern void gtk_assistant_set_current_page(GtkAssistant *
assistant, gint);
extern void gtk_assistant_set_forward_page_func(GtkAssistant *
assistant,
                                                GtkAssistantPageFunc,
                                                gpointer, GDestroyNotify);
extern void gtk_assistant_set_page_complete(GtkAssistant *
assistant,
                                                GtkWidget *, gboolean);
extern void gtk_assistant_set_page_header_image(GtkAssistant *
assistant,
                                                GtkWidget *, GdkPixbuf *);
extern void gtk_assistant_set_page_side_image(GtkAssistant *
assistant,
                                                GtkWidget *, GdkPixbuf *);
extern void gtk_assistant_set_page_title(GtkAssistant * assistant,
                                         GtkWidget *, const char *);
extern void gtk_assistant_set_page_type(GtkAssistant * assistant,
                                         GtkWidget *,
                                         GtkAssistantPageType);
extern void gtk_assistant_update_buttons_state(GtkAssistant *
assistant);
extern GType gtk_attach_options_get_type(void);
extern GtkWidget *gtk_bin_get_child(GtkBin * bin);
extern GType gtk_bin_get_type(void);
extern void gtk_binding_entry_add_signal(GtkBindingSet *
binding_set,
                                         guint keyval,
                                         GdkModifierType modifiers,
                                         const gchar * signal_name,
                                         guint n_args, ...);
extern
                                                    GTokenType
gtk_binding_entry_add_signal_from_string(GtkBindingSet *
binding_set,
                                         const char *);
extern void gtk_binding_entry_add_signal(GtkBindingSet *
binding_set,
                                         guint keyval,
                                         GdkModifierType modifiers,
                                         const gchar * signal_name,
                                         GSList * binding_args);
extern void gtk_binding_entry_remove(GtkBindingSet * binding_set,

```

```

        guint keyval,
        GdkModifierType modifiers);
extern void gtk_binding_entry_skip(GtkBindingSet * binding_set,
        guint keyval,
        GdkModifierType modifiers);
extern gboolean gtk_binding_set_activate(GtkBindingSet *
binding_set,
        guint keyval,
        GdkModifierType modifiers,
        GObject * object);
extern void gtk_binding_set_add_path(GtkBindingSet * binding_set,
        GtkPathType, const char *,
        GtkPathPriorityType);
extern GtkBindingSet *gtk_binding_set_by_class(gpointer
object_class);
extern GtkBindingSet *gtk_binding_set_find(const char *set_name);
extern GtkBindingSet *gtk_binding_set_new(const char *set_name);
extern gboolean gtk_bindings_activate(GObject * object, guint
keyval,
        GdkModifierType modifiers);
extern gboolean gtk_bindings_activate_event(GObject * object,
        GdkEventKey *);
extern GtkBorder *gtk_border_copy(const GtkBorder * border_);
extern void gtk_border_free(GtkBorder * border_);
extern GType gtk_border_get_type(void);
extern GtkBorder *gtk_border_new(void);
extern GType gtk_border_style_get_type(void);
extern gboolean gtk_box_get_homogeneous(GtkBox * box);
extern gint gtk_box_get_spacing(GtkBox * box);
extern GType gtk_box_get_type(void);
extern GtkWidget *gtk_box_new(GtkOrientation orientation, gint);
extern void gtk_box_pack_end(GtkBox * box, GtkWidget *, gboolean,
gboolean,
        guint);
extern void gtk_box_pack_start(GtkBox * box, GtkWidget *, gboolean,
        gboolean, guint);
extern void gtk_box_query_child_packing(GtkBox * box, GtkWidget *,
        gboolean *, gboolean *, guint *,
        GtkPackType *);
extern void gtk_box_reorder_child(GtkBox * box, GtkWidget *, gint);
extern void gtk_box_set_child_packing(GtkBox * box, GtkWidget *,
        gboolean,
        gboolean, guint, GtkPackType);
extern void gtk_box_set_homogeneous(GtkBox * box, gboolean);
extern void gtk_box_set_spacing(GtkBox * box, gint);
extern void gtk_buildable_add_child(GtkBuildable * buildable,
        GtkBuilder *,
        GObject *, const char *);
extern GObject *gtk_buildable_construct_child(GtkBuildable *
buildable,
        GtkBuilder *, const char *);
extern void gtk_buildable_custom_finished(GtkBuildable * buildable,
        GtkBuilder *, GObject *,
        const char *, void *);
extern void gtk_buildable_custom_tag_end(GtkBuildable * buildable,
        GtkBuilder *, GObject *,
        const char *, void **);
extern gboolean gtk_buildable_custom_tag_start(GtkBuildable *
buildable,
        GtkBuilder *, GObject *,
        const char *,
        GMarkupParser *, void **);
extern GObject *gtk_buildable_get_internal_child(GtkBuildable *
buildable,
        GtkBuilder *,
        const char *);

```

```

extern const char *gtk_buildable_get_name(GtkBuildable * buildable);
extern GType gtk_buildable_get_type(void);
extern void gtk_buildable_parser_finished(GtkBuildable * buildable,
                                           GtkBuilder *);
extern void gtk_buildable_set_buildable_property(GtkBuildable *
buildable,
                                           GtkBuilder *,
                                           const char *,
                                           const GValue *);
extern void gtk_buildable_set_name(GtkBuildable * buildable, const
char *);
extern guint gtk_builder_add_from_file(GtkBuilder * builder, const
char *,
                                           GError * *);
extern guint gtk_builder_add_from_resource(GtkBuilder * builder,
                                           const char *, GError * *);
extern guint gtk_builder_add_from_string(GtkBuilder * builder,
                                           const char *, gsize, GError * *);
extern guint gtk_builder_add_objects_from_file(GtkBuilder *
builder,
                                           const char *, gchar * *,
                                           GError * *);
extern guint gtk_builder_add_objects_from_resource(GtkBuilder *
builder,
                                           const char *, gchar * *,
                                           GError * *);
extern guint gtk_builder_add_objects_from_string(GtkBuilder *
builder,
                                           const char *, gsize,
                                           gchar * *, GError * *);
extern void gtk_builder_connect_signals(GtkBuilder * builder,
gpointer);
extern void gtk_builder_connect_signals_full(GtkBuilder * builder,
                                           GtkBuilderConnectFunc,
                                           gpointer);
extern GType gtk_builder_error_get_type(void);
extern GQuark gtk_builder_error_quark(void);
extern GObject *gtk_builder_get_object(GtkBuilder * builder, const
char *);
extern GSList *gtk_builder_get_objects(GtkBuilder * builder);
extern const char *gtk_builder_get_translation_domain(GtkBuilder *
builder);
extern GType gtk_builder_get_type(void);
extern GType gtk_builder_get_type_from_name(GtkBuilder * builder,
                                           const char *);
extern GtkBuilder *gtk_builder_new(void);
extern void gtk_builder_set_translation_domain(GtkBuilder *
builder,
                                           const char *);
extern gboolean gtk_builder_value_from_string(GtkBuilder * builder,
                                           GParamSpec *, const char *,
                                           GValue *, GError * *);
extern gboolean gtk_builder_value_from_string_type(GtkBuilder *
builder,
                                           GType, const char *,
                                           GValue *, GError * *);
extern
gboolean
gtk_button_box_get_child_non_homogeneous(GtkButtonBox *
widget,
                                           GtkWidget *);
extern gboolean gtk_button_box_get_child_secondary(GtkButtonBox *
widget,
                                           GtkWidget *);
extern GtkButtonBoxStyle gtk_button_box_get_layout(GtkButtonBox *
widget);
extern GType gtk_button_box_get_type(void);

```

```

extern GtkWidget *gtk_button_box_new(GtkOrientation orientation);
extern void gtk_button_box_set_child_non_homogeneous(GtkButtonBox
* widget,
                                GtkWidget *,
                                gboolean);
extern void gtk_button_box_set_child_secondary(GtkButtonBox *
widget,
                                GtkWidget *, gboolean);
extern void gtk_button_box_set_layout(GtkButtonBox * widget,
                                GtkButtonBoxStyle);
extern GType gtk_button_box_style_get_type(void);
extern void gtk_button_clicked(GtkButton * button);
extern void gtk_button_enter(GtkButton * button);
extern void gtk_button_get_alignment(GtkButton * button, gfloat *,
                                gfloat *);
extern gboolean gtk_button_get_always_show_image(GtkButton *
button);
extern GdkWindow *gtk_button_get_event_window(GtkButton * button);
extern gboolean gtk_button_get_focus_on_click(GtkButton * button);
extern GtkWidget *gtk_button_get_image(GtkButton * button);
extern GtkPositionType gtk_button_get_image_position(GtkButton *
button);
extern const char *gtk_button_get_label(GtkButton * button);
extern GtkReliefStyle gtk_button_get_relief(GtkButton * button);
extern GType gtk_button_get_type(void);
extern gboolean gtk_button_get_use_stock(GtkButton * button);
extern gboolean gtk_button_get_use_underline(GtkButton * button);
extern void gtk_button_leave(GtkButton * button);
extern GtkWidget *gtk_button_new(void);
extern GtkWidget *gtk_button_new_from_stock(const char *stock_id);
extern GtkWidget *gtk_button_new_with_label(const char *label);
extern GtkWidget *gtk_button_new_with_mnemonic(const char *label);
extern void gtk_button_pressed(GtkButton * button);
extern void gtk_button_released(GtkButton * button);
extern void gtk_button_set_alignment(GtkButton * button, gfloat,
gfloat);
extern void gtk_button_set_always_show_image(GtkButton * button,
gboolean);
extern void gtk_button_set_focus_on_click(GtkButton * button,
gboolean);
extern void gtk_button_set_image(GtkButton * button, GtkWidget *);
extern void gtk_button_set_image_position(GtkButton * button,
                                GtkPositionType);
extern void gtk_button_set_label(GtkButton * button, const char *);
extern void gtk_button_set_relief(GtkButton * button,
                                GtkReliefStyle);
extern void gtk_button_set_use_stock(GtkButton * button, gboolean);
extern void gtk_button_set_use_underline(GtkButton * button,
gboolean);
extern GType gtk_buttons_type_get_type(void);
extern gboolean gtk_cairo_should_draw_window(cairo_t * cr,
GdkWindow *);
extern void gtk_cairo_transform_to_window(cairo_t * cr, GtkWidget
*,
                                GdkWindow *);
extern void gtk_calendar_clear_marks(GtkCalendar * calendar);
extern GType gtk_calendar_display_options_get_type(void);
extern void gtk_calendar_get_date(GtkCalendar * calendar, guint *,
guint *);
extern gboolean gtk_calendar_get_day_is_marked(GtkCalendar *
calendar,
                                guint);
extern gint gtk_calendar_get_detail_height_rows(GtkCalendar *
calendar);

```

```

extern gint gtk_calendar_get_detail_width_chars(GtkCalendar *
calendar);
extern GtkCalendarDisplayOptions
gtk_calendar_get_display_options(GtkCalendar * calendar);
extern GType gtk_calendar_get_type(void);
extern void gtk_calendar_mark_day(GtkCalendar * calendar, guint);
extern GtkWidget *gtk_calendar_new(void);
extern void gtk_calendar_select_day(GtkCalendar * calendar, guint);
extern void gtk_calendar_select_month(GtkCalendar * calendar, guint,
guint);
extern void gtk_calendar_set_detail_func(GtkCalendar * calendar,
GtkCalendarDetailFunc, gpointer,
GDestroyNotify);
extern void gtk_calendar_set_detail_height_rows(GtkCalendar *
calendar,
gint);
extern void gtk_calendar_set_detail_width_chars(GtkCalendar *
calendar,
gint);
extern void gtk_calendar_set_display_options(GtkCalendar *
calendar,
GtkCalendarDisplayOptions);
extern void gtk_calendar_unmark_day(GtkCalendar * calendar, guint);
extern gboolean gtk_cell_area_activate(GtkCellArea * area,
GtkCellAreaContext *, GtkWidget *,
const GdkRectangle *,
GtkCellRendererState, gboolean);
extern gboolean gtk_cell_area_activate_cell(GtkCellArea * area,
GtkWidget *, GtkCellRenderer
*,
GdkEvent *,
const GdkRectangle *,
GtkCellRendererState);
extern void gtk_cell_area_add(GtkCellArea * area, GtkCellRenderer
*);
extern void gtk_cell_area_add_focus_sibling(GtkCellArea * area,
GtkCellRenderer *,
GtkCellRenderer *);
extern void gtk_cell_area_add_with_properties(GtkCellArea * area,
GtkCellRenderer *,
const char *, ...);
extern void gtk_cell_area_apply_attributes(GtkCellArea * area,
GtkTreeModel *, GtkTreeIter *,
gboolean, gboolean);
extern void gtk_cell_area_attribute_connect(GtkCellArea * area,
GtkCellRenderer *,
const char *, gint);
extern void gtk_cell_area_attribute_disconnect(GtkCellArea * area,
GtkCellRenderer *,
const char *);
extern gint gtk_cell_area_box_get_spacing(GtkCellAreaBox * box);
extern GType gtk_cell_area_box_get_type(void);
extern GtkWidget *gtk_cell_area_box_new(void);
extern void gtk_cell_area_box_pack_end(GtkCellAreaBox * box,
GtkCellRenderer *, gboolean,
gboolean, gboolean);
extern void gtk_cell_area_box_pack_start(GtkCellAreaBox * box,
GtkCellRenderer *, gboolean,
gboolean, gboolean);
extern void gtk_cell_area_box_set_spacing(GtkCellAreaBox * box,
gint);
extern void gtk_cell_area_cell_get(GtkCellArea * area,
GtkCellRenderer *,
const char *, ...);
extern void gtk_cell_area_cell_get_property(GtkCellArea * area,
GtkCellRenderer *,

```

```

                                const char *, GValue *);
extern void gtk_cell_area_cell_get_valist(GtkCellArea * area,
                                const gchar *
                                first_property_name,
                                const char *, va_list var_args);
extern void gtk_cell_area_cell_set(GtkCellArea * area,
GtkCellRenderer *,
                                const char *, ...);
extern void gtk_cell_area_cell_set_property(GtkCellArea * area,
                                GtkCellRenderer *,
                                const char *, const GValue *);
extern void gtk_cell_area_cell_set_valist(GtkCellArea * area,
                                const gchar *
                                first_property_name,
                                const char *, va_list var_args);
extern GParamSpec
*gtk_cell_area_class_find_cell_property(GtkCellAreaClass
                                * aclass,
                                const char *);
extern void
gtk_cell_area_class_install_cell_property(GtkCellAreaClass *
                                aclass, guint,
                                GParamSpec *);
extern GParamSpec
**gtk_cell_area_class_list_cell_properties(GtkCellAreaClass *
aclass,
                                guint *);
extern void gtk_cell_area_context_allocate(GtkCellAreaContext *
context,
                                gint, gint);
extern void
gtk_cell_area_context_get_allocation(GtkCellAreaContext *
                                context, gint *, gint *);
extern
*gtk_cell_area_context_get_area(GtkCellAreaContext *
                                GtkCellArea
                                context);
extern void
gtk_cell_area_context_get_preferred_height(GtkCellAreaContext *
                                context, gint *,
                                gint *);
extern void
gtk_cell_area_context_get_preferred_height_for_width(GtkCellAreaC
ontext *
                                context, gint, gint *,
                                gint *);
extern
gtk_cell_area_context_get_preferred_width(GtkCellAreaContext *
                                context, gint *,
                                gint *);
extern void
gtk_cell_area_context_get_preferred_width_for_height(GtkCellAreaC
ontext *
                                context, gint, gint *,
                                gint *);
extern GType gtk_cell_area_context_get_type(void);
extern void
gtk_cell_area_context_push_preferred_height(GtkCellAreaContext
                                * context, gint,
                                gint);
extern void
gtk_cell_area_context_push_preferred_width(GtkCellAreaContext *
                                context, gint,
                                gint);
extern void
gtk_cell_area_context_reset(GtkCellAreaContext *
context);

```

```

extern gboolean gtk_cell_area_focus(GtkCellArea * area,
GtkDirectionType);
extern void gtk_cell_area_foreach(GtkCellArea * area,
GtkCellCallback,
gpointer);
extern void gtk_cell_area_foreach_alloc(GtkCellArea * area,
GtkCellAreaContext *, GtkWidget *,
const GdkRectangle *,
const GdkRectangle *,
GtkCellAllocCallback, gpointer);
extern void gtk_cell_area_get_cell_allocation(GtkCellArea * area,
GtkCellAreaContext *,
GtkWidget *,
GtkCellRenderer *,
const GdkRectangle *,
GdkRectangle *);
extern
GtkCellRenderer
*gtk_cell_area_get_cell_at_position(GtkCellArea *
area,

GtkCellAreaContext
*, GtkWidget *,
const
GdkRectangle *,
gint, gint,
GdkRectangle *);
extern
const
char
*gtk_cell_area_get_current_path_string(GtkCellArea *
area);
extern GtkCellEditable *gtk_cell_area_get_edit_widget(GtkCellArea
* area);
extern GtkCellRenderer *gtk_cell_area_get_edited_cell(GtkCellArea
* area);
extern GtkCellRenderer *gtk_cell_area_get_focus_cell(GtkCellArea *
area);
extern
GtkCellRenderer
*gtk_cell_area_get_focus_from_sibling(GtkCellArea *
area,

GtkCellRenderer
*);
extern const GList *gtk_cell_area_get_focus_siblings(GtkCellArea *
area,

GtkCellRenderer *);
extern void gtk_cell_area_get_preferred_height(GtkCellArea * area,
GtkCellAreaContext *,

```

```

gint *);
extern void gtk_cell_area_get_preferred_width(GtkCellArea * area,
                                             GtkCellAreaContext *,
                                             GtkWidget *, gint *, gint *);
extern void gtk_cell_area_get_preferred_width_for_height(GtkCellArea *
                                                         area,
                                                         GtkCellAreaContext
                                                         *, GtkWidget *,
                                                         gint, gint *,
                                                         gint *);
extern void gtk_cell_area_get_request_mode(GtkCellArea *
                                           area);
extern GType gtk_cell_area_get_type(void);
extern gboolean gtk_cell_area_has_renderer(GtkCellArea * area,
                                           GtkCellRenderer *);
extern void gtk_cell_area_inner_cell_area(GtkCellArea * area,
                                           GtkWidget *,
                                           const GdkRectangle *,
                                           GdkRectangle *);
extern gboolean gtk_cell_area_is_activatable(GtkCellArea * area);
extern gboolean gtk_cell_area_is_focus_sibling(GtkCellArea * area,
                                              GtkCellRenderer *,
                                              GtkCellRenderer *);
extern void gtk_cell_area_remove(GtkCellArea * area,
                                 GtkCellRenderer *);
extern void gtk_cell_area_remove_focus_sibling(GtkCellArea * area,
                                              GtkCellRenderer *,
                                              GtkCellRenderer *);
extern void gtk_cell_area_render(GtkCellArea * area,
                                 GtkCellAreaContext *,
                                 GtkWidget *, cairo_t *,
                                 const GdkRectangle *,
                                 const GdkRectangle *,
                                 GtkCellRendererState, gboolean);
extern void gtk_cell_area_request_renderer(GtkCellArea * area,
                                           GtkCellRenderer *,
                                           GtkOrientation, GtkWidget *,
                                           gint, gint *, gint *);
extern void gtk_cell_area_set_focus_cell(GtkCellArea * area,
                                         GtkCellRenderer *);
extern void gtk_cell_area_stop_editing(GtkCellArea * area,
                                       gboolean);
extern void gtk_cell_editable_editing_done(GtkCellEditable *
                                           cell_editable);
extern GType gtk_cell_editable_get_type(void);
extern void gtk_cell_editable_remove_widget(GtkCellEditable *
                                           cell_editable);
extern void gtk_cell_editable_start_editing(GtkCellEditable *
                                           cell_editable, GdkEvent *);
extern void gtk_cell_layout_add_attribute(GtkCellLayout *
                                          cell_layout,
                                          GtkCellRenderer *, const char *,
                                          gint);
extern void gtk_cell_layout_clear(GtkCellLayout * cell_layout);
extern void gtk_cell_layout_clear_attributes(GtkCellLayout *
                                             cell_layout,
                                             GtkCellRenderer *);
extern GtkWidget *gtk_cell_layout_get_area(GtkCellLayout *
                                           cell_layout);
extern GLib::List *gtk_cell_layout_get_cells(GtkCellLayout *
                                             cell_layout);
extern GType gtk_cell_layout_get_type(void);
extern void gtk_cell_layout_pack_end(GtkCellLayout * cell_layout,
                                     GtkCellRenderer *, gboolean);

```



```

extern void gtk_cell_layout_pack_start(GtkCellLayout * cell_layout,
                                       GtkCellRenderer *, gboolean);
extern void gtk_cell_layout_reorder(GtkCellLayout * cell_layout,
                                       GtkCellRenderer *, gint);
extern void gtk_cell_layout_set_attributes(GtkCellLayout *
cell_layout,
                                       GtkCellRenderer *, ...);
extern void gtk_cell_layout_set_cell_data_func(GtkCellLayout *
cell_layout,
                                       GtkCellRenderer *,
                                       GtkCellLayoutDataFunc,
                                       gpointer, GDestroyNotify);
extern GType gtk_cell_renderer_accel_get_type(void);
extern GType gtk_cell_renderer_accel_mode_get_type(void);
extern GtkCellRenderer *gtk_cell_renderer_accel_new(void);
extern gboolean gtk_cell_renderer_activate(GtkCellRenderer * cell,
                                       GdkEvent *, GtkWidget *,
                                       const char *,
                                       const GdkRectangle *,
                                       const GdkRectangle *,
                                       GtkCellRendererState);
extern GType gtk_cell_renderer_combo_get_type(void);
extern GtkCellRenderer *gtk_cell_renderer_combo_new(void);
extern void gtk_cell_renderer_get_aligned_area(GtkCellRenderer *
cell,
                                       GtkWidget *,
                                       GtkCellRendererState,
                                       const GdkRectangle *,
                                       GdkRectangle *);
extern void gtk_cell_renderer_get_alignment(GtkCellRenderer * cell,
                                       gfloat *, gfloat *);
extern void gtk_cell_renderer_get_fixed_size(GtkCellRenderer *
cell,
                                       gint *, gint *);
extern void gtk_cell_renderer_get_padding(GtkCellRenderer * cell,
                                       gint *,
                                       gint *);
extern void gtk_cell_renderer_get_preferred_height(GtkCellRenderer
* cell,
                                       GtkWidget *, gint *,
                                       gint *);
extern void
gtk_cell_renderer_get_preferred_height_for_width(GtkCellRenderer *
cell,
                                       GtkWidget *, gint, gint *,
                                       gint *);
extern void gtk_cell_renderer_get_preferred_size(GtkCellRenderer *
cell,
                                       GtkWidget *,
                                       GtkRequisition *,
                                       GtkRequisition *);
extern void gtk_cell_renderer_get_preferred_width(GtkCellRenderer
* cell,
                                       GtkWidget *, gint *,
                                       gint *);
extern void
gtk_cell_renderer_get_preferred_width_for_height(GtkCellRenderer *
cell,
                                       GtkWidget *, gint, gint *,
                                       gint *);
extern GtkSizeRequestMode
gtk_cell_renderer_get_request_mode(GtkCellRenderer * cell);
extern gboolean gtk_cell_renderer_get_sensitive(GtkCellRenderer *
cell);
extern void gtk_cell_renderer_get_size(GtkCellRenderer * cell,
                                       GtkWidget *,

```

```

        const GdkRectangle *, gint *,
        gint *, gint *, gint *);
extern GtkWidget *gtk_cell_renderer_get_state(GtkCellRenderer *
cell,
        GtkWidget *,
        GtkCellRendererState);
extern GType gtk_cell_renderer_get_type(void);
extern gboolean gtk_cell_renderer_get_visible(GtkCellRenderer *
cell);
extern gboolean gtk_cell_renderer_is_activatable(GtkCellRenderer *
cell);
extern GType gtk_cell_renderer_mode_get_type(void);
extern GType gtk_cell_renderer_pixbuf_get_type(void);
extern GtkWidget *gtk_cell_renderer_pixbuf_new(void);
extern GType gtk_cell_renderer_progress_get_type(void);
extern GtkWidget *gtk_cell_renderer_progress_new(void);
extern void gtk_cell_renderer_render(GtkCellRenderer * cell,
cairo_t *,
        GtkWidget *, const GdkRectangle *,
        const GdkRectangle *,
        GtkCellRendererState);
extern void gtk_cell_renderer_set_alignment(GtkCellRenderer * cell,
gfloat,
        gfloat);
extern void gtk_cell_renderer_set_fixed_size(GtkCellRenderer *
cell, gint,
        gint);
extern void gtk_cell_renderer_set_padding(GtkCellRenderer * cell,
gint,
        gint);
extern void gtk_cell_renderer_set_sensitive(GtkCellRenderer * cell,
gboolean);
extern void gtk_cell_renderer_set_visible(GtkCellRenderer * cell,
gboolean);
extern GType gtk_cell_renderer_spin_get_type(void);
extern GtkWidget *gtk_cell_renderer_spin_new(void);
extern GType gtk_cell_renderer_spinner_get_type(void);
extern GtkWidget *gtk_cell_renderer_spinner_new(void);
extern GtkWidget *gtk_cell_renderer_spinner_new(GtkCellEditable
*gtk_cell_renderer_start_editing(GtkCellRenderer *
        cell, GdkEvent *,
        GtkWidget *,
        const char *,
        const GdkRectangle
        *,
        const GdkRectangle
        *,
        GtkCellRendererState);
extern GType gtk_cell_renderer_state_get_type(void);
extern void gtk_cell_renderer_stop_editing(GtkCellRenderer * cell,
gboolean);
extern GType gtk_cell_renderer_text_get_type(void);
extern GtkWidget *gtk_cell_renderer_text_new(void);
extern void
gtk_cell_renderer_text_set_fixed_height_from_font(GtkCellRenderer
Text *
        renderer, gint);
extern gboolean
gtk_cell_renderer_toggle_get_activatable(GtkCellRendererToggle *
toggle);
extern
        gboolean
gtk_cell_renderer_toggle_get_active(GtkCellRendererToggle *
toggle);
extern
        gboolean
gtk_cell_renderer_toggle_get_radio(GtkCellRendererToggle *

```

```

toggle);
extern GType gtk_cell_renderer_toggle_get_type(void);
extern GtkWidget *gtk_cell_renderer_toggle_new(void);
extern void
gtk_cell_renderer_toggle_set_activatable(GtkCellRendererToggle
                                         * toggle, gboolean);
extern void
gtk_cell_renderer_toggle_set_active(GtkCellRendererToggle *
                                     toggle, gboolean);
extern void
gtk_cell_renderer_toggle_set_radio(GtkCellRendererToggle *
                                   toggle, gboolean);
extern GtkWidget *gtk_cell_view_get_displayed_row(GtkCellView *
                                                  cell_view);
extern gboolean gtk_cell_view_get_draw_sensitive(GtkCellView *
                                                  cell_view);
extern gboolean gtk_cell_view_get_fit_model(GtkCellView *
                                             cell_view);
extern GtkTreeModel *gtk_cell_view_get_model(GtkCellView *
                                              cell_view);
extern gboolean gtk_cell_view_get_size_of_row(GtkCellView *
                                              cell_view,
                                              GtkTreePath *
                                              GtkRequisition *);
extern GType gtk_cell_view_get_type(void);
extern GtkWidget *gtk_cell_view_new(void);
extern GtkWidget *gtk_cell_view_new_with_context(GtkCellArea *
                                                  area,
                                                  GtkCellAreaContext *);
extern GtkWidget *gtk_cell_view_new_with_markup(const char
                                                *markup);
extern GtkWidget *gtk_cell_view_new_with_pixbuf(GdkPixbuf *
                                                 pixbuf);
extern GtkWidget *gtk_cell_view_new_with_text(const char *text);
extern void gtk_cell_view_set_background_color(GtkCellView *
                                              cell_view,
                                              const GdkColor *);
extern void gtk_cell_view_set_background_rgba(GtkCellView *
                                              cell_view,
                                              const GdkRGBA *);
extern void gtk_cell_view_set_displayed_row(GtkCellView *
                                             cell_view,
                                             GtkTreePath *);
extern void gtk_cell_view_set_draw_sensitive(GtkCellView *
                                              cell_view,
                                              gboolean);
extern void gtk_cell_view_set_fit_model(GtkCellView * cell_view,
                                         gboolean);
extern void gtk_cell_view_set_model(GtkCellView * cell_view,
                                    GtkTreeModel *);
extern GType gtk_check_button_get_type(void);
extern GtkWidget *gtk_check_button_new(void);
extern GtkWidget *gtk_check_button_new_with_label(const char
                                                  *label);
extern GtkWidget *gtk_check_button_new_with_mnemonic(const char
                                                  *label);
extern gboolean gtk_check_menu_item_get_active(GtkCheckMenuItem *
                                              check_menu_item);
extern gboolean
gtk_check_menu_item_get_draw_as_radio(GtkCheckMenuItem *
                                       check_menu_item);
extern gboolean
gtk_check_menu_item_get_inconsistent(GtkCheckMenuItem *
                                       check_menu_item);
extern GType gtk_check_menu_item_get_type(void);
extern GtkWidget *gtk_check_menu_item_new(void);

```

```

extern GtkWidget *gtk_check_menu_item_new_with_label(const char
*label);
extern GtkWidget *gtk_check_menu_item_new_with_mnemonic(const char
*label);
extern void gtk_check_menu_item_set_active(GtkCheckMenuItem *
check_menu_item, gboolean);
extern void gtk_check_menu_item_set_draw_as_radio(GtkCheckMenuItem
*
check_menu_item,
gboolean);
extern void gtk_check_menu_item_set_inconsistent(GtkCheckMenuItem
*
check_menu_item,
gboolean);
extern void gtk_check_menu_item_toggled(GtkCheckMenuItem *
check_menu_item);
extern const char *gtk_check_version(guint required_major, guint,
guint);
extern void gtk_clipboard_clear(GtkClipboard * clipboard);
extern GtkClipboard *gtk_clipboard_get(GdkAtom selection);
extern GdkDisplay *gtk_clipboard_get_display(GtkClipboard *
clipboard);
extern GtkClipboard *gtk_clipboard_get_for_display(GdkDisplay *
display,
GdkAtom);
extern GObject *gtk_clipboard_get_owner(GtkClipboard * clipboard);
extern GType gtk_clipboard_get_type(void);
extern void gtk_clipboard_request_contents(GtkClipboard *
clipboard,
GdkAtom,
GtkClipboardReceivedFunc,
gpointer);
extern void gtk_clipboard_request_image(GtkClipboard * clipboard,
GtkClipboardImageReceivedFunc,
gpointer);
extern void gtk_clipboard_request_rich_text(GtkClipboard *
clipboard,
GtkTextBuffer *,
GtkClipboardRichTextReceivedFunc,
gpointer);
extern void gtk_clipboard_request_targets(GtkClipboard * clipboard,
GtkClipboardTargetsReceivedFunc,
gpointer);
extern void gtk_clipboard_request_text(GtkClipboard * clipboard,
GtkClipboardTextReceivedFunc,
gpointer);
extern void gtk_clipboard_request_uris(GtkClipboard * clipboard,
GtkClipboardURIRceivedFunc,
gpointer);
extern void gtk_clipboard_set_can_store(GtkClipboard * clipboard,
const GtkTargetEntry *, gint);
extern void gtk_clipboard_set_image(GtkClipboard * clipboard,
GdkPixbuf *);
extern void gtk_clipboard_set_text(GtkClipboard * clipboard, const
char *,
gint);
extern gboolean gtk_clipboard_set_with_data(GtkClipboard *
clipboard,
const GtkTargetEntry *, guint,
GtkClipboardGetFunc,
GtkClipboardClearFunc,
gpointer);
extern gboolean gtk_clipboard_set_with_owner(GtkClipboard *
clipboard,

```

```

const GtkTargetEntry *, guint,
GtkClipboardGetFunc,
GtkClipboardClearFunc,
GObject *);
extern void gtk_clipboard_store(GtkClipboard * clipboard);
extern          GtkSelectionData
*gtk_clipboard_wait_for_contents(GtkClipboard *
                                clipboard,
                                GdkAtom);
extern  GdkPixbuf  *gtk_clipboard_wait_for_image(GtkClipboard *
clipboard);
extern  guint8  *gtk_clipboard_wait_for_rich_text(GtkClipboard *
clipboard,
                                GtkTextBuffer *, GdkAtom *,
                                gsize *);
extern  gboolean  gtk_clipboard_wait_for_targets(GtkClipboard *
clipboard,
                                GdkAtom * *, gint *);
extern  gchar  *gtk_clipboard_wait_for_text(GtkClipboard *
clipboard);
extern  gchar  **gtk_clipboard_wait_for_uris(GtkClipboard *
clipboard);
extern  gboolean  gtk_clipboard_wait_is_image_available(GtkClipboard
*
                                clipboard);
extern
gtk_clipboard_wait_is_rich_text_available(GtkClipboard *
                                clipboard,
                                GtkTextBuffer *);
extern
gtk_clipboard_wait_is_target_available(GtkClipboard *
                                clipboard, GdkAtom);
extern  gboolean  gtk_clipboard_wait_is_text_available(GtkClipboard
*
                                clipboard);
extern  gboolean  gtk_clipboard_wait_is_uris_available(GtkClipboard
*
                                clipboard);
extern  guint16  gtk_color_button_get_alpha(GtkColorButton * button);
extern  void  gtk_color_button_get_color(GtkColorButton * button,
                                GdkColor *);
extern  void  gtk_color_button_get_rgba(GtkColorButton * button,
GdkRGBA *);
extern  const char  *gtk_color_button_get_title(GtkColorButton *
button);
extern  GType  gtk_color_button_get_type(void);
extern  gboolean  gtk_color_button_get_use_alpha(GtkColorButton *
button);
extern  GtkWidget  *gtk_color_button_new(void);
extern  GtkWidget  *gtk_color_button_new_with_color(const GdkColor *
color);
extern  GtkWidget  *gtk_color_button_new_with_rgba(const GdkRGBA *
rgba);
extern  void  gtk_color_button_set_alpha(GtkColorButton * button,
guint16);
extern  void  gtk_color_button_set_color(GtkColorButton * button,
const GdkColor *);
extern  void  gtk_color_button_set_rgba(GtkColorButton * button,
const GdkRGBA *);
extern  void  gtk_color_button_set_title(GtkColorButton * button,
const char *);
extern  void  gtk_color_button_set_use_alpha(GtkColorButton * button,
gboolean);
extern  void  gtk_color_chooser_add_palette(GtkColorChooser *
chooser,
                                GtkOrientation, gint, gint,

```

```

        GdkRGBA *);
extern GType gtk_color_chooser_dialog_get_type(void);
extern GtkWidget *gtk_color_chooser_dialog_new(const char *title,
        GtkWidget *);
extern void gtk_color_chooser_get_rgba(GtkColorChooser * chooser,
        GdkRGBA *);
extern GType gtk_color_chooser_get_type(void);
extern gboolean gtk_color_chooser_get_use_alpha(GtkColorChooser *
chooser);
extern void gtk_color_chooser_set_rgba(GtkColorChooser * chooser,
        const GdkRGBA *);
extern void gtk_color_chooser_set_use_alpha(GtkColorChooser *
chooser,
        gboolean);
extern GType gtk_color_chooser_widget_get_type(void);
extern GtkWidget *gtk_color_chooser_widget_new(void);
extern gint gtk_combo_box_get_active(GtkComboBox * combo_box);
extern const char *gtk_combo_box_get_active_id(GtkComboBox *
combo_box);
extern gboolean gtk_combo_box_get_active_iter(GtkComboBox *
combo_box,
        GtkTreeIter *);
extern gboolean gtk_combo_box_get_add_tearoffs(GtkComboBox *
combo_box);
extern GtkSensitivityType
gtk_combo_box_get_button_sensitivity(GtkComboBox
        *
        combo_box);
extern gint gtk_combo_box_get_column_span_column(GtkComboBox *
combo_box);
extern gint gtk_combo_box_get_entry_text_column(GtkComboBox *
combo_box);
extern gboolean gtk_combo_box_get_focus_on_click(GtkComboBox *
combo_box);
extern gboolean gtk_combo_box_get_has_entry(GtkComboBox *
combo_box);
extern gint gtk_combo_box_get_id_column(GtkComboBox * combo_box);
extern GtkTreeModel *gtk_combo_box_get_model(GtkComboBox *
combo_box);
extern AtkObject *gtk_combo_box_get_popup_accessible(GtkComboBox *
        combo_box);
extern gboolean gtk_combo_box_get_popup_fixed_width(GtkComboBox *
        combo_box);
extern GtkTreeViewRowSeparatorFunc
gtk_combo_box_get_row_separator_func(GtkComboBox * combo_box);
extern gint gtk_combo_box_get_row_span_column(GtkComboBox *
        combo_box);
extern const char *gtk_combo_box_get_title(GtkComboBox * combo_box);
extern GType gtk_combo_box_get_type(void);
extern gint gtk_combo_box_get_wrap_width(GtkComboBox * combo_box);
extern GtkWidget *gtk_combo_box_new(void);
extern GtkWidget *gtk_combo_box_new_with_area(GtkCellArea * area);
extern GtkWidget
*gtk_combo_box_new_with_area_and_entry(GtkCellArea *
        area);
extern GtkWidget *gtk_combo_box_new_with_entry(void);
extern GtkWidget *gtk_combo_box_new_with_model(GtkTreeModel *
model);
extern GtkWidget
*gtk_combo_box_new_with_model_and_entry(GtkTreeModel *
        model);
extern void gtk_combo_box_popdown(GtkComboBox * combo_box);
extern void gtk_combo_box_popup(GtkComboBox * combo_box);
extern void gtk_combo_box_popup_for_device(GtkComboBox * combo_box,
        GdkDevice *);
extern void gtk_combo_box_set_active(GtkComboBox * combo_box, gint);

```

```

extern gboolean gtk_combo_box_set_active_id(GtkComboBox *
combo_box,
                                const char *);
extern void gtk_combo_box_set_active_iter(GtkComboBox * combo_box,
                                GtkTreeIter *);
extern void gtk_combo_box_set_add_tearoffs(GtkComboBox * combo_box,
                                gboolean);
extern void gtk_combo_box_set_button_sensitivity(GtkComboBox *
combo_box,
                                GtkSensitivityType);
extern void gtk_combo_box_set_column_span_column(GtkComboBox *
combo_box,
                                gint);
extern void gtk_combo_box_set_entry_text_column(GtkComboBox *
combo_box,
                                gint);
extern void gtk_combo_box_set_focus_on_click(GtkComboBox * combo,
                                gboolean);
extern void gtk_combo_box_set_id_column(GtkComboBox * combo_box,
                                gint);
extern void gtk_combo_box_set_model(GtkComboBox * combo_box,
                                GtkTreeModel *);
extern void gtk_combo_box_set_popup_fixed_width(GtkComboBox *
combo_box,
                                gboolean);
extern void gtk_combo_box_set_row_separator_func(GtkComboBox *
combo_box,
                                GtkTreeViewRowSeparatorFunc,
                                gpointer, GDestroyNotify);
extern void gtk_combo_box_set_row_span_column(GtkComboBox *
combo_box,
                                gint);
extern void gtk_combo_box_set_title(GtkComboBox * combo_box, const
char *);
extern void gtk_combo_box_set_wrap_width(GtkComboBox * combo_box,
                                gint);
extern void gtk_combo_box_text_append(GtkComboBoxText * combo_box,
                                const char *, const char *);
extern void gtk_combo_box_text_append_text(GtkComboBoxText *
combo_box,
                                const char *);
extern gchar *gtk_combo_box_text_get_active_text(GtkComboBoxText *
combo_box);
extern GType gtk_combo_box_text_get_type(void);
extern void gtk_combo_box_text_insert(GtkComboBoxText * combo_box,
                                gint,
                                const char *, const char *);
extern void gtk_combo_box_text_insert_text(GtkComboBoxText *
combo_box,
                                gint, const char *);
extern GtkWidget *gtk_combo_box_text_new(void);
extern GtkWidget *gtk_combo_box_text_new_with_entry(void);
extern void gtk_combo_box_text_prepend(GtkComboBoxText * combo_box,
                                const char *, const char *);
extern void gtk_combo_box_text_prepend_text(GtkComboBoxText *
combo_box,
                                const char *);
extern void gtk_combo_box_text_remove(GtkComboBoxText * combo_box,
                                gint);
extern void gtk_combo_box_text_remove_all(GtkComboBoxText *
combo_box);
extern void gtk_container_add(GtkContainer * container, GtkWidget
*);
extern void gtk_container_add_with_properties(GtkContainer *
container,

```

```

        GtkWidget *, const char *,
        ...);
extern void gtk_container_check_resize(GtkContainer * container);
extern void gtk_container_child_get(GtkContainer * container,
GtkWidget *,
        const char *, ...);
extern void gtk_container_child_get_property(GtkContainer *
container,
        GtkWidget *, const char *,
        GValue *);
extern void gtk_container_child_get_valist(GtkContainer *
container,
        const gchar *
        first_property_name,
        const char *, va_list
var_args);
extern void gtk_container_child_notify(GtkContainer * container,
        GtkWidget *, const char *);
extern void gtk_container_child_set(GtkContainer * container,
GtkWidget *,
        const char *, ...);
extern void gtk_container_child_set_property(GtkContainer *
container,
        GtkWidget *, const char *,
        const GValue *);
extern void gtk_container_child_set_valist(GtkContainer *
container,
        const gchar *
        first_property_name,
        const char *, va_list
var_args);
extern GType gtk_container_child_type(GtkContainer * container);
extern GParamSpec
*gtk_container_class_find_child_property(GObjectClass *
        cclass,
        const char *);
extern void
gtk_container_class_handle_border_width(GtkContainerClass *
        klass);
extern void
gtk_container_class_install_child_property(GtkContainerClass *
        cclass, guint,
        GParamSpec *);
extern GParamSpec
**gtk_container_class_list_child_properties(GObjectClass
        * cclass,
        guint *);
extern void gtk_container_forall(GtkContainer * container,
GtkCallback,
        gpointer);
extern void gtk_container_foreach(GtkContainer * container,
GtkCallback,
        gpointer);
extern guint gtk_container_get_border_width(GtkContainer *
container);
extern GList *gtk_container_get_children(GtkContainer * container);
extern gboolean gtk_container_get_focus_chain(GtkContainer *
container,
        GList * *);
extern GtkWidget *gtk_container_get_focus_child(GtkContainer *
container);
extern GtkAdjustment
*gtk_container_get_focus_hadjustment(GtkContainer *
        container);
extern GtkAdjustment
*gtk_container_get_focus_vadjustment(GtkContainer *

```



```

        container);
extern                               GtkWidgetPath
*gtk_container_get_path_for_child(GtkContainer *
        container,
        GtkWidget *);
extern GtkResizeMode gtk_container_get_resize_mode(GtkContainer *
        container);
extern GType gtk_container_get_type(void);
extern void gtk_container_propagate_draw(GtkContainer * container,
        GtkWidget *, cairo_t *);
extern void gtk_container_remove(GtkContainer * container,
        GtkWidget *);
extern void gtk_container_resize_children(GtkContainer *
        container);
extern void gtk_container_set_border_width(GtkContainer *
        container,
        guint);
extern void gtk_container_set_focus_chain(GtkContainer * container,
        GList *);
extern void gtk_container_set_focus_child(GtkContainer * container,
        GtkWidget *);
extern void gtk_container_set_focus_hadjustment(GtkContainer *
        container,
        GtkAdjustment *);
extern void gtk_container_set_focus_vadjustment(GtkContainer *
        container,
        GtkAdjustment *);
extern void gtk_container_set_reallocate_redraws(GtkContainer *
        container,
        gboolean);
extern void gtk_container_set_resize_mode(GtkContainer * container,
        GtkResizeMode);
extern void gtk_container_unset_focus_chain(GtkContainer *
        container);
extern GType gtk_corner_type_get_type(void);
extern GType gtk_css_provider_error_get_type(void);
extern GQuark gtk_css_provider_error_quark(void);
extern GtkCssProvider *gtk_css_provider_get_default(void);
extern GtkCssProvider *gtk_css_provider_get_named(const char *name,
        const char *);
extern GType gtk_css_provider_get_type(void);
extern gboolean gtk_css_provider_load_from_data(GtkCssProvider *
        css_provider, const char *,
        gssize, GError * *);
extern gboolean gtk_css_provider_load_from_file(GtkCssProvider *
        css_provider, GFile *,
        GError * *);
extern gboolean gtk_css_provider_load_from_path(GtkCssProvider *
        css_provider, const char *,
        GError * *);
extern GtkCssProvider *gtk_css_provider_new(void);
extern char *gtk_css_provider_to_string(GtkCssProvider * provider);
extern guint gtk_css_section_get_end_line(const GtkCssSection *
        section);
extern guint gtk_css_section_get_end_position(const GtkCssSection
        *
        section);
extern GFile *gtk_css_section_get_file(const GtkCssSection *
        section);
extern GtkCssSection *gtk_css_section_get_parent(const
        GtkCssSection *
        section);
extern GtkCssSectionType gtk_css_section_get_section_type(const
        GtkCssSection *
        section);

```

```

extern guint gtk_css_section_get_start_line(const GtkCssSection *
section);
extern          guint          gtk_css_section_get_start_position(const
GtkCssSection *
                                section);
extern GType gtk_css_section_get_type(void);
extern GtkCssSection *gtk_css_section_ref(GtkCssSection * section);
extern GType gtk_css_section_type_get_type(void);
extern void gtk_css_section_unref(GtkCssSection * section);
extern GType gtk_debug_flag_get_type(void);
extern GType gtk_delete_type_get_type(void);
extern GType gtk_dest_defaults_get_type(void);
extern void gtk_device_grab_add(GtkWidget * widget, GdkDevice *,
gboolean);
extern void gtk_device_grab_remove(GtkWidget * widget, GdkDevice
*);
extern void gtk_dialog_add_action_widget(GtkDialog * dialog,
GtkWidget *,
                                gint);
extern GtkWidget *gtk_dialog_add_button(GtkDialog * dialog, const
char *,
                                gint);
extern void gtk_dialog_add_buttons(GtkDialog * dialog, const char
*, ...);
extern GType gtk_dialog_flags_get_type(void);
extern GtkWidget *gtk_dialog_get_action_area(GtkDialog * dialog);
extern GtkWidget *gtk_dialog_get_content_area(GtkDialog * dialog);
extern gint gtk_dialog_get_response_for_widget(GtkDialog * dialog,
GtkWidget *);
extern GType gtk_dialog_get_type(void);
extern GtkWidget *gtk_dialog_get_widget_for_response(GtkDialog *
dialog,
                                gint);
extern GtkWidget *gtk_dialog_new(void);
extern GtkWidget *gtk_dialog_new_with_buttons(const char *title,
                                GtkWidget *, GtkDialogFlags,
                                const char *, ...);
extern void gtk_dialog_response(GtkDialog * dialog, gint);
extern gint gtk_dialog_run(GtkDialog * dialog);
extern void gtk_dialog_set_alternative_button_order(GtkDialog *
dialog,
                                gint, ...);
extern          void          gtk_dialog_set_alternative_button_order_from_array(GtkDialog *
                                dialog,
                                gint,
                                gint *);
extern void gtk_dialog_set_default_response(GtkDialog * dialog,
gint);
extern void gtk_dialog_set_response_sensitive(GtkDialog * dialog,
gint,
                                gboolean);
extern GType gtk_direction_type_get_type(void);
extern void gtk_disable_setlocale(void);
extern gint gtk_distribute_natural_allocation(gint extra_space,
gint,
                                GtkRequestedSize *);
extern GdkDragContext *gtk_drag_begin(GtkWidget * widget,
                                GtkTargetList * targets,
                                GdkDragAction actions, gint button,
                                GdkEvent * event);
extern gboolean gtk_drag_check_threshold(GtkWidget * widget, gint,
gint,
                                gint, gint);
extern void gtk_drag_dest_add_image_targets(GtkWidget * widget);
extern void gtk_drag_dest_add_text_targets(GtkWidget * widget);

```

```

extern void gtk_drag_dest_add_uri_targets(GtkWidget * widget);
extern GdkAtom gtk_drag_dest_find_target(GtkWidget * widget,
                                         GdkDragContext *,
                                         GtkTargetList *);
extern GtkTargetList *gtk_drag_dest_get_target_list(GtkWidget *
widget);
extern gboolean gtk_drag_dest_get_track_motion(GtkWidget * widget);
extern void gtk_drag_dest_set(GtkWidget * widget, GtkDestDefaults
flags,
                               gint n_targets, GdkDragAction actions);
extern void gtk_drag_dest_set_proxy(GtkWidget * widget,
                                    GdkWindow * proxy_window,
                                    GdkDragProtocol protocol,
                                    gboolean use_coordinates);
extern void gtk_drag_dest_set_target_list(GtkWidget * widget,
                                           GtkTargetList *);
extern void gtk_drag_dest_set_track_motion(GtkWidget * widget,
                                           gboolean);
extern void gtk_drag_dest_unset(GtkWidget * widget);
extern void gtk_drag_finish(GdkDragContext * context, gboolean,
                             gboolean,
                             guint32);
extern void gtk_drag_get_data(GtkWidget * widget, GdkDragContext *
context,
                              GdkAtom target, guint32 time_);
extern GtkWidget *gtk_drag_get_source_widget(GdkDragContext *
context);
extern void gtk_drag_highlight(GtkWidget * widget);
extern GType gtk_drag_result_get_type(void);
extern void gtk_drag_set_icon_default(GdkDragContext * context);
extern void gtk_drag_set_icon_gicon(GdkDragContext * context, GIcon
*,
                                     gint, gint);
extern void gtk_drag_set_icon_name(GdkDragContext * context, const
char *,
                                   gint, gint);
extern void gtk_drag_set_icon_pixbuf(GdkDragContext * context,
                                      GdkPixbuf *,
                                      gint, gint);
extern void gtk_drag_set_icon_stock(GdkDragContext * context, const
char *,
                                    gint, gint);
extern void gtk_drag_set_icon_surface(GdkDragContext * context,
                                       cairo_surface_t *);
extern void gtk_drag_set_icon_widget(GdkDragContext * context,
                                      GtkWidget *,
                                      gint, gint);
extern void gtk_drag_source_add_image_targets(GtkWidget * widget);
extern void gtk_drag_source_add_text_targets(GtkWidget * widget);
extern void gtk_drag_source_add_uri_targets(GtkWidget * widget);
extern GtkTargetList *gtk_drag_source_get_target_list(GtkWidget *
widget);
extern void gtk_drag_source_set(GtkWidget * widget,
                                GdkModifierType start_button_mask,
                                gint n_targets, GdkDragAction actions);
extern void gtk_drag_source_set_icon_gicon(GtkWidget * widget,
                                             GIcon *);
extern void gtk_drag_source_set_icon_name(GtkWidget * widget,
                                           const char *);
extern void gtk_drag_source_set_icon_pixbuf(GtkWidget * widget,
                                             GdkPixbuf *);
extern void gtk_drag_source_set_icon_stock(GtkWidget * widget,
                                           const char *);
extern void gtk_drag_source_set_target_list(GtkWidget * widget,
                                             GtkTargetList *);
extern void gtk_drag_source_unset(GtkWidget * widget);

```

```

extern void gtk_drag_unhighlight(GtkWidget * widget);
extern void gtk_draw_insertion_cursor(GtkWidget * widget, cairo_t
*,
                                const GdkRectangle *, gboolean,
                                GtkTextDirection, gboolean);
extern GType gtk_drawing_area_get_type(void);
extern GtkWidget *gtk_drawing_area_new(void);
extern void gtk_editable_copy_clipboard(GtkEditable * editable);
extern void gtk_editable_cut_clipboard(GtkEditable * editable);
extern void gtk_editable_delete_selection(GtkEditable * editable);
extern void gtk_editable_delete_text(GtkEditable * editable, gint,
gint);
extern gchar *gtk_editable_get_chars(GtkEditable * editable, gint,
gint);
extern gboolean gtk_editable_get_editable(GtkEditable * editable);
extern gint gtk_editable_get_position(GtkEditable * editable);
extern gboolean gtk_editable_get_selection_bounds(GtkEditable *
editable,
                                gint *, gint *);
extern GType gtk_editable_get_type(void);
extern void gtk_editable_insert_text(GtkEditable * editable, const
char *,
                                gint, gint *);
extern void gtk_editable_paste_clipboard(GtkEditable * editable);
extern void gtk_editable_select_region(GtkEditable * editable, gint,
gint);
extern void gtk_editable_set_editable(GtkEditable * editable,
gboolean);
extern void gtk_editable_set_position(GtkEditable * editable, gint);
extern guint gtk_entry_buffer_delete_text(GtkEntryBuffer * buffer,
gint,
gint);
extern void gtk_entry_buffer_emit_deleted_text(GtkEntryBuffer *
buffer,
                                guint, guint);
extern void gtk_entry_buffer_emit_inserted_text(GtkEntryBuffer *
buffer,
                                guint, const char *,
                                guint);
extern gsize gtk_entry_buffer_get_bytes(GtkEntryBuffer * buffer);
extern guint gtk_entry_buffer_get_length(GtkEntryBuffer * buffer);
extern gint gtk_entry_buffer_get_max_length(GtkEntryBuffer *
buffer);
extern const char *gtk_entry_buffer_get_text(GtkEntryBuffer *
buffer);
extern GType gtk_entry_buffer_get_type(void);
extern guint gtk_entry_buffer_insert_text(GtkEntryBuffer * buffer,
gint,
                                const char *, gint);
extern GtkWidget *gtk_entry_buffer_new(const char *initial_chars,
gint);
extern void gtk_entry_buffer_set_max_length(GtkEntryBuffer *
buffer, gint);
extern void gtk_entry_buffer_set_text(GtkEntryBuffer * buffer,
const char *, gint);
extern void gtk_entry_completion_complete(GtkEntryCompletion *
completion);
extern gchar
*gtk_entry_completion_compute_prefix(GtkEntryCompletion *
completion,
const char *);
extern void gtk_entry_completion_delete_action(GtkEntryCompletion
*
completion, gint);
extern const char

```

```

    *gtk_entry_completion_get_completion_prefix(GtkEntryCompletion
*
                                                completion);
extern
                                                GtkWidget
*gtk_entry_completion_get_entry(GtkEntryCompletion *
                                                completion);
extern gboolean
gtk_entry_completion_get_inline_completion(GtkEntryCompletion *
                                                completion);
extern gboolean
gtk_entry_completion_get_inline_selection(GtkEntryCompletion *
completion);
extern
                                                gint
gtk_entry_completion_get_minimum_key_length(GtkEntryCompletion
                                                * completion);
extern
                                                GtkTreeModel
*gtk_entry_completion_get_model(GtkEntryCompletion *
                                                completion);
extern gboolean
gtk_entry_completion_get_popup_completion(GtkEntryCompletion *
completion);
extern
                                                gboolean
gtk_entry_completion_get_popup_set_width(GtkEntryCompletion
                                                * completion);
extern gboolean
gtk_entry_completion_get_popup_single_match(GtkEntryCompletion *
                                                completion);
extern
                                                gint
gtk_entry_completion_get_text_column(GtkEntryCompletion *
                                                completion);
extern GType gtk_entry_completion_get_type(void);
extern
                                                void
gtk_entry_completion_insert_action_markup(GtkEntryCompletion *
                                                completion, gint,
                                                const char *);
extern
                                                void
gtk_entry_completion_insert_action_text(GtkEntryCompletion *
                                                completion, gint,
                                                const char *);
extern void gtk_entry_completion_insert_prefix(GtkEntryCompletion
*
                                                completion);
extern GtkEntryCompletion *gtk_entry_completion_new(void);
extern
                                                GtkEntryCompletion
*gtk_entry_completion_new_with_area(GtkCellArea *
                                                area);
extern
                                                void
gtk_entry_completion_set_inline_completion(GtkEntryCompletion *
                                                completion,
                                                gboolean);
extern
                                                void
gtk_entry_completion_set_inline_selection(GtkEntryCompletion *
                                                completion,
                                                gboolean);
extern void gtk_entry_completion_set_match_func(GtkEntryCompletion
*
                                                completion,

GtkEntryCompletionMatchFunc,
                                                gpointer, GDestroyNotify);
extern
                                                void
gtk_entry_completion_set_minimum_key_length(GtkEntryCompletion
                                                * completion,
                                                gint);
extern void gtk_entry_completion_set_model(GtkEntryCompletion *
completion,

```

```

                                GtkTreeModel *);
extern void
gtk_entry_completion_set_popup_completion(GtkEntryCompletion *
                                           completion,
                                           gboolean);
extern void
gtk_entry_completion_set_popup_set_width(GtkEntryCompletion *
                                           completion, gboolean);
extern void
gtk_entry_completion_set_popup_single_match(GtkEntryCompletion
                                             * completion,
                                             gboolean);
extern void
gtk_entry_completion_set_text_column(GtkEntryCompletion *
                                      completion, gint);
extern gboolean gtk_entry_get_activates_default(GtkEntry * entry);
extern gfloat gtk_entry_get_alignment(GtkEntry * entry);
extern PangoAttrList *gtk_entry_get_attributes(GtkEntry * entry);
extern GtkEntryBuffer *gtk_entry_get_buffer(GtkEntry * entry);
extern GtkEntryCompletion *gtk_entry_get_completion(GtkEntry *
entry);
extern gint gtk_entry_get_current_icon_drag_source(GtkEntry *
entry);
extern GtkAdjustment *gtk_entry_get_cursor_hadjustment(GtkEntry *
entry);
extern gboolean gtk_entry_get_has_frame(GtkEntry * entry);
extern gboolean gtk_entry_get_icon_activatable(GtkEntry * entry,
                                                GtkEntryIconPosition);
extern void gtk_entry_get_icon_area(GtkEntry * entry,
                                    GtkEntryIconPosition,
                                    GdkRectangle *);
extern gint gtk_entry_get_icon_at_pos(GtkEntry * entry, gint, gint);
extern GIcon *gtk_entry_get_icon_gicon(GtkEntry * entry,
                                       GtkEntryIconPosition);
extern const char *gtk_entry_get_icon_name(GtkEntry * entry,
                                           GtkEntryIconPosition);
extern GdkPixbuf *gtk_entry_get_icon_pixbuf(GtkEntry * entry,
                                           GtkEntryIconPosition);
extern gboolean gtk_entry_get_icon_sensitive(GtkEntry * entry,
                                             GtkEntryIconPosition);
extern const char *gtk_entry_get_icon_stock(GtkEntry * entry,
                                             GtkEntryIconPosition);
extern GtkImageType gtk_entry_get_icon_storage_type(GtkEntry *
entry,
                                                     GtkEntryIconPosition);
extern gchar *gtk_entry_get_icon_tooltip_markup(GtkEntry * entry,
                                                GtkEntryIconPosition);
extern gchar *gtk_entry_get_icon_tooltip_text(GtkEntry * entry,
                                              GtkEntryIconPosition);
extern const GtkBorder *gtk_entry_get_inner_border(GtkEntry *
entry);
extern GtkInputHints gtk_entry_get_input_hints(GtkEntry * entry);
extern GtkInputPurpose gtk_entry_get_input_purpose(GtkEntry *
entry);
extern gunichar gtk_entry_get_invisible_char(GtkEntry * entry);
extern PangoLayout *gtk_entry_get_layout(GtkEntry * entry);
extern void gtk_entry_get_layout_offsets(GtkEntry * entry, gint *,
gint *);
extern gint gtk_entry_get_max_length(GtkEntry * entry);
extern gboolean gtk_entry_get_overwrite_mode(GtkEntry * entry);
extern const char *gtk_entry_get_placeholder_text(GtkEntry * entry);
extern gdouble gtk_entry_get_progress_fraction(GtkEntry * entry);
extern gdouble gtk_entry_get_progress_pulse_step(GtkEntry * entry);
extern const char *gtk_entry_get_text(GtkEntry * entry);
extern void gtk_entry_get_text_area(GtkEntry * entry, GdkRectangle
*);

```

```

extern guint16 gtk_entry_get_text_length(GtkEntry * entry);
extern GType gtk_entry_get_type(void);
extern gboolean gtk_entry_get_visibility(GtkEntry * entry);
extern gint gtk_entry_get_width_chars(GtkEntry * entry);
extern GType gtk_entry_icon_position_get_type(void);
extern gboolean gtk_entry_im_context_filter_keypress(GtkEntry *
entry,
GdkEventKey *);
extern gint gtk_entry_layout_index_to_text_index(GtkEntry * entry,
gint);
extern GtkWidget *gtk_entry_new(void);
extern GtkWidget *gtk_entry_new_with_buffer(GtkEntryBuffer *
buffer);
extern void gtk_entry_progress_pulse(GtkEntry * entry);
extern void gtk_entry_reset_im_context(GtkEntry * entry);
extern void gtk_entry_set_activates_default(GtkEntry * entry,
gboolean);
extern void gtk_entry_set_alignment(GtkEntry * entry, gfloat);
extern void gtk_entry_set_attributes(GtkEntry * entry,
PangoAttrList *);
extern void gtk_entry_set_buffer(GtkEntry * entry, GtkEntryBuffer
*);
extern void gtk_entry_set_completion(GtkEntry * entry,
GtkEntryCompletion *);
extern void gtk_entry_set_cursor_hadjustment(GtkEntry * entry,
GtkAdjustment *);
extern void gtk_entry_set_has_frame(GtkEntry * entry, gboolean);
extern void gtk_entry_set_icon_activatable(GtkEntry * entry,
GtkEntryIconPosition,
gboolean);
extern void gtk_entry_set_icon_drag_source(GtkEntry * entry,
GtkEntryIconPosition icon_pos,
GtkTargetList * target_list,
GdkDragAction actions);
extern void gtk_entry_set_icon_from_gicon(GtkEntry * entry,
GtkEntryIconPosition, GIcon *);
extern void gtk_entry_set_icon_from_icon_name(GtkEntry * entry,
GtkEntryIconPosition,
const char *);
extern void gtk_entry_set_icon_from_pixbuf(GtkEntry * entry,
GtkEntryIconPosition,
GdkPixbuf *);
extern void gtk_entry_set_icon_from_stock(GtkEntry * entry,
GtkEntryIconPosition,
const char *);
extern void gtk_entry_set_icon_sensitive(GtkEntry * entry,
GtkEntryIconPosition, gboolean);
extern void gtk_entry_set_icon_tooltip_markup(GtkEntry * entry,
GtkEntryIconPosition,
const char *);
extern void gtk_entry_set_icon_tooltip_text(GtkEntry * entry,
GtkEntryIconPosition,
const char *);
extern void gtk_entry_set_inner_border(GtkEntry * entry,
const GtkBorder *);
extern void gtk_entry_set_input_hints(GtkEntry * entry,
GtkInputHints);
extern void gtk_entry_set_input_purpose(GtkEntry * entry,
GtkInputPurpose);
extern void gtk_entry_set_invisible_char(GtkEntry * entry,
gunichar);
extern void gtk_entry_set_max_length(GtkEntry * entry, gint);
extern void gtk_entry_set_overwrite_mode(GtkEntry * entry,
gboolean);
extern void gtk_entry_set_placeholder_text(GtkEntry * entry, const
char *);

```

```

extern void gtk_entry_set_progress_fraction(GtkEntry * entry,
gdouble);
extern void gtk_entry_set_progress_pulse_step(GtkEntry * entry,
gdouble);
extern void gtk_entry_set_text(GtkEntry * entry, const char *);
extern void gtk_entry_set_visibility(GtkEntry * entry, gboolean);
extern void gtk_entry_set_width_chars(GtkEntry * entry, gint);
extern gint gtk_entry_text_index_to_layout_index(GtkEntry * entry,
gint);
extern void gtk_entry_unset_invisible_char(GtkEntry * entry);
extern gboolean gtk_event_box_get_above_child(GtkEventBox *
event_box);
extern GType gtk_event_box_get_type(void);
extern gboolean gtk_event_box_get_visible_window(GtkEventBox *
event_box);
extern GtkWidget *gtk_event_box_new(void);
extern void gtk_event_box_set_above_child(GtkEventBox * event_box,
gboolean);
extern void gtk_event_box_set_visible_window(GtkEventBox *
event_box,
gboolean);
extern gboolean gtk_events_pending(void);
extern gboolean gtk_expander_get_expanded(GtkExpander * expander);
extern const char *gtk_expander_get_label(GtkExpander * expander);
extern gboolean gtk_expander_get_label_fill(GtkExpander *
expander);
extern GtkWidget *gtk_expander_get_label_widget(GtkExpander *
expander);
extern gboolean gtk_expander_get_resize_toplevel(GtkExpander *
expander);
extern gint gtk_expander_get_spacing(GtkExpander * expander);
extern GType gtk_expander_get_type(void);
extern gboolean gtk_expander_get_use_markup(GtkExpander *
expander);
extern gboolean gtk_expander_get_use_underline(GtkExpander *
expander);
extern GtkWidget *gtk_expander_new(const char *label);
extern GtkWidget *gtk_expander_new_with_mnemonic(const char
*label);
extern void gtk_expander_set_expanded(GtkExpander * expander,
gboolean);
extern void gtk_expander_set_label(GtkExpander * expander, const
char *);
extern void gtk_expander_set_label_fill(GtkExpander * expander,
gboolean);
extern void gtk_expander_set_label_widget(GtkExpander * expander,
GtkWidget *);
extern void gtk_expander_set_resize_toplevel(GtkExpander *
expander,
gboolean);
extern void gtk_expander_set_spacing(GtkExpander * expander, gint);
extern void gtk_expander_set_use_markup(GtkExpander * expander,
gboolean);
extern void gtk_expander_set_use_underline(GtkExpander * expander,
gboolean);
extern GType gtk_expander_style_get_type(void);
extern gboolean gtk_false(void);
extern GType gtk_file_chooser_action_get_type(void);
extern void gtk_file_chooser_add_filter(GtkFileChooser * chooser,
GtkFileFilter *);
extern
gboolean
gtk_file_chooser_add_shortcut_folder(GtkFileChooser *
chooser, const char *,
GError * *);
extern
gboolean
gtk_file_chooser_add_shortcut_folder_uri(GtkFileChooser *

```



```

        chooser,
        const char *,
        GError * *) ;

extern gboolean
gtk_file_chooser_button_get_focus_on_click(GtkFileChooserButton *
button);
extern          const          char
*gtk_file_chooser_button_get_title(GtkFileChooserButton *
button);
extern GType gtk_file_chooser_button_get_type(void);
extern          gint
gtk_file_chooser_button_get_width_chars(GtkFileChooserButton *
button);
extern GtkWidget *gtk_file_chooser_button_new(const char *title,
GtkFileChooserAction);
extern          GtkWidget
*gtk_file_chooser_button_new_with_dialog(GtkWidget *
dialog);
extern          void
gtk_file_chooser_button_set_focus_on_click(GtkFileChooserButton
* button, gboolean);
extern void gtk_file_chooser_button_set_title(GtkFileChooserButton
*
button, const char *);
extern          void
gtk_file_chooser_button_set_width_chars(GtkFileChooserButton *
button, gint);
extern GType gtk_file_chooser_confirmation_get_type(void);
extern GType gtk_file_chooser_dialog_get_type(void);
extern GtkWidget *gtk_file_chooser_dialog_new(const char *title,
GtkWindow *,
GtkFileChooserAction,
const char *, ...);
extern GType gtk_file_chooser_error_get_type(void);
extern GQuark gtk_file_chooser_error_quark(void);
extern          GtkFileChooserAction
gtk_file_chooser_get_action(GtkFileChooser *
chooser);
extern gboolean gtk_file_chooser_get_create_folders(GtkFileChooser
*
chooser);
extern gchar *gtk_file_chooser_get_current_folder(GtkFileChooser *
chooser);
extern          GFile
*gtk_file_chooser_get_current_folder_file(GtkFileChooser *
chooser);
extern          gchar
*gtk_file_chooser_get_current_folder_uri(GtkFileChooser *
chooser);
extern gboolean
gtk_file_chooser_get_do_overwrite_confirmation(GtkFileChooser *
chooser);
extern GtkWidget *gtk_file_chooser_get_extra_widget(GtkFileChooser
*
chooser);
extern GFile *gtk_file_chooser_get_file(GtkFileChooser * chooser);
extern gchar *gtk_file_chooser_get_filename(GtkFileChooser *
chooser);
extern GSList *gtk_file_chooser_get_filenames(GtkFileChooser *
chooser);
extern GSList *gtk_file_chooser_get_files(GtkFileChooser *
chooser);
extern GtkFileFilter *gtk_file_chooser_get_filter(GtkFileChooser *
chooser);
extern gboolean gtk_file_chooser_get_local_only(GtkFileChooser *
chooser);

```

```

extern GFile *gtk_file_chooser_get_preview_file(GtkFileChooser *
chooser);
extern char *gtk_file_chooser_get_preview_filename(GtkFileChooser
*
chooser);
extern char *gtk_file_chooser_get_preview_uri(GtkFileChooser *
chooser);
extern GtkWidget
*gtk_file_chooser_get_preview_widget(GtkFileChooser *
chooser);
extern gboolean
gtk_file_chooser_get_preview_widget_active(GtkFileChooser *
chooser);
extern gboolean
gtk_file_chooser_get_select_multiple(GtkFileChooser *
chooser);
extern gboolean gtk_file_chooser_get_show_hidden(GtkFileChooser *
chooser);
extern GType gtk_file_chooser_get_type(void);
extern gchar *gtk_file_chooser_get_uri(GtkFileChooser * chooser);
extern GSList *gtk_file_chooser_get_uris(GtkFileChooser * chooser);
extern gboolean
gtk_file_chooser_get_use_preview_label(GtkFileChooser *
chooser);
extern GSList *gtk_file_chooser_list_filters(GtkFileChooser *
chooser);
extern GSList
*gtk_file_chooser_list_shortcut_folder_uris(GtkFileChooser *
chooser);
extern GSList
*gtk_file_chooser_list_shortcut_folders(GtkFileChooser *
chooser);
extern void gtk_file_chooser_remove_filter(GtkFileChooser *
chooser,
GtkFileFilter *);
extern gboolean
gtk_file_chooser_remove_shortcut_folder(GtkFileChooser *
chooser,
const char *,
GError * *);
extern gboolean
gtk_file_chooser_remove_shortcut_folder_uri(GtkFileChooser
* chooser,
const char *,
GError * *);
extern void gtk_file_chooser_select_all(GtkFileChooser * chooser);
extern gboolean gtk_file_chooser_select_file(GtkFileChooser *
chooser,
GFile *, GError * *);
extern gboolean gtk_file_chooser_select_filename(GtkFileChooser *
chooser,
const char *);
extern gboolean gtk_file_chooser_select_uri(GtkFileChooser *
chooser,
const char *);
extern void gtk_file_chooser_set_action(GtkFileChooser * chooser,
GtkFileChooserAction);
extern void gtk_file_chooser_set_create_folders(GtkFileChooser *
chooser,
gboolean);
extern gboolean gtk_file_chooser_set_current_folder(GtkFileChooser
*
chooser, const char *);
extern gboolean
gtk_file_chooser_set_current_folder_file(GtkFileChooser *
chooser, GFile *,

```

```

                                GError * *);
extern                                gboolean
gtk_file_chooser_set_current_folder_uri(GtkFileChooser *
                                chooser,
                                const char *);
extern void gtk_file_chooser_set_current_name(GtkFileChooser *
chooser,
                                const char *);
extern                                void
gtk_file_chooser_set_do_overwrite_confirmation(GtkFileChooser *
                                chooser,
                                gboolean);
extern void gtk_file_chooser_set_extra_widget(GtkFileChooser *
chooser,
                                GtkWidget *);
extern gboolean gtk_file_chooser_set_file(GtkFileChooser * chooser,
                                GFile *, GError * *);
extern gboolean gtk_file_chooser_set_filename(GtkFileChooser *
chooser,
                                const char *);
extern void gtk_file_chooser_set_filter(GtkFileChooser * chooser,
                                GtkFileFilter *);
extern void gtk_file_chooser_set_local_only(GtkFileChooser *
chooser,
                                gboolean);
extern void gtk_file_chooser_set_preview_widget(GtkFileChooser *
chooser,
                                GtkWidget *);
extern                                void
gtk_file_chooser_set_preview_widget_active(GtkFileChooser *
                                chooser, gboolean);
extern void gtk_file_chooser_set_select_multiple(GtkFileChooser *
chooser,
                                gboolean);
extern void gtk_file_chooser_set_show_hidden(GtkFileChooser *
chooser,
                                gboolean);
extern gboolean gtk_file_chooser_set_uri(GtkFileChooser * chooser,
                                const char *);
extern void gtk_file_chooser_set_use_preview_label(GtkFileChooser *
*
                                chooser, gboolean);
extern void gtk_file_chooser_unselect_all(GtkFileChooser *
chooser);
extern void gtk_file_chooser_unselect_file(GtkFileChooser *
chooser,
                                GFile *);
extern void gtk_file_chooser_unselect_filename(GtkFileChooser *
chooser,
                                const char *);
extern void gtk_file_chooser_unselect_uri(GtkFileChooser * chooser,
                                const char *);
extern GType gtk_file_chooser_widget_get_type(void);
extern GtkWidget *gtk_file_chooser_widget_new(GtkFileChooserAction
action);
extern void gtk_file_filter_add_custom(GtkFileFilter * filter,
                                GtkFileFilterFlags,
                                GtkFileFilterFunc, gpointer,
                                GDestroyNotify);
extern void gtk_file_filter_add_mime_type(GtkFileFilter * filter,
                                const char *);
extern void gtk_file_filter_add_pattern(GtkFileFilter * filter,
                                const char *);
extern void gtk_file_filter_add_pixbuf_formats(GtkFileFilter *
filter);
extern gboolean gtk_file_filter_filter(GtkFileFilter * filter,

```

```

        const GtkFileFilterInfo *);
extern GType gtk_file_filter_flags_get_type(void);
extern const char *gtk_file_filter_get_name(GtkFileFilter * filter);
extern GtkFileFilterFlags gtk_file_filter_get_needed(GtkFileFilter
*
        filter);
extern GType gtk_file_filter_get_type(void);
extern GtkFileFilter *gtk_file_filter_new(void);
extern void gtk_file_filter_set_name(GtkFileFilter * filter, const
char *);
extern GType gtk_fixed_get_type(void);
extern void gtk_fixed_move(GtkFixed * fixed, GtkWidget *, gint,
gint);
extern GtkWidget *gtk_fixed_new(void);
extern void gtk_fixed_put(GtkFixed * fixed, GtkWidget *, gint,
gint);
extern const char *gtk_font_button_get_font_name(GtkFontButton *
font_button);
extern gboolean gtk_font_button_get_show_size(GtkFontButton *
font_button);
extern gboolean gtk_font_button_get_show_style(GtkFontButton *
font_button);
extern const char *gtk_font_button_get_title(GtkFontButton *
font_button);
extern GType gtk_font_button_get_type(void);
extern gboolean gtk_font_button_get_use_font(GtkFontButton *
font_button);
extern gboolean gtk_font_button_get_use_size(GtkFontButton *
font_button);
extern GtkWidget *gtk_font_button_new(void);
extern GtkWidget *gtk_font_button_new_with_font(const char
*fontname);
extern gboolean gtk_font_button_set_font_name(GtkFontButton *
font_button,
        const char *);
extern void gtk_font_button_set_show_size(GtkFontButton *
font_button,
        gboolean);
extern void gtk_font_button_set_show_style(GtkFontButton *
font_button,
        gboolean);
extern void gtk_font_button_set_title(GtkFontButton * font_button,
const char *);
extern void gtk_font_button_set_use_font(GtkFontButton *
font_button,
        gboolean);
extern void gtk_font_button_set_use_size(GtkFontButton *
font_button,
        gboolean);
extern GType gtk_font_chooser_dialog_get_type(void);
extern GtkWidget *gtk_font_chooser_dialog_new(const char *title,
        GtkWindow *);
extern gchar *gtk_font_chooser_get_font(GtkFontChooser *
fontchooser);
extern
        PangoFontDescription
*gtk_font_chooser_get_font_desc(GtkFontChooser
        * fontchooser);
extern
        PangoFontFace
*gtk_font_chooser_get_font_face(GtkFontChooser *
        fontchooser);
extern
        PangoFontFamily
*gtk_font_chooser_get_font_family(GtkFontChooser *
        fontchooser);
extern gint gtk_font_chooser_get_font_size(GtkFontChooser *
fontchooser);
extern gchar *gtk_font_chooser_get_preview_text(GtkFontChooser *

```

```

fontchooser);

extern                               gboolean
gtk_font_chooser_get_show_preview_entry(GtkFontChooser *
                                       fontchooser);

extern GType gtk_font_chooser_get_type(void);
extern void  gtk_font_chooser_set_filter_func(GtkFontChooser *
fontchooser,
                                             GtkFontFilterFunc, gpointer,
                                             GDestroyNotify);
extern void gtk_font_chooser_set_font(GtkFontChooser * fontchooser,
                                     const char *);
extern void  gtk_font_chooser_set_font_desc(GtkFontChooser *
fontchooser,
                                             const PangoFontDescription *);
extern void  gtk_font_chooser_set_preview_text(GtkFontChooser *
fontchooser,
                                             const char *);
extern void gtk_font_chooser_set_show_preview_entry(GtkFontChooser
*
                                             fontchooser, gboolean);
extern GType gtk_font_chooser_widget_get_type(void);
extern GtkWidget *gtk_font_chooser_widget_new(void);
extern const char *gtk_frame_get_label(GtkFrame * frame);
extern void gtk_frame_get_label_align(GtkFrame * frame, gfloat *,
                                     gfloat *);
extern GtkWidget *gtk_frame_get_label_widget(GtkFrame * frame);
extern GtkShadowType gtk_frame_get_shadow_type(GtkFrame * frame);
extern GType gtk_frame_get_type(void);
extern GtkWidget *gtk_frame_new(const char *label);
extern void gtk_frame_set_label(GtkFrame * frame, const char *);
extern void gtk_frame_set_label_align(GtkFrame * frame, gfloat,
                                     gfloat);
extern void gtk_frame_set_label_widget(GtkFrame * frame, GtkWidget
*);
extern void  gtk_frame_set_shadow_type(GtkFrame * frame,
                                     GtkShadowType);
extern guint gtk_get_binary_age(void);
extern GdkEvent *gtk_get_current_event(void);
extern GdkDevice *gtk_get_current_event_device(void);
extern gboolean  gtk_get_current_event_state(GdkModifierType *
state);
extern guint32 gtk_get_current_event_time(void);
extern guint gtk_get_debug_flags(void);
extern PangoLanguage *gtk_get_default_language(void);
extern GtkWidget *gtk_get_event_widget(GdkEvent * event);
extern guint gtk_get_interface_age(void);
extern guint gtk_get_major_version(void);
extern guint gtk_get_micro_version(void);
extern guint gtk_get_minor_version(void);
extern GOptionGroup *gtk_get_option_group(gboolean
open_default_display);
extern void gtk_grab_add(GtkWidget * widget);
extern GtkWidget *gtk_grab_get_current(void);
extern void gtk_grab_remove(GtkWidget * widget);
extern void  gtk_gradient_add_color_stop(GtkGradient * gradient,
gdoube,
                                     GtkSymbolicColor *);
extern GType gtk_gradient_get_type(void);
extern GtkGradient *gtk_gradient_new_linear(gdouble x0, gdouble,
gdoube,
                                     gdouble);
extern GtkGradient *gtk_gradient_new_radial(gdouble x0, gdouble,
gdoube,
                                     gdouble, gdouble, gdouble);
extern GtkGradient *gtk_gradient_ref(GtkGradient * gradient);
extern gboolean gtk_gradient_resolve(GtkGradient * gradient,

```

```

                                GtkStyleProperties *,
                                cairo_pattern_t * *);
extern                                cairo_pattern_t
*gtk_gradient_resolve_for_context(GtkGradient *
                                gradient,
                                GtkStyleContext
                                *);
extern char *gtk_gradient_to_string(GtkGradient * gradient);
extern void gtk_gradient_unref(GtkGradient * gradient);
extern void gtk_grid_attach(GtkGrid * grid, GtkWidget *, gint, gint,
gint,
                                gint);
extern void gtk_grid_attach_next_to(GtkGrid * grid, GtkWidget *,
                                GtkWidget *, GtkPositionType, gint,
                                gint);
extern GtkWidget *gtk_grid_get_child_at(GtkGrid * grid, gint, gint);
extern gboolean gtk_grid_get_column_homogeneous(GtkGrid * grid);
extern guint gtk_grid_get_column_spacing(GtkGrid * grid);
extern gboolean gtk_grid_get_row_homogeneous(GtkGrid * grid);
extern guint gtk_grid_get_row_spacing(GtkGrid * grid);
extern GType gtk_grid_get_type(void);
extern void gtk_grid_insert_column(GtkGrid * grid, gint);
extern void gtk_grid_insert_next_to(GtkGrid * grid, GtkWidget *,
                                GtkPositionType);
extern void gtk_grid_insert_row(GtkGrid * grid, gint);
extern GtkWidget *gtk_grid_new(void);
extern void gtk_grid_set_column_homogeneous(GtkGrid * grid,
gboolean);
extern void gtk_grid_set_column_spacing(GtkGrid * grid, guint);
extern void gtk_grid_set_row_homogeneous(GtkGrid * grid, gboolean);
extern void gtk_grid_set_row_spacing(GtkGrid * grid, guint);
extern void gtk_hsv_to_rgb(gdouble, gdouble, gdouble, gdouble *,
                                gdouble *, gdouble *);
extern void gtk_icon_factory_add(GtkIconFactory * factory, const
char *,
                                GtkIconSet *);
extern void gtk_icon_factory_add_default(GtkIconFactory * factory);
extern GType gtk_icon_factory_get_type(void);
extern GtkIconSet *gtk_icon_factory_lookup(GtkIconFactory *
factory,
                                const char *);
extern GtkIconSet *gtk_icon_factory_lookup_default(const char
*stock_id);
extern GtkIconFactory *gtk_icon_factory_new(void);
extern void gtk_icon_factory_remove_default(GtkIconFactory *
factory);
extern GtkIconInfo *gtk_icon_info_copy(GtkIconInfo * icon_info);
extern void gtk_icon_info_free(GtkIconInfo * icon_info);
extern gboolean gtk_icon_info_get_attach_points(GtkIconInfo *
icon_info,
                                GdkPoint * *, gint *);
extern gint gtk_icon_info_get_base_size(GtkIconInfo * icon_info);
extern GdkPixbuf *gtk_icon_info_get_builtin_pixbuf(GtkIconInfo *
icon_info);
extern const char *gtk_icon_info_get_display_name(GtkIconInfo *
icon_info);
extern gboolean gtk_icon_info_get_embedded_rect(GtkIconInfo *
icon_info,
                                GdkRectangle *);
extern const char *gtk_icon_info_get_filename(GtkIconInfo *
icon_info);
extern GType gtk_icon_info_get_type(void);
extern GdkPixbuf *gtk_icon_info_load_icon(GtkIconInfo * icon_info,
                                GError * *);
extern GdkPixbuf *gtk_icon_info_load_symbolic(GtkIconInfo *
icon_info,

```

```

const GdkRGBA *,
const GdkRGBA *,
const GdkRGBA *,
const GdkRGBA *, gboolean *,
GError * *);

extern                                     GdkPixbuf
*gtk_icon_info_load_symbolic_for_context(GtkIconInfo *
                                         icon_info,
                                         GtkStyleContext
                                         *, gboolean *,
                                         GError * *);

extern                                     GdkPixbuf
*gtk_icon_info_load_symbolic_for_style(GtkIconInfo *
                                         icon_info,
                                         GtkStyle *,
                                         GtkStateType,
                                         gboolean *,
                                         GError * *);

extern GtkIconInfo *gtk_icon_info_new_for_pixbuf(GtkIconTheme *
icon_theme,
                                                  GdkPixbuf *);

extern void gtk_icon_info_set_raw_coordinates(GtkIconInfo *
icon_info,
                                              gboolean);

extern GType gtk_icon_lookup_flags_get_type(void);
extern void gtk_icon_set_add_source(GtkIconSet * icon_set,
                                     const GtkIconSource *);
extern GtkIconSet *gtk_icon_set_copy(GtkIconSet * icon_set);
extern void gtk_icon_set_get_sizes(GtkIconSet * icon_set,
GtkIconSize * *,
                                gint *);
extern GType gtk_icon_set_get_type(void);
extern GtkIconSet *gtk_icon_set_new(void);
extern GtkIconSet *gtk_icon_set_new_from_pixbuf(GdkPixbuf *
pixbuf);
extern GtkIconSet *gtk_icon_set_ref(GtkIconSet * icon_set);
extern GdkPixbuf *gtk_icon_set_render_icon(GtkIconSet * icon_set,
                                           GtkStyle *, GtkTextDirection,
                                           GtkStateType, GtkIconSize,
                                           GtkWidget *, const char *);
extern GdkPixbuf *gtk_icon_set_render_icon_pixbuf(GtkIconSet *
icon_set,
                                                  GtkStyleContext *,
                                                  GtkIconSize);
extern void gtk_icon_set_unref(GtkIconSet * icon_set);
extern GtkIconSize gtk_icon_size_from_name(const char *name);
extern const char *gtk_icon_size_get_name(GtkIconSize size);
extern GType gtk_icon_size_get_type(void);
extern gboolean gtk_icon_size_lookup(GtkIconSize size, gint *, gint
*);
extern gboolean gtk_icon_size_lookup_for_settings(GtkSettings *
settings,
                                                  GtkIconSize, gint *,
                                                  gint *);
extern GtkIconSize gtk_icon_size_register(const char *name, gint,
gint);
extern void gtk_icon_size_register_alias(const char *alias,
GtkIconSize);
extern GtkIconSource *gtk_icon_source_copy(const GtkIconSource *
source);
extern void gtk_icon_source_free(GtkIconSource * source);
extern GtkTextDirection gtk_icon_source_get_direction(const
GtkIconSource *
source);
extern gboolean gtk_icon_source_get_direction_wildcarded(const
GtkIconSource *

```

```

        source);
extern      const      char      *gtk_icon_source_get_filename(const
GtkIconSource *
        source);
extern      const      char      *gtk_icon_source_get_icon_name(const
GtkIconSource *
        source);
extern GdkPixbuf *gtk_icon_source_get_pixbuf(const GtkIconSource *
source);
extern GtkIconSize gtk_icon_source_get_size(const GtkIconSource *
source);
extern      gboolean      gtk_icon_source_get_size_wildcarded(const
GtkIconSource *
        source);
extern GtkStateType gtk_icon_source_get_state(const GtkIconSource
*
        source);
extern      gboolean      gtk_icon_source_get_state_wildcarded(const
GtkIconSource *
        source);
extern GType gtk_icon_source_get_type(void);
extern GtkIconSource *gtk_icon_source_new(void);
extern void gtk_icon_source_set_direction(GtkIconSource * source,
        GtkTextDirection);
extern void gtk_icon_source_set_direction_wildcarded(GtkIconSource
*
        source, gboolean);
extern void gtk_icon_source_set_filename(GtkIconSource * source,
        const char *);
extern void gtk_icon_source_set_icon_name(GtkIconSource * source,
        const char *);
extern void gtk_icon_source_set_pixbuf(GtkIconSource * source,
        GdkPixbuf *);
extern void gtk_icon_source_set_size(GtkIconSource * source,
        GtkIconSize);
extern void gtk_icon_source_set_size_wildcarded(GtkIconSource *
source,
        gboolean);
extern void gtk_icon_source_set_state(GtkIconSource * source,
        GtkStateType);
extern void gtk_icon_source_set_state_wildcarded(GtkIconSource *
source,
        gboolean);
extern void gtk_icon_theme_add_built_in_icon(const char *icon_name,
        gint,
        GdkPixbuf *);
extern void gtk_icon_theme_append_search_path(GtkIconTheme *
icon_theme,
        const char *);
extern GtkIconInfo *gtk_icon_theme_choose_icon(GtkIconTheme *
icon_theme,
        const char **, gint,
        GtkIconLookupFlags);
extern GType gtk_icon_theme_error_get_type(void);
extern GQuark gtk_icon_theme_error_quark(void);
extern GtkIconTheme *gtk_icon_theme_get_default(void);
extern char *gtk_icon_theme_get_example_icon_name(GtkIconTheme *
        icon_theme);
extern GtkIconTheme *gtk_icon_theme_get_for_screen(GdkScreen *
screen);
extern      gint      *gtk_icon_theme_get_icon_sizes(GtkIconTheme *
icon_theme,
        const char *);
extern void gtk_icon_theme_get_search_path(GtkIconTheme *
icon_theme,
        gchar * **, gint *);

```



```

extern GType gtk_icon_theme_get_type(void);
extern gboolean gtk_icon_theme_has_icon(GtkIconTheme * icon_theme,
                                         const char *);
extern GList *gtk_icon_theme_list_contexts(GtkIconTheme *
icon_theme);
extern GList *gtk_icon_theme_list_icons(GtkIconTheme * icon_theme,
                                         const char *);
extern GdkPixbuf *gtk_icon_theme_load_icon(GtkIconTheme *
icon_theme,
                                         const char *, gint,
                                         GtkIconLookupFlags, GError *
*);
extern GtkIconInfo *gtk_icon_theme_lookup_by_gicon(GtkIconTheme *
icon_theme, GIcon *,
gint,
GtkIconLookupFlags);
extern GtkIconInfo *gtk_icon_theme_lookup_icon(GtkIconTheme *
icon_theme,
                                         const char *, gint,
                                         GtkIconLookupFlags);
extern GtkIconTheme *gtk_icon_theme_new(void);
extern void gtk_icon_theme_prepend_search_path(GtkIconTheme *
icon_theme,
                                         const char *);
extern gboolean gtk_icon_theme_rescan_if_needed(GtkIconTheme *
icon_theme);
extern void gtk_icon_theme_set_custom_theme(GtkIconTheme *
icon_theme,
                                         const char *);
extern void gtk_icon_theme_set_screen(GtkIconTheme * icon_theme,
GdkScreen *);
extern void gtk_icon_theme_set_search_path(GtkIconTheme *
icon_theme,
                                         const char **, gint);
extern void
gtk_icon_view_convert_widget_to_bin_window_coords(GtkIconView *
icon_view,
gint, gint,
gint *,
gint *);
extern cairo_surface_t *gtk_icon_view_create_drag_icon(GtkIconView *
icon_view,
GtkTreePath *);
extern GType gtk_icon_view_drop_position_get_type(void);
extern void gtk_icon_view_enable_model_drag_dest(GtkIconView *
icon_view,
gint n_targets,
GdkDragAction actions);
extern void gtk_icon_view_enable_model_drag_source(GtkIconView *
icon_view,
GdkModifierType
start_button_mask,
gint n_targets,
GdkDragAction actions);
extern gboolean gtk_icon_view_get_cell_rect(GtkIconView *
icon_view,
GtkTreePath *,
GtkCellRenderer *,
GdkRectangle *);
extern gint gtk_icon_view_get_column_spacing(GtkIconView *
icon_view);
extern gint gtk_icon_view_get_columns(GtkIconView * icon_view);
extern gboolean gtk_icon_view_get_cursor(GtkIconView * icon_view,
GtkTreePath * *,
GtkCellRenderer * *);

```

```

extern gboolean gtk_icon_view_get_dest_item_at_pos(GtkIconView *
icon_view,
                                                    gint, gint,
                                                    GtkTreePath * *,
                                                    GtkIconViewDropPosition
                                                    *);
extern void      gtk_icon_view_get_drag_dest_item(GtkIconView *
icon_view,
                                                    GtkTreePath * *,
                                                    GtkIconViewDropPosition *);
extern gboolean  gtk_icon_view_get_item_at_pos(GtkIconView *
icon_view,
                                                    gint, gint, GtkTreePath * *,
                                                    GtkCellRenderer * *);
extern gint      gtk_icon_view_get_item_column(GtkIconView * icon_view,
                                                    GtkTreePath *);
extern
                                                    GtkOrientation
gtk_icon_view_get_item_orientation(GtkIconView *
icon_view);
extern gint      gtk_icon_view_get_item_padding(GtkIconView *
icon_view);
extern gint      gtk_icon_view_get_item_row(GtkIconView * icon_view,
                                                    GtkTreePath *);
extern gint      gtk_icon_view_get_item_width(GtkIconView * icon_view);
extern gint      gtk_icon_view_get_margin(GtkIconView * icon_view);
extern gint      gtk_icon_view_get_markup_column(GtkIconView *
icon_view);
extern GtkTreeModel *gtk_icon_view_get_model(GtkIconView *
icon_view);
extern GtkTreePath *gtk_icon_view_get_path_at_pos(GtkIconView *
icon_view,
                                                    gint, gint);
extern gint      gtk_icon_view_get_pixbuf_column(GtkIconView *
icon_view);
extern gboolean  gtk_icon_view_get_reorderable(GtkIconView *
icon_view);
extern gint      gtk_icon_view_get_row_spacing(GtkIconView * icon_view);
extern GList     *gtk_icon_view_get_selected_items(GtkIconView *
icon_view);
extern
                                                    GtkSelectionMode
gtk_icon_view_get_selection_mode(GtkIconView *
icon_view);
extern gint      gtk_icon_view_get_spacing(GtkIconView * icon_view);
extern gint      gtk_icon_view_get_text_column(GtkIconView * icon_view);
extern gint      gtk_icon_view_get_tooltip_column(GtkIconView *
icon_view);
extern gboolean  gtk_icon_view_get_tooltip_context(GtkIconView *
icon_view,
                                                    gint *, gint *, gboolean,
                                                    GtkTreeModel * *,
                                                    GtkTreePath * *,
                                                    GtkTreeIter *);
extern GType      gtk_icon_view_get_type(void);
extern gboolean  gtk_icon_view_get_visible_range(GtkIconView *
icon_view,
                                                    GtkTreePath * *,
                                                    GtkTreePath * *);
extern void      gtk_icon_view_item_activated(GtkIconView * icon_view,
                                                    GtkTreePath *);
extern GtkWidget *gtk_icon_view_new(void);
extern GtkWidget *gtk_icon_view_new_with_area(GtkCellArea * area);
extern GtkWidget *gtk_icon_view_new_with_model(GtkTreeModel *
model);
extern gboolean  gtk_icon_view_path_is_selected(GtkIconView *
icon_view,
                                                    GtkTreePath *);

```

```

extern void gtk_icon_view_scroll_to_path(GtkIconView * icon_view,
                                         GtkTreePath *, gboolean, gfloat,
                                         gfloat);
extern void gtk_icon_view_select_all(GtkIconView * icon_view);
extern void gtk_icon_view_select_path(GtkIconView * icon_view,
                                       GtkTreePath *);
extern void gtk_icon_view_selected_foreach(GtkIconView * icon_view,
                                           GtkIconViewForeachFunc,
                                           gpointer);
extern void gtk_icon_view_set_column_spacing(GtkIconView *
icon_view,
                                           gint);
extern void gtk_icon_view_set_columns(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_cursor(GtkIconView * icon_view,
                                     GtkTreePath *, GtkCellRenderer *,
                                     gboolean);
extern void gtk_icon_view_set_drag_dest_item(GtkIconView *
icon_view,
                                           GtkTreePath *,
                                           GtkIconViewDropPosition);
extern void gtk_icon_view_set_item_orientation(GtkIconView *
icon_view,
                                           GtkOrientation);
extern void gtk_icon_view_set_item_padding(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_item_width(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_margin(GtkIconView * icon_view, gint);
extern void gtk_icon_view_set_markup_column(GtkIconView *
icon_view, gint);
extern void gtk_icon_view_set_model(GtkIconView * icon_view,
                                    GtkTreeModel *);
extern void gtk_icon_view_set_pixbuf_column(GtkIconView *
icon_view, gint);
extern void gtk_icon_view_set_reorderable(GtkIconView * icon_view,
gboolean);
extern void gtk_icon_view_set_row_spacing(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_selection_mode(GtkIconView *
icon_view,
                                           GtkSelectionMode);
extern void gtk_icon_view_set_spacing(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_text_column(GtkIconView * icon_view,
gint);
extern void gtk_icon_view_set_tooltip_cell(GtkIconView * icon_view,
                                           GtkTooltip *, GtkTreePath *,
                                           GtkCellRenderer *);
extern void gtk_icon_view_set_tooltip_column(GtkIconView *
icon_view,
                                           gint);
extern void gtk_icon_view_set_tooltip_item(GtkIconView * icon_view,
                                           GtkTooltip *, GtkTreePath *);
extern void gtk_icon_view_unselect_all(GtkIconView * icon_view);
extern void gtk_icon_view_unselect_path(GtkIconView * icon_view,
                                       GtkTreePath *);
extern void gtk_icon_view_unset_model_drag_dest(GtkIconView *
icon_view);
extern void gtk_icon_view_unset_model_drag_source(GtkIconView *
icon_view);
extern gboolean gtk_im_context_delete_surrounding(GtkIMContext *
context,
                                           gint, gint);
extern gboolean gtk_im_context_filter_keypress(GtkIMContext *
context,

```

```

                                GdkEventKey *);
extern void gtk_im_context_focus_in(GtkIMContext * context);
extern void gtk_im_context_focus_out(GtkIMContext * context);
extern void gtk_im_context_get_preedit_string(GtkIMContext *
context,
                                gchar * *, PangoAttrList * *,
                                gint *);
extern gboolean gtk_im_context_get_surrounding(GtkIMContext *
context,
                                gchar * *, gint *);
extern GType gtk_im_context_get_type(void);
extern void gtk_im_context_reset(GtkIMContext * context);
extern void gtk_im_context_set_client_window(GtkIMContext *
context,
                                GdkWindow *);
extern void gtk_im_context_set_cursor_location(GtkIMContext *
context,
                                const GdkRectangle *);
extern void gtk_im_context_set_surrounding(GtkIMContext * context,
                                const char *, gint, gint);
extern void gtk_im_context_set_use_preedit(GtkIMContext * context,
                                gboolean);
extern void gtk_im_context_simple_add_table(GtkIMContextSimple *
context_simple, guint16 *,
                                gint, gint);
extern GType gtk_im_context_simple_get_type(void);
extern GtkIMContext *gtk_im_context_simple_new(void);
extern void gtk_im_multicontext_append_menuitems(GtkIMMulticontext *
context, GtkMenuShell *);
extern const char
*gtk_im_multicontext_get_context_id(GtkIMMulticontext *
context);
extern GType gtk_im_multicontext_get_type(void);
extern GtkIMContext *gtk_im_multicontext_new(void);
extern void gtk_im_multicontext_set_context_id(GtkIMMulticontext *
context,
                                const char *);
extern GType gtk_im_preedit_style_get_type(void);
extern GType gtk_im_status_style_get_type(void);
extern void gtk_image_clear(GtkImage * image);
extern GdkPixbufAnimation *gtk_image_get_animation(GtkImage *
image);
extern void gtk_image_get_gicon(GtkImage * image, GIcon * *,
                                GtkIconSize *);
extern void gtk_image_get_icon_name(GtkImage * image, const char
**,
                                GtkIconSize *);
extern void gtk_image_get_icon_set(GtkImage * image, GtkIconSet *
*,
                                GtkIconSize *);
extern GdkPixbuf *gtk_image_get_pixbuf(GtkImage * image);
extern gint gtk_image_get_pixel_size(GtkImage * image);
extern void gtk_image_get_stock(GtkImage * image, gchar * *,
                                GtkIconSize *);
extern GtkImageType gtk_image_get_storage_type(GtkImage * image);
extern GType gtk_image_get_type(void);
extern gboolean
gtk_image_menu_item_get_always_show_image(GtkImageMenuItem *
image_menu_item);
extern GtkWidget *gtk_image_menu_item_get_image(GtkImageMenuItem *
image_menu_item);
extern GType gtk_image_menu_item_get_type(void);
extern gboolean gtk_image_menu_item_get_use_stock(GtkImageMenuItem *

```

```

        image_menu_item);
extern GtkWidget *gtk_image_menu_item_new(void);
extern GtkWidget *gtk_image_menu_item_new_from_stock(const char
*stock_id,
        GtkAccelGroup *);
extern GtkWidget *gtk_image_menu_item_new_with_label(const char
*label);
extern GtkWidget *gtk_image_menu_item_new_with_mnemonic(const char
*label);
extern void gtk_image_menu_item_set_accel_group(GtkImageMenuItem *
        image_menu_item,
        GtkAccelGroup *);
extern
        void
gtk_image_menu_item_set_always_show_image(GtkImageMenuItem *
        image_menu_item,
        gboolean);
extern void gtk_image_menu_item_set_image(GtkImageMenuItem *
        image_menu_item, GtkWidget *);
extern void gtk_image_menu_item_set_use_stock(GtkImageMenuItem *
        image_menu_item, gboolean);
extern GtkWidget *gtk_image_new(void);
extern GtkWidget *gtk_image_new_from_animation(GdkPixbufAnimation
*
        animation);
extern GtkWidget *gtk_image_new_from_file(const char *filename);
extern GtkWidget *gtk_image_new_from_gicon(GIcon * icon,
GtkIconSize);
extern GtkWidget *gtk_image_new_from_icon_name(const char
*icon_name,
        GtkIconSize);
extern GtkWidget *gtk_image_new_from_icon_set(GtkIconSet *
icon_set,
        GtkIconSize);
extern GtkWidget *gtk_image_new_from_pixbuf(GdkPixbuf * pixbuf);
extern GtkWidget *gtk_image_new_from_resource(const char
*resource_path);
extern GtkWidget *gtk_image_new_from_stock(const char *stock_id,
        GtkIconSize);
extern void gtk_image_set_from_animation(GtkImage * image,
        GdkPixbufAnimation *);
extern void gtk_image_set_from_file(GtkImage * image, const char
*);
extern void gtk_image_set_from_gicon(GtkImage * image, GIcon *,
        GtkIconSize);
extern void gtk_image_set_from_icon_name(GtkImage * image, const
char *,
        GtkIconSize);
extern void gtk_image_set_from_icon_set(GtkImage * image,
GtkIconSet *,
        GtkIconSize);
extern void gtk_image_set_from_pixbuf(GtkImage * image, GdkPixbuf
*);
extern void gtk_image_set_from_resource(GtkImage * image, const
char *);
extern void gtk_image_set_from_stock(GtkImage * image, const char
*,
        GtkIconSize);
extern void gtk_image_set_pixel_size(GtkImage * image, gint);
extern GType gtk_image_type_get_type(void);
extern void gtk_info_bar_add_action_widget(GtkInfoBar * info_bar,
        GtkWidget *, gint);
extern GtkWidget *gtk_info_bar_add_button(GtkInfoBar * info_bar,
        const char *, gint);
extern void gtk_info_bar_add_buttons(GtkInfoBar * info_bar, const
char *,
        ...);

```

```

extern GtkWidget *gtk_info_bar_get_action_area(GtkInfoBar *
info_bar);
extern GtkWidget *gtk_info_bar_get_content_area(GtkInfoBar *
info_bar);
extern GtkMessageType gtk_info_bar_get_message_type(GtkInfoBar *
info_bar);
extern GType gtk_info_bar_get_type(void);
extern GtkWidget *gtk_info_bar_new(void);
extern GtkWidget *gtk_info_bar_new_with_buttons(const char
*first_button_text, ...);
extern void gtk_info_bar_response(GtkInfoBar * info_bar, gint);
extern void gtk_info_bar_set_default_response(GtkInfoBar *
info_bar, gint);
extern void gtk_info_bar_set_message_type(GtkInfoBar * info_bar,
GtkMessageType);
extern void gtk_info_bar_set_response_sensitive(GtkInfoBar *
info_bar,
gint, gboolean);
extern void gtk_init(int *argc, char ***);
extern gboolean gtk_init_check(int *argc, char ***);
extern gboolean gtk_init_with_args(gint * argc, gchar ** , const
char *,
const GOptionEntry *, const char *,
GError * *);
extern GType gtk_input_hints_get_type(void);
extern GType gtk_input_purpose_get_type(void);
extern GdkScreen *gtk_invisible_get_screen(GtkInvisible *
invisible);
extern GType gtk_invisible_get_type(void);
extern GtkWidget *gtk_invisible_new(void);
extern GtkWidget *gtk_invisible_new_for_screen(GdkScreen * screen);
extern void gtk_invisible_set_screen(GtkInvisible * invisible,
GdkScreen *);
extern GType gtk_junction_sides_get_type(void);
extern GType gtk_justification_get_type(void);
extern guint gtk_key_snooper_install(GtkKeySnoopFunc snooper,
gpointer);
extern void gtk_key_snooper_remove(guint snooper_handler_id);
extern gdouble gtk_label_get_angle(GtkLabel * label);
extern PangoAttrList *gtk_label_get_attributes(GtkLabel * label);
extern const char *gtk_label_get_current_uri(GtkLabel * label);
extern PangoEllipsizeMode gtk_label_get_ellipsize(GtkLabel *
label);
extern GtkJustification gtk_label_get_justify(GtkLabel * label);
extern const char *gtk_label_get_label(GtkLabel * label);
extern PangoLayout *gtk_label_get_layout(GtkLabel * label);
extern void gtk_label_get_layout_offsets(GtkLabel * label, gint *,
gint *);
extern gboolean gtk_label_get_line_wrap(GtkLabel * label);
extern PangoWrapMode gtk_label_get_line_wrap_mode(GtkLabel *
label);
extern gint gtk_label_get_max_width_chars(GtkLabel * label);
extern guint gtk_label_get_mnemonic_keyval(GtkLabel * label);
extern GtkWidget *gtk_label_get_mnemonic_widget(GtkLabel * label);
extern gboolean gtk_label_get_selectable(GtkLabel * label);
extern gboolean gtk_label_get_selection_bounds(GtkLabel * label,
gint *,
gint *);
extern gboolean gtk_label_get_single_line_mode(GtkLabel * label);
extern const char *gtk_label_get_text(GtkLabel * label);
extern gboolean gtk_label_get_track_visited_links(GtkLabel *
label);
extern GType gtk_label_get_type(void);
extern gboolean gtk_label_get_use_markup(GtkLabel * label);
extern gboolean gtk_label_get_use_underline(GtkLabel * label);
extern gint gtk_label_get_width_chars(GtkLabel * label);

```

```

extern GtkWidget *gtk_label_new(const char *str);
extern GtkWidget *gtk_label_new_with_mnemonic(const char *str);
extern void gtk_label_select_region(GtkLabel * label, gint, gint);
extern void gtk_label_set_angle(GtkLabel * label, gdouble);
extern void gtk_label_set_attributes(GtkLabel * label,
PangoAttrList *);
extern void gtk_label_set_ellipsize(GtkLabel * label,
PangoEllipsizeMode);
extern void gtk_label_set_justify(GtkLabel * label,
GtkJustification);
extern void gtk_label_set_label(GtkLabel * label, const char *);
extern void gtk_label_set_line_wrap(GtkLabel * label, gboolean);
extern void gtk_label_set_line_wrap_mode(GtkLabel * label,
PangoWrapMode);
extern void gtk_label_set_markup(GtkLabel * label, const char *);
extern void gtk_label_set_markup_with_mnemonic(GtkLabel * label,
const char *);
extern void gtk_label_set_max_width_chars(GtkLabel * label, gint);
extern void gtk_label_set_mnemonic_widget(GtkLabel * label,
GtkWidget *);
extern void gtk_label_set_pattern(GtkLabel * label, const char *);
extern void gtk_label_set_selectable(GtkLabel * label, gboolean);
extern void gtk_label_set_single_line_mode(GtkLabel * label,
gboolean);
extern void gtk_label_set_text(GtkLabel * label, const char *);
extern void gtk_label_set_text_with_mnemonic(GtkLabel * label,
const char *);
extern void gtk_label_set_track_visited_links(GtkLabel * label,
gboolean);
extern void gtk_label_set_use_markup(GtkLabel * label, gboolean);
extern void gtk_label_set_use_underline(GtkLabel * label, gboolean);
extern void gtk_label_set_width_chars(GtkLabel * label, gint);
extern GdkWindow *gtk_layout_get_bin_window(GtkLayout * layout);
extern GtkAdjustment *gtk_layout_get_hadjustment(GtkLayout *
layout);
extern void gtk_layout_get_size(GtkLayout * layout, guint *, guint
*);
extern GType gtk_layout_get_type(void);
extern GtkAdjustment *gtk_layout_get_vadjustment(GtkLayout *
layout);
extern void gtk_layout_move(GtkLayout * layout, GtkWidget *, gint,
gint);
extern GtkWidget *gtk_layout_new(GtkAdjustment * hadjustment,
GtkAdjustment *);
extern void gtk_layout_put(GtkLayout * layout, GtkWidget *, gint,
gint);
extern void gtk_layout_set_hadjustment(GtkLayout * layout,
GtkAdjustment *);
extern void gtk_layout_set_size(GtkLayout * layout, guint, guint);
extern void gtk_layout_set_vadjustment(GtkLayout * layout,
GtkAdjustment *);
extern void gtk_level_bar_add_offset_value(GtkLevelBar * self,
const char *, gdouble);
extern gdouble gtk_level_bar_get_max_value(GtkLevelBar * self);
extern gdouble gtk_level_bar_get_min_value(GtkLevelBar * self);
extern GtkLevelBarMode gtk_level_bar_get_mode(GtkLevelBar * self);
extern gboolean gtk_level_bar_get_offset_value(GtkLevelBar * self,
const char *, gdouble *);
extern GType gtk_level_bar_get_type(void);
extern gdouble gtk_level_bar_get_value(GtkLevelBar * self);
extern GType gtk_level_bar_mode_get_type(void);
extern GtkWidget *gtk_level_bar_new(void);
extern GtkWidget *gtk_level_bar_new_for_interval(gdouble min_value,
gdouble);
extern void gtk_level_bar_remove_offset_value(GtkLevelBar * self,
const char *);

```

```

extern void gtk_level_bar_set_max_value(GtkLevelBar * self,
gdouble);
extern void gtk_level_bar_set_min_value(GtkLevelBar * self,
gdouble);
extern void gtk_level_bar_set_mode(GtkLevelBar * self,
GtkLevelBarMode);
extern void gtk_level_bar_set_value(GtkLevelBar * self, gdouble);
extern GType gtk_license_get_type(void);
extern GType gtk_link_button_get_type(void);
extern const char *gtk_link_button_get_uri(GtkLinkButton *
link_button);
extern gboolean gtk_link_button_get_visited(GtkLinkButton *
link_button);
extern GtkWidget *gtk_link_button_new(const char *uri);
extern GtkWidget *gtk_link_button_new_with_label(const char *uri,
const char *);
extern void gtk_link_button_set_uri(GtkLinkButton * link_button,
const char *);
extern void gtk_link_button_set_visited(GtkLinkButton *
link_button,
gboolean);
extern void gtk_list_store_append(GtkListStore * list_store,
GtkTreeIter *);
extern void gtk_list_store_clear(GtkListStore * list_store);
extern GType gtk_list_store_get_type(void);
extern void gtk_list_store_insert(GtkListStore * list_store,
GtkTreeIter *,
gint);
extern void gtk_list_store_insert_after(GtkListStore * list_store,
GtkTreeIter *, GtkTreeIter *);
extern void gtk_list_store_insert_before(GtkListStore * list_store,
GtkTreeIter *, GtkTreeIter *);
extern void gtk_list_store_insert_with_values(GtkListStore *
list_store,
GtkTreeIter *, gint, ...);
extern void gtk_list_store_insert_with_valuesv(GtkListStore *
list_store,
GtkTreeIter *, gint, gint *,
GValue *, gint);
extern gboolean gtk_list_store_iter_is_valid(GtkListStore *
list_store,
GtkTreeIter *);
extern void gtk_list_store_move_after(GtkListStore * store,
GtkTreeIter *,
GtkTreeIter *);
extern void gtk_list_store_move_before(GtkListStore * store,
GtkTreeIter *,
GtkTreeIter *);
extern GtkWidget *gtk_list_store_new(gint n_columns, ...);
extern GtkWidget *gtk_list_store_newv(gint n_columns, GType *);
extern void gtk_list_store_prepend(GtkListStore * list_store,
GtkTreeIter *);
extern gboolean gtk_list_store_remove(GtkListStore * list_store,
GtkTreeIter *);
extern void gtk_list_store_reorder(GtkListStore * store, gint *);
extern void gtk_list_store_set(GtkListStore * list_store,
GtkTreeIter *,
...);
extern void gtk_list_store_set_column_types(GtkListStore *
list_store,
gint, GType *);
extern void gtk_list_store_set_valist(GtkListStore * list_store,
GtkTreeIter * iter,
va_list var_args);
extern void gtk_list_store_set_value(GtkListStore * list_store,
GtkTreeIter *, gint, GValue *);

```



```

extern void gtk_list_store_set_valuesv(GtkListStore * list_store,
                                       GtkTreeIter *, gint *, GValue *,
                                       gint);
extern void gtk_list_store_swap(GtkListStore * store, GtkTreeIter
*,
                               GtkTreeIter *);
extern GPermission *gtk_lock_button_get_permission(GtkLockButton *
button);
extern GType gtk_lock_button_get_type(void);
extern GtkWidget *gtk_lock_button_new(GPermission * permission);
extern void gtk_lock_button_set_permission(GtkLockButton * button,
                                           GPermission *);

extern void gtk_main(void);
extern void gtk_main_do_event(GdkEvent * event);
extern gboolean gtk_main_iteration(void);
extern gboolean gtk_main_iteration_do(gboolean blocking);
extern guint gtk_main_level(void);
extern void gtk_main_quit(void);
extern void gtk_menu_attach(GtkMenu * menu, GtkWidget *, gint,
gint,
                               gint, gint);
extern void gtk_menu_attach_to_widget(GtkMenu * menu, GtkWidget *,
GtkMenuDetachFunc);
extern
                               GtkPackDirection
gtk_menu_bar_get_child_pack_direction(GtkMenuBar *
                               menubar);
extern GtkPackDirection gtk_menu_bar_get_pack_direction(GtkMenuBar
*
                               menubar);

extern GType gtk_menu_bar_get_type(void);
extern GtkWidget *gtk_menu_bar_new(void);
extern GtkWidget *gtk_menu_bar_new_from_model(GMenuModel * model);
extern void gtk_menu_bar_set_child_pack_direction(GtkMenuBar *
menubar,
                               GtkPackDirection);
extern void gtk_menu_bar_set_pack_direction(GtkMenuBar * menubar,
GtkPackDirection);
extern GtkWidget *gtk_menu_button_get_align_widget(GtkMenuButton *
menu_button);
extern GtkArrowType gtk_menu_button_get_direction(GtkMenuButton *
menu_button);
extern GMenuModel *gtk_menu_button_get_menu_model(GtkMenuButton *
menu_button);
extern
GtkMenu
*gtk_menu_button_get_popup(GtkMenuButton
*
menu_button);
extern GType gtk_menu_button_get_type(void);
extern GtkWidget *gtk_menu_button_new(void);
extern void gtk_menu_button_set_align_widget(GtkMenuButton *
menu_button,
                               GtkWidget *);
extern void
gtk_menu_button_set_direction(GtkMenuButton
*
menu_button,
                               GtkArrowType);
extern void
gtk_menu_button_set_menu_model(GtkMenuButton
*
menu_button,
                               GMenuModel *);
extern void gtk_menu_button_set_popup(GtkMenuButton * menu_button,
GtkWidget *);
extern void gtk_menu_detach(GtkMenu * menu);
extern GType gtk_menu_direction_type_get_type(void);
extern GtkAccelGroup *gtk_menu_get_accel_group(GtkMenu * menu);
extern const char *gtk_menu_get_accel_path(GtkMenu * menu);
extern GtkWidget *gtk_menu_get_active(GtkMenu * menu);
extern GtkWidget *gtk_menu_get_attach_widget(GtkMenu * menu);
extern GList *gtk_menu_get_for_attach_widget(GtkWidget * widget);
extern gint gtk_menu_get_monitor(GtkMenu * menu);

```

```

extern gboolean gtk_menu_get_reserve_toggle_size(GtkMenu * menu);
extern gboolean gtk_menu_get_tearoff_state(GtkMenu * menu);
extern const char *gtk_menu_get_title(GtkMenu * menu);
extern GType gtk_menu_get_type(void);
extern void gtk_menu_item_activate(GtkMenuItem * menu_item);
extern void gtk_menu_item_deselect(GtkMenuItem * menu_item);
extern const char *gtk_menu_item_get_accel_path(GtkMenuItem *
menu_item);
extern const char *gtk_menu_item_get_label(GtkMenuItem * menu_item);
extern gboolean gtk_menu_item_get_reserve_indicator(GtkMenuItem *
menu_item);
extern gboolean gtk_menu_item_get_right_justified(GtkMenuItem *
menu_item);
extern GtkWidget *gtk_menu_item_get_submenu(GtkMenuItem *
menu_item);
extern GType gtk_menu_item_get_type(void);
extern gboolean gtk_menu_item_get_use_underline(GtkMenuItem *
menu_item);
extern GtkWidget *gtk_menu_item_new(void);
extern GtkWidget *gtk_menu_item_new_with_label(const char *label);
extern GtkWidget *gtk_menu_item_new_with_mnemonic(const char
*label);
extern void gtk_menu_item_select(GtkMenuItem * menu_item);
extern void gtk_menu_item_set_accel_path(GtkMenuItem * menu_item,
const char *);
extern void gtk_menu_item_set_label(GtkMenuItem * menu_item, const
char *);
extern void gtk_menu_item_set_reserve_indicator(GtkMenuItem *
menu_item,
gboolean);
extern void gtk_menu_item_set_right_justified(GtkMenuItem *
menu_item,
gboolean);
extern void gtk_menu_item_set_submenu(GtkMenuItem * menu_item,
GtkWidget *);
extern void gtk_menu_item_set_use_underline(GtkMenuItem *
menu_item,
gboolean);
extern void gtk_menu_item_toggle_size_allocate(GtkMenuItem *
menu_item,
gint);
extern void gtk_menu_item_toggle_size_request(GtkMenuItem *
menu_item,
gint *);
extern GtkWidget *gtk_menu_new(void);
extern GtkWidget *gtk_menu_new_from_model(GMenuModel * model);
extern void gtk_menu_popdown(GtkMenu * menu);
extern void gtk_menu_popup(GtkMenu * menu, GtkWidget *, GtkWidget
*,
GtkMenuPositionFunc, gpointer, guint,
guint32);
extern void gtk_menu_popup_for_device(GtkMenu * menu, GdkDevice *,
GtkWidget *, GtkWidget *,
GtkMenuPositionFunc, gpointer,
GDestroyNotify, guint, guint32);
extern void gtk_menu_reorder_child(GtkMenu * menu, GtkWidget *,
gint);
extern void gtk_menu_reposition(GtkMenu * menu);
extern void gtk_menu_set_accel_group(GtkMenu * menu, GtkAccelGroup
*);
extern void gtk_menu_set_accel_path(GtkMenu * menu, const char *);
extern void gtk_menu_set_active(GtkMenu * menu, guint);
extern void gtk_menu_set_monitor(GtkMenu * menu, gint);
extern void gtk_menu_set_reserve_toggle_size(GtkMenu * menu,
gboolean);
extern void gtk_menu_set_screen(GtkMenu * menu, GdkScreen *);

```

```

extern void gtk_menu_set_tearoff_state(GtkMenu * menu, gboolean);
extern void gtk_menu_set_title(GtkMenu * menu, const char *);
extern void gtk_menu_shell_activate_item(GtkMenuShell * menu_shell,
                                         GtkWidget *, gboolean);
extern void gtk_menu_shell_append(GtkMenuShell * menu_shell,
                                   GtkWidget *);
extern void gtk_menu_shell_bind_model(GtkMenuShell * menu_shell,
                                       GMenuModel *, const char *,
                                       gboolean);
extern void gtk_menu_shell_cancel(GtkMenuShell * menu_shell);
extern void gtk_menu_shell_deactivate(GtkMenuShell * menu_shell);
extern void gtk_menu_shell_deselect(GtkMenuShell * menu_shell);
extern GtkWidget *gtk_menu_shell_get_parent_shell(GtkMenuShell *
                                                  menu_shell);
extern GtkWidget *gtk_menu_shell_get_selected_item(GtkMenuShell *
                                                  menu_shell);
extern gboolean gtk_menu_shell_get_take_focus(GtkMenuShell *
                                              menu_shell);
extern GType gtk_menu_shell_get_type(void);
extern void gtk_menu_shell_insert(GtkMenuShell * menu_shell,
                                   GtkWidget *,
                                   gint);
extern void gtk_menu_shell_prepend(GtkMenuShell * menu_shell,
                                    GtkWidget *);
extern void gtk_menu_shell_select_first(GtkMenuShell * menu_shell,
                                         gboolean);
extern void gtk_menu_shell_select_item(GtkMenuShell * menu_shell,
                                        GtkWidget *);
extern void gtk_menu_shell_set_take_focus(GtkMenuShell *
                                           menu_shell,
                                           gboolean);
extern GtkWidget *gtk_menu_tool_button_get_menu(GtkMenuToolButton *
                                                button);
extern GType gtk_menu_tool_button_get_type(void);
extern GtkToolItem *gtk_menu_tool_button_new(GtkWidget *
                                              icon_widget,
                                              const char *);
extern GtkToolItem *gtk_menu_tool_button_new_from_stock(const char *
                                                         *stock_id);
extern void gtk_menu_tool_button_set_arrow_tooltip_markup(GtkMenuToolButton *
                                                          button,
                                                          const char *);
extern void gtk_menu_tool_button_set_arrow_tooltip_text(GtkMenuToolButton *
                                                         button,
                                                         const char *);
extern void gtk_menu_tool_button_set_menu(GtkMenuToolButton *
                                           button,
                                           GtkWidget *);
extern void gtk_message_dialog_format_secondary_markup(GtkMessageDialog *
                                                       message_dialog,
                                                       const char *, ...);
extern void gtk_message_dialog_format_secondary_text(GtkMessageDialog *
                                                     message_dialog,
                                                     const char *, ...);
extern GtkWidget *gtk_message_dialog_get_image(GtkMessageDialog *
                                                dialog);
extern GtkWidget *gtk_message_dialog_get_message_area(GtkMessageDialog *
                                                       message_dialog);
extern GType gtk_message_dialog_get_type(void);
extern GtkWidget *gtk_message_dialog_new(GtkWindow * parent,

```

```

        GtkDialogFlags, GtkMessageType,
        GtkButtonsType, const char *,
        ...);
extern GtkWidget *gtk_message_dialog_new_with_markup(GtkWindow *
parent,
        GtkDialogFlags,
        GtkMessageType,
        GtkButtonsType,
        const char *, ...);
extern void gtk_message_dialog_set_image(GtkMessageDialog * dialog,
        GtkWidget *);
extern void gtk_message_dialog_set_markup(GtkMessageDialog *
        message_dialog, const char *);
extern GType gtk_message_type_get_type(void);
extern void gtk_misc_get_alignment(GtkMisc * misc, gfloat *, gfloat
*);
extern void gtk_misc_get_padding(GtkMisc * misc, gint *, gint *);
extern GType gtk_misc_get_type(void);
extern void gtk_misc_set_alignment(GtkMisc * misc, gfloat, gfloat);
extern void gtk_misc_set_padding(GtkMisc * misc, gint, gint);
extern GtkWidget *gtk_mount_operation_get_parent(GtkMountOperation
* op);
extern GdkScreen *gtk_mount_operation_get_screen(GtkMountOperation
* op);
extern GType gtk_mount_operation_get_type(void);
extern gboolean gtk_mount_operation_is_showing(GtkMountOperation *
op);
extern GMountOperation *gtk_mount_operation_new(GtkWindow *
parent);
extern void gtk_mount_operation_set_parent(GtkMountOperation * op,
        GtkWidget *);
extern void gtk_mount_operation_set_screen(GtkMountOperation * op,
        GdkScreen *);
extern GType gtk_movement_step_get_type(void);
extern gint gtk_notebook_append_page(GtkNotebook * notebook,
        GtkWidget *,
        GtkWidget *);
extern gint gtk_notebook_append_page_menu(GtkNotebook * notebook,
        GtkWidget *, GtkWidget *,
        GtkWidget *);
extern GtkWidget *gtk_notebook_get_action_widget(GtkNotebook *
notebook,
        GtkPackType);
extern gint gtk_notebook_get_current_page(GtkNotebook * notebook);
extern const char *gtk_notebook_get_group_name(GtkNotebook *
notebook);
extern GtkWidget *gtk_notebook_get_menu_label(GtkNotebook *
notebook,
        GtkWidget *);
extern const char *gtk_notebook_get_menu_label_text(GtkNotebook *
notebook,
        GtkWidget *);
extern gint gtk_notebook_get_n_pages(GtkNotebook * notebook);
extern GtkWidget *gtk_notebook_get_nth_page(GtkNotebook * notebook,
        gint);
extern gboolean gtk_notebook_get_scrollable(GtkNotebook *
notebook);
extern gboolean gtk_notebook_get_show_border(GtkNotebook *
notebook);
extern gboolean gtk_notebook_get_show_tabs(GtkNotebook * notebook);
extern gboolean gtk_notebook_get_tab_detachable(GtkNotebook *
notebook,
        GtkWidget *);
extern guint16 gtk_notebook_get_tab_hborder(GtkNotebook *
notebook);

```

```

extern GtkWidget *gtk_notebook_get_tab_label(GtkNotebook *
notebook,
                                           GtkWidget *);
extern const char *gtk_notebook_get_tab_label_text(GtkNotebook *
notebook,
                                           GtkWidget *);
extern GtkPositionType gtk_notebook_get_tab_pos(GtkNotebook *
notebook);
extern gboolean gtk_notebook_get_tab_reorderable(GtkNotebook *
notebook,
                                           GtkWidget *);
extern guint16 gtk_notebook_get_tab_vborder(GtkNotebook *
notebook);
extern GType gtk_notebook_get_type(void);
extern gint gtk_notebook_insert_page(GtkNotebook * notebook,
GtkWidget *,
                                           GtkWidget *, gint);
extern gint gtk_notebook_insert_page_menu(GtkNotebook * notebook,
                                           GtkWidget *, GtkWidget *,
                                           GtkWidget *, gint);
extern GtkWidget *gtk_notebook_new(void);
extern void gtk_notebook_next_page(GtkNotebook * notebook);
extern gint gtk_notebook_page_num(GtkNotebook * notebook, GtkWidget
*);
extern void gtk_notebook_popup_disable(GtkNotebook * notebook);
extern void gtk_notebook_popup_enable(GtkNotebook * notebook);
extern gint gtk_notebook_prepend_page(GtkNotebook * notebook,
GtkWidget *,
                                           GtkWidget *);
extern gint gtk_notebook_prepend_page_menu(GtkNotebook * notebook,
                                           GtkWidget *, GtkWidget *,
                                           GtkWidget *);
extern void gtk_notebook_prev_page(GtkNotebook * notebook);
extern void gtk_notebook_remove_page(GtkNotebook * notebook, gint);
extern void gtk_notebook_reorder_child(GtkNotebook * notebook,
GtkWidget *,
                                           gint);
extern void gtk_notebook_set_action_widget(GtkNotebook * notebook,
                                           GtkWidget *, GtkPackType);
extern void gtk_notebook_set_current_page(GtkNotebook * notebook,
gint);
extern void gtk_notebook_set_group_name(GtkNotebook * notebook,
const char *);
extern void gtk_notebook_set_menu_label(GtkNotebook * notebook,
                                           GtkWidget *, GtkWidget *);
extern void gtk_notebook_set_menu_label_text(GtkNotebook *
notebook,
                                           GtkWidget *, const char *);
extern void gtk_notebook_set_scrollable(GtkNotebook * notebook,
gboolean);
extern void gtk_notebook_set_show_border(GtkNotebook * notebook,
gboolean);
extern void gtk_notebook_set_show_tabs(GtkNotebook * notebook,
gboolean);
extern void gtk_notebook_set_tab_detachable(GtkNotebook * notebook,
                                           GtkWidget *, gboolean);
extern void gtk_notebook_set_tab_label(GtkNotebook * notebook,
GtkWidget *,
                                           GtkWidget *);
extern void gtk_notebook_set_tab_label_text(GtkNotebook * notebook,
                                           GtkWidget *, const char *);
extern void gtk_notebook_set_tab_pos(GtkNotebook * notebook,
                                           GtkPositionType);
extern void gtk_notebook_set_tab_reorderable(GtkNotebook *
notebook,
                                           GtkWidget *, gboolean);

```

```

extern GType gtk_notebook_tab_get_type(void);
extern GType gtk_number_up_layout_get_type(void);
extern GIcon
*gtk_numerable_icon_get_background_gicon(GtkNumerableIcon *
self);
extern const char
*gtk_numerable_icon_get_background_icon_name(GtkNumerableIcon
* self);
extern gint gtk_numerable_icon_get_count(GtkNumerableIcon * self);
extern const char *gtk_numerable_icon_get_label(GtkNumerableIcon *
self);
extern GtkStyleContext
*gtk_numerable_icon_get_style_context(GtkNumerableIcon * self);
extern GType gtk_numerable_icon_get_type(void);
extern GIcon *gtk_numerable_icon_new(GIcon * base_icon);
extern GIcon *gtk_numerable_icon_new_with_style_context(GIcon *
base_icon,
GtkStyleContext *);
extern void
gtk_numerable_icon_set_background_gicon(GtkNumerableIcon *
self, GIcon *);
extern void
gtk_numerable_icon_set_background_icon_name(GtkNumerableIcon *
self,
const char *);
extern void gtk_numerable_icon_set_count(GtkNumerableIcon * self,
gint);
extern void gtk_numerable_icon_set_label(GtkNumerableIcon * self,
const char *);
extern void gtk_numerable_icon_set_style_context(GtkNumerableIcon
* self,
GtkStyleContext *);
extern void
*gtk_offscreen_window_get_pixbuf(GtkOffscreenWindow *
offscreen);
extern void
*gtk_offscreen_window_get_surface(GtkOffscreenWindow
* offscreen);
extern GType gtk_offscreen_window_get_type(void);
extern GtkWidget *gtk_offscreen_window_new(void);
extern GtkOrientation gtk_orientable_get_orientation(GtkOrientable
*
orientable);
extern GType gtk_orientable_get_type(void);
extern void
gtk_orientable_set_orientation(GtkOrientable *
orientable,
GtkOrientation);
extern GType gtk_orientation_get_type(void);
extern void gtk_overlay_add_overlay(GtkOverlay * overlay, GtkWidget
*);
extern GType gtk_overlay_get_type(void);
extern GtkWidget *gtk_overlay_new(void);
extern GType gtk_pack_direction_get_type(void);
extern GType gtk_pack_type_get_type(void);
extern GType gtk_page_orientation_get_type(void);
extern GType gtk_page_set_get_type(void);
extern GtkPageSetup *gtk_page_setup_copy(GtkPageSetup * other);
extern gdouble
gtk_page_setup_get_bottom_margin(GtkPageSetup *
setup,
GtkUnit);
extern gdouble
gtk_page_setup_get_left_margin(GtkPageSetup * setup,
GtkUnit);
extern
GtkPageOrientation
gtk_page_setup_get_orientation(GtkPageSetup *
setup);
extern gdouble
gtk_page_setup_get_page_height(GtkPageSetup * setup,

```

```

        GtkUnit);
extern gdouble gtk_page_setup_get_page_width(GtkPageSetup * setup,
        GtkUnit);
extern gdouble gtk_page_setup_get_paper_height(GtkPageSetup *
setup,
        GtkUnit);
extern GtkPaperSize *gtk_page_setup_get_paper_size(GtkPageSetup *
setup);
extern gdouble gtk_page_setup_get_paper_width(GtkPageSetup * setup,
        GtkUnit);
extern gdouble gtk_page_setup_get_right_margin(GtkPageSetup *
setup,
        GtkUnit);
extern gdouble gtk_page_setup_get_top_margin(GtkPageSetup * setup,
        GtkUnit);
extern GType gtk_page_setup_get_type(void);
extern gboolean gtk_page_setup_load_file(GtkPageSetup * setup,
        const char *, GError * *);
extern gboolean gtk_page_setup_load_key_file(GtkPageSetup * setup,
        GKeyFile *, const char *,
        GError * *);
extern GtkPageSetup *gtk_page_setup_new(void);
extern GtkPageSetup *gtk_page_setup_new_from_file(const char
*file_name,
        GError * *);
extern GtkPageSetup *gtk_page_setup_new_from_key_file(GKeyFile *
key_file,
        const char *,
        GError * *);
extern void gtk_page_setup_set_bottom_margin(GtkPageSetup * setup,
gdouble,
        GtkUnit);
extern void gtk_page_setup_set_left_margin(GtkPageSetup * setup,
gdouble,
        GtkUnit);
extern void gtk_page_setup_set_orientation(GtkPageSetup * setup,
        GtkPageOrientation);
extern void gtk_page_setup_set_paper_size(GtkPageSetup * setup,
        GtkPaperSize *);
extern
        void
gtk_page_setup_set_paper_size_and_default_margins(GtkPageSetup
        * setup,
        GtkPaperSize
        *);
extern void gtk_page_setup_set_right_margin(GtkPageSetup * setup,
gdouble,
        GtkUnit);
extern void gtk_page_setup_set_top_margin(GtkPageSetup * setup,
gdouble,
        GtkUnit);
extern gboolean gtk_page_setup_to_file(GtkPageSetup * setup, const
char *,
        GError * *);
extern void gtk_page_setup_to_key_file(GtkPageSetup * setup,
GKeyFile *,
        const char *);
extern void gtk_paned_add1(GtkPaned * paned, GtkWidget *);
extern void gtk_paned_add2(GtkPaned * paned, GtkWidget *);
extern GtkWidget *gtk_paned_get_child1(GtkPaned * paned);
extern GtkWidget *gtk_paned_get_child2(GtkPaned * paned);
extern GdkWindow *gtk_paned_get_handle_window(GtkPaned * paned);
extern gint gtk_paned_get_position(GtkPaned * paned);
extern GType gtk_paned_get_type(void);
extern GtkWidget *gtk_paned_new(GtkOrientation orientation);
extern void gtk_paned_pack1(GtkPaned * paned, GtkWidget *, gboolean,
        gboolean);

```

```

extern void gtk_paned_pack2(GtkPaned * paned, GtkWidget *, gboolean,
                             gboolean);
extern void gtk_paned_set_position(GtkPaned * paned, gint);
extern GtkPaperSize *gtk_paper_size_copy(GtkPaperSize * other);
extern void gtk_paper_size_free(GtkPaperSize * size);
extern const char *gtk_paper_size_get_default(void);
extern
                             gdouble
gtk_paper_size_get_default_bottom_margin(GtkPaperSize *
                                         size, GtkUnit);
extern gdouble gtk_paper_size_get_default_left_margin(GtkPaperSize
* size,
                                         GtkUnit);
extern
                             gdouble
gtk_paper_size_get_default_right_margin(GtkPaperSize * size,
                                         GtkUnit);
extern gdouble gtk_paper_size_get_default_top_margin(GtkPaperSize
* size,
                                         GtkUnit);
extern const char *gtk_paper_size_get_display_name(GtkPaperSize *
size);
extern gdouble gtk_paper_size_get_height(GtkPaperSize * size,
GtkUnit);
extern const char *gtk_paper_size_get_name(GtkPaperSize * size);
extern GList *gtk_paper_size_get_paper_sizes(gboolean
include_custom);
extern const char *gtk_paper_size_get_ppd_name(GtkPaperSize * size);
extern GType gtk_paper_size_get_type(void);
extern gdouble gtk_paper_size_get_width(GtkPaperSize * size,
GtkUnit);
extern gboolean gtk_paper_size_is_custom(GtkPaperSize * size);
extern gboolean gtk_paper_size_is_equal(GtkPaperSize * size1,
                                       GtkPaperSize * size2);
extern GtkPaperSize *gtk_paper_size_new(const char *name);
extern GtkPaperSize *gtk_paper_size_new_custom(const char *name,
                                              const char *, gdouble,
                                              gdouble, GtkUnit);
extern GtkPaperSize *gtk_paper_size_new_from_key_file(GKeyFile *
key_file,
                                              const char *,
                                              GError * *);
extern GtkPaperSize *gtk_paper_size_new_from_ppd(const char
*ppd_name,
                                              const char *, gdouble,
                                              gdouble);
extern void gtk_paper_size_set_size(GtkPaperSize * size, gdouble,
gdouble,
                             GtkUnit);
extern void gtk_paper_size_to_key_file(GtkPaperSize * size,
GKeyFile *,
                             const char *);
extern gboolean gtk_parse_args(int *argc, char ***);
extern GType gtk_path_priority_type_get_type(void);
extern GType gtk_path_type_get_type(void);
extern void gtk_plug_construct(GtkPlug * plug, Window);
extern void gtk_plug_construct_for_display(GtkPlug * plug,
GdkDisplay *,
                             Window);
extern gboolean gtk_plug_get_embedded(GtkPlug * plug);
extern Window gtk_plug_get_id(GtkPlug * plug);
extern GdkWindow *gtk_plug_get_socket_window(GtkPlug * plug);
extern GType gtk_plug_get_type(void);
extern GtkWidget *gtk_plug_new(Window socket_id);
extern GtkWidget *gtk_plug_new_for_display(GdkDisplay * display,
Window);
extern GType gtk_policy_type_get_type(void);
extern GType gtk_position_type_get_type(void);

```



```

extern                                PangoContext
*gtk_print_context_create_pango_context(GtkPrintContext
                                      * context);

extern                                PangoLayout
*gtk_print_context_create_pango_layout(GtkPrintContext *
                                      context);

extern                                cairo_t
*gtk_print_context_get_cairo_context(GtkPrintContext *
                                      context);

extern gdouble   gtk_print_context_get_dpi_x(GtkPrintContext *
context);
extern gdouble   gtk_print_context_get_dpi_y(GtkPrintContext *
context);
extern gboolean  gtk_print_context_get_hard_margins(GtkPrintContext
*
                                      context, gdouble *,
                                      gdouble *, gdouble *,
                                      gdouble *);

extern gdouble   gtk_print_context_get_height(GtkPrintContext *
context);
extern                                GtkPageSetup
*gtk_print_context_get_page_setup(GtkPrintContext *
                                   context);

extern                                PangoFontMap
*gtk_print_context_get_pango_fontmap(GtkPrintContext *
                                      context);

extern GType      gtk_print_context_get_type(void);
extern gdouble    gtk_print_context_get_width(GtkPrintContext *
context);
extern void       gtk_print_context_set_cairo_context(GtkPrintContext *
context,
                                      cairo_t *, double, double);

extern GType      gtk_print_duplex_get_type(void);
extern GType      gtk_print_error_get_type(void);
extern GQuark     gtk_print_error_quark(void);
extern GType      gtk_print_operation_action_get_type(void);
extern void       gtk_print_operation_cancel(GtkPrintOperation * op);
extern void       gtk_print_operation_draw_page_finish(GtkPrintOperation
* op);
extern GtkPageSetup
*gtk_print_operation_get_default_page_setup(GtkPrintOperation
* op);
extern gboolean   gtk_print_operation_get_embed_page_setup(GtkPrintOperation
* op);
extern void       gtk_print_operation_get_error(GtkPrintOperation * op,
                                                GError * *);
extern gboolean   gtk_print_operation_get_has_selection(GtkPrintOperation *
op);
extern gint       gtk_print_operation_get_n_pages_to_print(GtkPrintOperation *
op);
extern GtkPrintSettings
*gtk_print_operation_get_print_settings(GtkPrintOperation *
op);
extern GtkPrintStatus
gtk_print_operation_get_status(GtkPrintOperation *
op);
extern const char
*gtk_print_operation_get_status_string(GtkPrintOperation
* op);
extern gboolean   gtk_print_operation_get_support_selection(GtkPrintOperation
* op);
extern GType      gtk_print_operation_get_type(void);

```

```

extern gboolean gtk_print_operation_is_finished(GtkPrintOperation
* op);
extern GtkPrintOperation *gtk_print_operation_new(void);
extern void
gtk_print_operation_preview_end_preview(GtkPrintOperationPreview *
preview);
extern GType gtk_print_operation_preview_get_type(void);
extern gboolean
gtk_print_operation_preview_is_selected(GtkPrintOperationPreview *
preview,
gint);
extern void
gtk_print_operation_preview_render_page(GtkPrintOperationPreview *
preview,
gint);
extern GType gtk_print_operation_result_get_type(void);
extern GtkPrintOperationResult
gtk_print_operation_run(GtkPrintOperation *
op,
GtkPrintOperationAction,
GtkWindow *,
GError **);
extern void gtk_print_operation_set_allow_async(GtkPrintOperation
* op,
gboolean);
extern void gtk_print_operation_set_current_page(GtkPrintOperation
* op,
gint);
extern void
gtk_print_operation_set_custom_tab_label(GtkPrintOperation *
op, const char *);
extern void
gtk_print_operation_set_default_page_setup(GtkPrintOperation *
op, GtkPageSetup *);
extern void
gtk_print_operation_set_defer_drawing(GtkPrintOperation * op);
extern void
gtk_print_operation_set_embed_page_setup(GtkPrintOperation *
op, gboolean);
extern void
gtk_print_operation_set_export_filename(GtkPrintOperation * op,
const char *);
extern void
gtk_print_operation_set_has_selection(GtkPrintOperation * op,
gboolean);
extern void gtk_print_operation_set_job_name(GtkPrintOperation *
op,
const char *);
extern void gtk_print_operation_set_n_pages(GtkPrintOperation * op,
gint);
extern void
gtk_print_operation_set_print_settings(GtkPrintOperation * op,
GtkPrintSettings *);
extern void
gtk_print_operation_set_show_progress(GtkPrintOperation * op,
gboolean);
extern void
gtk_print_operation_set_support_selection(GtkPrintOperation *
op, gboolean);
extern void
gtk_print_operation_set_track_print_status(GtkPrintOperation *
op, gboolean);
extern void gtk_print_operation_set_unit(GtkPrintOperation * op,
GtkUnit);

```

```

extern void
gtk_print_operation_set_use_full_page(GtkPrintOperation * op,
                                      gboolean);
extern GType gtk_print_pages_get_type(void);
extern GType gtk_print_quality_get_type(void);
extern GtkWidget *gtk_print_run_page_setup_dialog(GtkWindow *
parent,
                                                  GtkWidget *,
                                                  GtkPrintSettings *);
extern void gtk_print_run_page_setup_dialog_async(GtkWindow *
parent,
                                                  GtkWidget *,
                                                  GtkPrintSettings *,
                                                  GtkWidgetDoneFunc,
                                                  gpointer);
extern GtkPrintSettings *gtk_print_settings_copy(GtkPrintSettings
* other);
extern void gtk_print_settings_foreach(GtkPrintSettings * settings,
                                       GtkPrintSettingsFunc, gpointer);
extern const char *gtk_print_settings_get(GtkPrintSettings *
settings,
                                          const char *);
extern gboolean gtk_print_settings_get_bool(GtkPrintSettings *
settings,
                                          const char *);
extern gboolean gtk_print_settings_get_collate(GtkPrintSettings *
settings);
extern const char *gtk_print_settings_get_default_source(GtkPrintSettings *
settings);
extern const char *gtk_print_settings_get_dither(GtkPrintSettings
*
settings);
extern gdouble gtk_print_settings_get_double(GtkPrintSettings *
settings,
                                             const char *);
extern gdouble gtk_print_settings_get_double_with_default(GtkPrintSettings
* settings,
                                                         const char *,
                                                         gdouble);
extern GtkWidget *gtk_print_settings_get_duplex(GtkPrintSettings *
settings);
extern const char *gtk_print_settings_get_finishings(GtkPrintSettings *
settings);
extern gint gtk_print_settings_get_int(GtkPrintSettings * settings,
                                       const char *);
extern gint gtk_print_settings_get_int_with_default(GtkPrintSettings *
settings, const char *,
                                                    gint);
extern gdouble gtk_print_settings_get_length(GtkPrintSettings *
settings,
                                             const char *, GtkUnit);
extern const char *gtk_print_settings_get_media_type(GtkPrintSettings *
settings);
extern gint gtk_print_settings_get_n_copies(GtkPrintSettings *
settings);
extern gint gtk_print_settings_get_number_up(GtkPrintSettings *
settings);
extern GtkWidget *gtk_print_settings_get_number_up_layout(GtkPrintSettings
*
settings);

```

```

extern GtkPageOrientation
gtk_print_settings_get_orientation(GtkPrintSettings * settings);
extern          const          char
*gtk_print_settings_get_output_bin(GtkPrintSettings *
                                settings);
extern          GtkPageRange
*gtk_print_settings_get_page_ranges(GtkPrintSettings *
                                settings, gint *);
extern GtkPageSet gtk_print_settings_get_page_set(GtkPrintSettings
*
                                settings);
extern          gdouble
gtk_print_settings_get_paper_height(GtkPrintSettings *
                                settings, GtkUnit);
extern          GtkPaperSize
*gtk_print_settings_get_paper_size(GtkPrintSettings *
                                settings);
extern gdouble gtk_print_settings_get_paper_width(GtkPrintSettings
*
                                settings, GtkUnit);
extern          GtkPrintPages
gtk_print_settings_get_print_pages(GtkPrintSettings *
                                settings);
extern const char *gtk_print_settings_get_printer(GtkPrintSettings
*
                                settings);
extern gdouble gtk_print_settings_get_printer_dpi(GtkPrintSettings
*
                                settings);
extern          GtkPrintQuality
gtk_print_settings_get_quality(GtkPrintSettings *
                                settings);
extern gint gtk_print_settings_get_resolution(GtkPrintSettings *
settings);
extern gint gtk_print_settings_get_resolution_x(GtkPrintSettings *
                                settings);
extern gint gtk_print_settings_get_resolution_y(GtkPrintSettings *
                                settings);
extern gboolean gtk_print_settings_get_reverse(GtkPrintSettings *
                                settings);
extern gdouble gtk_print_settings_get_scale(GtkPrintSettings *
settings);
extern GType gtk_print_settings_get_type(void);
extern gboolean gtk_print_settings_get_use_color(GtkPrintSettings
*
                                settings);
extern gboolean gtk_print_settings_has_key(GtkPrintSettings *
settings,
                                const char *);
extern gboolean gtk_print_settings_load_file(GtkPrintSettings *
settings,
                                const char *, GError * *);
extern gboolean gtk_print_settings_load_key_file(GtkPrintSettings
*
                                settings, GKeyFile *,
                                const char *, GError * *);
extern GtkPrintSettings *gtk_print_settings_new(void);
extern GtkPrintSettings *gtk_print_settings_new_from_file(const
char
                                *file_name,
                                GError * *);
extern          GtkPrintSettings
*gtk_print_settings_new_from_key_file(GKeyFile *
                                key_file,
                                const char *,
                                GError * *);

```

```

extern void gtk_print_settings_set(GtkPrintSettings * settings,
                                   const char *, const char *);
extern void gtk_print_settings_set_bool(GtkPrintSettings *
settings,
                                   const char *, gboolean);
extern void gtk_print_settings_set_collate(GtkPrintSettings *
settings,
                                   gboolean);
extern void gtk_print_settings_set_default_source(GtkPrintSettings
*
                                   settings, const char *);
extern void gtk_print_settings_set_dither(GtkPrintSettings *
settings,
                                   const char *);
extern void gtk_print_settings_set_double(GtkPrintSettings *
settings,
                                   const char *, gdouble);
extern void gtk_print_settings_set_duplex(GtkPrintSettings *
settings,
                                   GtkPrintDuplex);
extern void gtk_print_settings_set_finishings(GtkPrintSettings *
settings,
                                   const char *);
extern void gtk_print_settings_set_int(GtkPrintSettings * settings,
                                   const char *, gint);
extern void gtk_print_settings_set_length(GtkPrintSettings *
settings,
                                   const char *, gdouble, GtkUnit);
extern void gtk_print_settings_set_media_type(GtkPrintSettings *
settings,
                                   const char *);
extern void gtk_print_settings_set_n_copies(GtkPrintSettings *
settings,
                                   gint);
extern void gtk_print_settings_set_number_up(GtkPrintSettings *
settings,
                                   gint);
extern void gtk_print_settings_set_number_up_layout(GtkPrintSettings *
settings,
                                   GtkNumberUpLayout);
extern void gtk_print_settings_set_orientation(GtkPrintSettings *
settings,
                                   GtkPageOrientation);
extern void gtk_print_settings_set_output_bin(GtkPrintSettings *
settings,
                                   const char *);
extern void gtk_print_settings_set_page_ranges(GtkPrintSettings *
settings,
                                   GtkPageRange *, gint);
extern void gtk_print_settings_set_page_set(GtkPrintSettings *
settings,
                                   GtkPageSet);
extern void gtk_print_settings_set_paper_height(GtkPrintSettings *
settings, gdouble,
                                   GtkUnit);
extern void gtk_print_settings_set_paper_size(GtkPrintSettings *
settings,
                                   GtkPaperSize *);
extern void gtk_print_settings_set_paper_width(GtkPrintSettings *
settings,
                                   gdouble, GtkUnit);
extern void gtk_print_settings_set_print_pages(GtkPrintSettings *
settings,
                                   GtkPrintPages);

```

```

extern void gtk_print_settings_set_printer(GtkPrintSettings *
settings,
const char *);
extern void gtk_print_settings_set_printer_lpi(GtkPrintSettings *
settings,
gdouble);
extern void gtk_print_settings_set_quality(GtkPrintSettings *
settings,
GtkPrintQuality);
extern void gtk_print_settings_set_resolution(GtkPrintSettings *
settings,
gint);
extern void gtk_print_settings_set_resolution_xy(GtkPrintSettings
*
settings, gint, gint);
extern void gtk_print_settings_set_reverse(GtkPrintSettings *
settings,
gboolean);
extern void gtk_print_settings_set_scale(GtkPrintSettings *
settings,
gdouble);
extern void gtk_print_settings_set_use_color(GtkPrintSettings *
settings,
gboolean);
extern gboolean gtk_print_settings_to_file(GtkPrintSettings *
settings,
const char *, GError * *);
extern void gtk_print_settings_to_key_file(GtkPrintSettings *
settings,
GKeyFile *, const char *);
extern void gtk_print_settings_unset(GtkPrintSettings * settings,
const char *);
extern GType gtk_print_status_get_type(void);
extern PangoEllipsizeMode
gtk_progress_bar_get_ellipsize(GtkProgressBar *
pbar);
extern gdouble gtk_progress_bar_get_fraction(GtkProgressBar *
pbar);
extern gboolean gtk_progress_bar_get_inverted(GtkProgressBar *
pbar);
extern gdouble gtk_progress_bar_get_pulse_step(GtkProgressBar *
pbar);
extern gboolean gtk_progress_bar_get_show_text(GtkProgressBar *
pbar);
extern const char *gtk_progress_bar_get_text(GtkProgressBar * pbar);
extern GType gtk_progress_bar_get_type(void);
extern GtkWidget *gtk_progress_bar_new(void);
extern void gtk_progress_bar_pulse(GtkProgressBar * pbar);
extern void gtk_progress_bar_set_ellipsize(GtkProgressBar * pbar,
PangoEllipsizeMode);
extern void gtk_progress_bar_set_fraction(GtkProgressBar * pbar,
gdouble);
extern void gtk_progress_bar_set_inverted(GtkProgressBar * pbar,
gboolean);
extern void gtk_progress_bar_set_pulse_step(GtkProgressBar * pbar,
gdouble);
extern void gtk_progress_bar_set_show_text(GtkProgressBar * pbar,
gboolean);
extern void gtk_progress_bar_set_text(GtkProgressBar * pbar, const
char *);
extern void gtk_propagate_event(GtkWidget * widget, GdkEvent *);
extern gint gtk_radio_action_get_current_value(GtkRadioAction *
action);
extern GSList *gtk_radio_action_get_group(GtkRadioAction * action);
extern GType gtk_radio_action_get_type(void);
extern void gtk_radio_action_join_group(GtkRadioAction * action,

```

```

        GtkRadioAction *);
extern GtkRadioAction *gtk_radio_action_new(const char *name, const
char *,
        const char *, const char *,
        gint);
extern void gtk_radio_action_set_current_value(GtkRadioAction *
action,
        gint);
extern void gtk_radio_action_set_group(GtkRadioAction * action,
GSLIST *);
extern GSLIST *gtk_radio_button_get_group(GtkRadioButton *
radio_button);
extern GType gtk_radio_button_get_type(void);
extern void gtk_radio_button_join_group(GtkRadioButton *
radio_button,
        GtkRadioButton *);
extern GtkWidget *gtk_radio_button_new(GSLIST * group);
extern GtkWidget *gtk_radio_button_new_from_widget(GtkRadioButton
*
        radio_group_member);
extern GtkWidget *gtk_radio_button_new_with_label(GSLIST * group,
const char *);
extern GtkWidget
*gtk_radio_button_new_with_label_from_widget(GtkRadioButton *
        radio_group_member,
        const char *);
extern GtkWidget *gtk_radio_button_new_with_mnemonic(GSLIST *
group,
        const char *);
extern GtkWidget
*gtk_radio_button_new_with_mnemonic_from_widget(GtkRadioButton
*
        radio_group_member,
        const char *);
extern void gtk_radio_button_set_group(GtkRadioButton *
radio_button,
        GSLIST *);
extern GSLIST *gtk_radio_menu_item_get_group(GtkRadioMenuItem *
radio_menu_item);
extern GType gtk_radio_menu_item_get_type(void);
extern GtkWidget *gtk_radio_menu_item_new(GSLIST * group);
extern
GtkWidget
*gtk_radio_menu_item_new_from_widget(GtkRadioMenuItem *
        group);
extern GtkWidget *gtk_radio_menu_item_new_with_label(GSLIST *
group,
        const char *);
extern GtkWidget
*gtk_radio_menu_item_new_with_label_from_widget(GtkRadioMenuItem *
        group, const char *);
extern GtkWidget *gtk_radio_menu_item_new_with_mnemonic(GSLIST *
group,
        const char *);
extern GtkWidget
*gtk_radio_menu_item_new_with_mnemonic_from_widget(GtkRadioMenuIt
em *
        group,
        const char *);
extern void gtk_radio_menu_item_set_group(GtkRadioMenuItem *
radio_menu_item, GSLIST *);
extern GSLIST *gtk_radio_tool_button_get_group(GtkRadioToolButton
*
        button);
extern GType gtk_radio_tool_button_get_type(void);

```

```

extern GtkWidget *gtk_radio_tool_button_new(GSList * group);
extern GtkWidget *gtk_radio_tool_button_new_from_stock(GSList *
group,
const char *);
extern GtkWidget
*gtk_radio_tool_button_new_from_widget(GtkRadioToolButton *
group);
extern GtkWidget

*gtk_radio_tool_button_new_with_stock_from_widget(GtkRadioToolBut
ton *
group, const char *);
extern void gtk_radio_tool_button_set_group(GtkRadioToolButton *
button,
GSList *);
extern GtkAdjustment *gtk_range_get_adjustment(GtkRange * range);
extern gdouble gtk_range_get_fill_level(GtkRange * range);
extern gboolean gtk_range_get_flippable(GtkRange * range);
extern gboolean gtk_range_get_inverted(GtkRange * range);
extern
GtkSensitivityType
gtk_range_get_lower_stepper_sensitivity(GtkRange
* range);
extern gint gtk_range_get_min_slider_size(GtkRange * range);
extern void gtk_range_get_range_rect(GtkRange * range, GdkRectangle
*);
extern gboolean gtk_range_get_restrict_to_fill_level(GtkRange *
range);
extern gint gtk_range_get_round_digits(GtkRange * range);
extern gboolean gtk_range_get_show_fill_level(GtkRange * range);
extern void gtk_range_get_slider_range(GtkRange * range, gint *,
gint *);
extern gboolean gtk_range_get_slider_size_fixed(GtkRange * range);
extern GType gtk_range_get_type(void);
extern
GtkSensitivityType
gtk_range_get_upper_stepper_sensitivity(GtkRange
* range);
extern gdouble gtk_range_get_value(GtkRange * range);
extern void gtk_range_set_adjustment(GtkRange * range,
GtkAdjustment *);
extern void gtk_range_set_fill_level(GtkRange * range, gdouble);
extern void gtk_range_set_flippable(GtkRange * range, gboolean);
extern void gtk_range_set_increments(GtkRange * range, gdouble,
gdouble);
extern void gtk_range_set_inverted(GtkRange * range, gboolean);
extern void gtk_range_set_lower_stepper_sensitivity(GtkRange *
range,
GtkSensitivityType);
extern void gtk_range_set_min_slider_size(GtkRange * range, gint);
extern void gtk_range_set_range(GtkRange * range, gdouble, gdouble);
extern void gtk_range_set_restrict_to_fill_level(GtkRange * range,
gboolean);
extern void gtk_range_set_round_digits(GtkRange * range, gint);
extern void gtk_range_set_show_fill_level(GtkRange * range,
gboolean);
extern void gtk_range_set_slider_size_fixed(GtkRange * range,
gboolean);
extern void gtk_range_set_upper_stepper_sensitivity(GtkRange *
range,
GtkSensitivityType);
extern void gtk_range_set_value(GtkRange * range, gdouble);
extern GType gtk_rc_flags_get_type(void);
extern gboolean gtk_rc_property_parse_border(const GParamSpec *
pspec,
const GString *, GValue *);
extern gboolean gtk_rc_property_parse_color(const GParamSpec *
pspec,

```



```

        const GString *, GValue *);
extern gboolean gtk_rc_property_parse_enum(const GParamSpec * pspec,
        const GString *, GValue *);
extern gboolean gtk_rc_property_parse_flags(const GParamSpec *
pspec,
        const GString *, GValue *);
extern gboolean gtk_rc_property_parse_requisition(const GParamSpec
* pspec,
        const GString *,
        GValue *);
extern GType gtk_rc_token_type_get_type(void);
extern gboolean gtk_recent_action_get_show_numbers(GtkRecentAction
*
        action);
extern GType gtk_recent_action_get_type(void);
extern GtkAction *gtk_recent_action_new(const char *name, const
char *,
        const char *, const char *);
extern GtkAction *gtk_recent_action_new_for_manager(const char
*name,
        const char *,
        const char *,
        GtkRecentManager *);
extern void gtk_recent_action_set_show_numbers(GtkRecentAction *
action,
        gboolean);
extern void gtk_recent_chooser_add_filter(GtkRecentChooser *
chooser,
        GtkRecentFilter *);
extern GType gtk_recent_chooser_dialog_get_type(void);
extern GtkWidget *gtk_recent_chooser_dialog_new(const char *title,
        GtkWindow *, const char *,
        ...);
extern GtkWidget *gtk_recent_chooser_dialog_new_for_manager(const
char
        *title,
        GtkWindow *,

        GtkRecentManager
        *,
        const char *,
        ...);
extern GType gtk_recent_chooser_error_get_type(void);
extern GQuark gtk_recent_chooser_error_quark(void);
extern GtkRecentInfo
*gtk_recent_chooser_get_current_item(GtkRecentChooser
        * chooser);
extern gchar *gtk_recent_chooser_get_current_uri(GtkRecentChooser
*
        chooser);
extern
        GtkRecentFilter
*gtk_recent_chooser_get_filter(GtkRecentChooser *
        chooser);
extern GList *gtk_recent_chooser_get_items(GtkRecentChooser *
chooser);
extern gint gtk_recent_chooser_get_limit(GtkRecentChooser *
chooser);
extern gboolean gtk_recent_chooser_get_local_only(GtkRecentChooser
*
        chooser);
extern
        gboolean
gtk_recent_chooser_get_select_multiple(GtkRecentChooser *
        chooser);
extern gboolean gtk_recent_chooser_get_show_icons(GtkRecentChooser
*

```

```

        chooser);
extern
gtk_recent_chooser_get_show_not_found(GtkRecentChooser *
        chooser);
extern
gtk_recent_chooser_get_show_private(GtkRecentChooser *
        chooser);
extern gboolean gtk_recent_chooser_get_show_tips(GtkRecentChooser
*
        chooser);
extern
        GtkRecentSortType
gtk_recent_chooser_get_sort_type(GtkRecentChooser
        * chooser);
extern GType gtk_recent_chooser_get_type(void);
extern gchar **gtk_recent_chooser_get_uris(GtkRecentChooser *
chooser,
        gsize *);
extern GSList *gtk_recent_chooser_list_filters(GtkRecentChooser *
chooser);
extern gboolean
gtk_recent_chooser_menu_get_show_numbers(GtkRecentChooserMenu *
menu);
extern GType gtk_recent_chooser_menu_get_type(void);
extern GtkWidget *gtk_recent_chooser_menu_new(void);
extern
        GtkWidget
*gtk_recent_chooser_menu_new_for_manager(GtkRecentManager
        * manager);
extern
        void
gtk_recent_chooser_menu_set_show_numbers(GtkRecentChooserMenu *
        menu, gboolean);
extern void gtk_recent_chooser_remove_filter(GtkRecentChooser *
chooser,
        GtkRecentFilter *);
extern void gtk_recent_chooser_select_all(GtkRecentChooser *
chooser);
extern gboolean gtk_recent_chooser_select_uri(GtkRecentChooser *
chooser,
        const char *, GError * *);
extern
        gboolean
gtk_recent_chooser_set_current_uri(GtkRecentChooser *
        chooser, const char *,
        GError * *);
extern void gtk_recent_chooser_set_filter(GtkRecentChooser *
chooser,
        GtkRecentFilter *);
extern void gtk_recent_chooser_set_limit(GtkRecentChooser *
chooser, gint);
extern void gtk_recent_chooser_set_local_only(GtkRecentChooser *
chooser,
        gboolean);
extern
        void
gtk_recent_chooser_set_select_multiple(GtkRecentChooser *
        chooser, gboolean);
extern void gtk_recent_chooser_set_show_icons(GtkRecentChooser *
chooser,
        gboolean);
extern void gtk_recent_chooser_set_show_not_found(GtkRecentChooser
*
        chooser, gboolean);
extern void gtk_recent_chooser_set_show_private(GtkRecentChooser *
chooser,
        gboolean);
extern void gtk_recent_chooser_set_show_tips(GtkRecentChooser *
chooser,
        gboolean);

```

```

extern void gtk_recent_chooser_set_sort_func(GtkRecentChooser *
chooser,
                                           GtkRecentSortFunc, gpointer,
                                           GDestroyNotify);
extern void gtk_recent_chooser_set_sort_type(GtkRecentChooser *
chooser,
                                           GtkRecentSortType);
extern void gtk_recent_chooser_unselect_all(GtkRecentChooser *
chooser);
extern void gtk_recent_chooser_unselect_uri(GtkRecentChooser *
chooser,
                                           const char *);
extern GType gtk_recent_chooser_widget_get_type(void);
extern GtkWidget *gtk_recent_chooser_widget_new(void);
extern GtkWidget
*gtk_recent_chooser_widget_new_for_manager(GtkRecentManager *
manager);
extern void gtk_recent_filter_add_age(GtkRecentFilter * filter,
gint);
extern void gtk_recent_filter_add_application(GtkRecentFilter *
filter,
                                           const char *);
extern void gtk_recent_filter_add_custom(GtkRecentFilter * filter,
                                           GtkRecentFilterFlags,
                                           GtkRecentFilterFunc, gpointer,
                                           GDestroyNotify);
extern void gtk_recent_filter_add_group(GtkRecentFilter * filter,
                                           const char *);
extern void gtk_recent_filter_add_mime_type(GtkRecentFilter *
filter,
                                           const char *);
extern void gtk_recent_filter_add_pattern(GtkRecentFilter * filter,
                                           const char *);
extern void gtk_recent_filter_add_pixbuf_formats(GtkRecentFilter *
filter);
extern gboolean gtk_recent_filter_filter(GtkRecentFilter * filter,
                                           const GtkRecentFilterInfo *);
extern GType gtk_recent_filter_flags_get_type(void);
extern const char *gtk_recent_filter_get_name(GtkRecentFilter *
filter);
extern
                                           GtkRecentFilterFlags
gtk_recent_filter_get_needed(GtkRecentFilter *
                                           filter);
extern GType gtk_recent_filter_get_type(void);
extern GtkRecentFilter *gtk_recent_filter_new(void);
extern void gtk_recent_filter_set_name(GtkRecentFilter * filter,
                                           const char *);
extern GAppInfo *gtk_recent_info_create_app_info(GtkRecentInfo *
info,
                                           const char *, GError * *);
extern gboolean gtk_recent_info_exists(GtkRecentInfo * info);
extern time_t gtk_recent_info_get_added(GtkRecentInfo * info);
extern gint gtk_recent_info_get_age(GtkRecentInfo * info);
extern gboolean gtk_recent_info_get_application_info(GtkRecentInfo
* info,
                                           const char *,
                                           const char **,
                                           guint *, time_t *);
extern gchar **gtk_recent_info_get_applications(GtkRecentInfo *
info,
                                           gsize *);
extern const char *gtk_recent_info_get_description(GtkRecentInfo *
info);
extern const char *gtk_recent_info_get_display_name(GtkRecentInfo
* info);
extern GIcon *gtk_recent_info_get_gicon(GtkRecentInfo * info);

```

```

extern gchar **gtk_recent_info_get_groups(GtkRecentInfo * info,
                                           gsize *);
extern GdkPixbuf *gtk_recent_info_get_icon(GtkRecentInfo * info,
                                           gint);
extern const char *gtk_recent_info_get_mime_type(GtkRecentInfo *
info);
extern time_t gtk_recent_info_get_modified(GtkRecentInfo * info);
extern gboolean gtk_recent_info_get_private_hint(GtkRecentInfo *
info);
extern gchar *gtk_recent_info_get_short_name(GtkRecentInfo * info);
extern GType gtk_recent_info_get_type(void);
extern const char *gtk_recent_info_get_uri(GtkRecentInfo * info);
extern gchar *gtk_recent_info_get_uri_display(GtkRecentInfo *
info);
extern time_t gtk_recent_info_get_visited(GtkRecentInfo * info);
extern gboolean gtk_recent_info_has_application(GtkRecentInfo *
info,
                                           const char *);
extern gboolean gtk_recent_info_has_group(GtkRecentInfo * info,
                                           const char *);
extern gboolean gtk_recent_info_is_local(GtkRecentInfo * info);
extern gchar *gtk_recent_info_last_application(GtkRecentInfo *
info);
extern gboolean gtk_recent_info_match(GtkRecentInfo * info_a,
                                      GtkRecentInfo *);
extern GtkRecentInfo *gtk_recent_info_ref(GtkRecentInfo * info);
extern void gtk_recent_info_unref(GtkRecentInfo * info);
extern gboolean gtk_recent_manager_add_full(GtkRecentManager *
manager,
                                           const char *,
                                           const GtkRecentData *);
extern gboolean gtk_recent_manager_add_item(GtkRecentManager *
manager,
                                           const char *);
extern GType gtk_recent_manager_error_get_type(void);
extern GQuark gtk_recent_manager_error_quark(void);
extern GtkRecentManager *gtk_recent_manager_get_default(void);
extern GList *gtk_recent_manager_get_items(GtkRecentManager *
manager);
extern GType gtk_recent_manager_get_type(void);
extern gboolean gtk_recent_manager_has_item(GtkRecentManager *
manager,
                                           const char *);
extern
                                           GtkRecentInfo
*gtk_recent_manager_lookup_item(GtkRecentManager *
manager, const char *,
                                GError * *);
extern gboolean gtk_recent_manager_move_item(GtkRecentManager *
manager,
                                           const char *, const char *,
                                           GError * *);
extern GtkRecentManager *gtk_recent_manager_new(void);
extern gint gtk_recent_manager_purge_items(GtkRecentManager *
manager,
                                           GError * *);
extern gboolean gtk_recent_manager_remove_item(GtkRecentManager *
manager,
                                           const char *, GError * *);
extern GType gtk_recent_sort_type_get_type(void);
extern GType gtk_region_flags_get_type(void);
extern GType gtk_relief_style_get_type(void);
extern void gtk_render_activity(GtkStyleContext * context, cairo_t
*,
                                gdouble, gdouble, gdouble, gdouble);
extern void gtk_render_arrow(GtkStyleContext * context, cairo_t *,
gdouble,

```

```

                                gdouble, gdouble, gdouble);
extern void gtk_render_background(GtkStyleContext * context,
cairo_t *,
                                gdouble, gdouble, gdouble, gdouble);
extern void gtk_render_check(GtkStyleContext * context, cairo_t *,
gdouble,
                                gdouble, gdouble, gdouble);
extern void gtk_render_expander(GtkStyleContext * context, cairo_t
*,
                                gdouble, gdouble, gdouble, gdouble);
extern void gtk_render_extension(GtkStyleContext * context, cairo_t
*,
                                gdouble, gdouble, gdouble, gdouble,
                                GtkPositionType);
extern void gtk_render_focus(GtkStyleContext * context, cairo_t *,
gdouble,
                                gdouble, gdouble, gdouble);
extern void gtk_render_frame(GtkStyleContext * context, cairo_t *,
gdouble,
                                gdouble, gdouble, gdouble);
extern void gtk_render_frame_gap(GtkStyleContext * context, cairo_t
*,
                                gdouble, gdouble, gdouble, gdouble,
                                GtkPositionType, gdouble, gdouble);
extern void gtk_render_handle(GtkStyleContext * context, cairo_t *,
                                gdouble, gdouble, gdouble, gdouble);
extern void gtk_render_icon(GtkStyleContext * context, cairo_t *,
                                GdkPixbuf *, gdouble, gdouble);
extern GdkPixbuf *gtk_render_icon_pixbuf(GtkStyleContext * context,
                                const GtkIconSource * source,
                                GtkIconSize size);
extern void gtk_render_insertion_cursor(GtkStyleContext * context,
cairo_t *, gdouble, gdouble,
                                PangoLayout *, int,
                                PangoDirection);
extern void gtk_render_layout(GtkStyleContext * context, cairo_t *,
                                gdouble, gdouble, PangoLayout *);
extern void gtk_render_line(GtkStyleContext * context, cairo_t *,
gdouble,
                                gdouble, gdouble, gdouble);
extern void gtk_render_option(GtkStyleContext * context, cairo_t *,
                                gdouble, gdouble, gdouble, gdouble);
extern void gtk_render_slider(GtkStyleContext * context, cairo_t *,
                                gdouble, gdouble, gdouble, gdouble,
                                GtkOrientation);
extern GtkRequisition *gtk_requisition_copy(const GtkRequisition *
requisition);
extern void gtk_requisition_free(GtkRequisition * requisition);
extern GType gtk_requisition_get_type(void);
extern GtkRequisition *gtk_requisition_new(void);
extern GType gtk_resize_mode_get_type(void);
extern GType gtk_response_type_get_type(void);
extern void gtk_rgb_to_hsv(gdouble r, gdouble, gdouble, gdouble *,
                                gdouble *, gdouble *);
extern void gtk_scale_add_mark(GtkScale * scale, gdouble,
                                GtkPositionType,
                                const char *);
extern
                                GtkAdjustment
*gtk_scale_button_get_adjustment(GtkScaleButton *
                                button);
extern GtkWidget *gtk_scale_button_get_minus_button(GtkScaleButton
*
                                button);
extern GtkWidget *gtk_scale_button_get_plus_button(GtkScaleButton
*
                                button);

```

```

extern GtkWidget *gtk_scale_button_get_popup(GtkScaleButton *
button);
extern GType gtk_scale_button_get_type(void);
extern gdouble gtk_scale_button_get_value(GtkScaleButton * button);
extern GtkWidget *gtk_scale_button_new(GtkIconSize size, gdouble,
gdouble,
                                gdouble, const char **);
extern void gtk_scale_button_set_adjustment(GtkScaleButton *
button,
                                GtkAdjustment *);
extern void gtk_scale_button_set_icons(GtkScaleButton * button,
                                const char **);
extern void gtk_scale_button_set_value(GtkScaleButton * button,
gdouble);
extern void gtk_scale_clear_marks(GtkScale * scale);
extern gint gtk_scale_get_digits(GtkScale * scale);
extern gboolean gtk_scale_get_draw_value(GtkScale * scale);
extern gboolean gtk_scale_get_has_origin(GtkScale * scale);
extern PangoLayout *gtk_scale_get_layout(GtkScale * scale);
extern void gtk_scale_get_layout_offsets(GtkScale * scale, gint *,
gint *);
extern GType gtk_scale_get_type(void);
extern GtkPositionType gtk_scale_get_value_pos(GtkScale * scale);
extern GtkWidget *gtk_scale_new(GtkOrientation orientation,
                                GtkAdjustment *);
extern GtkWidget *gtk_scale_new_with_range(GtkOrientation
orientation,
                                gdouble, gdouble, gdouble);
extern void gtk_scale_set_digits(GtkScale * scale, gint);
extern void gtk_scale_set_draw_value(GtkScale * scale, gboolean);
extern void gtk_scale_set_has_origin(GtkScale * scale, gboolean);
extern void gtk_scale_set_value_pos(GtkScale * scale,
GtkPositionType);
extern GType gtk_scroll_step_get_type(void);
extern GType gtk_scroll_type_get_type(void);
extern GtkAdjustment *gtk_scrollable_get_hadjustment(GtkScrollable
*
                                scrollable);
extern
                                GtkScrollablePolicy
gtk_scrollable_get_hscroll_policy(GtkScrollable
                                * scrollable);
extern GType gtk_scrollable_get_type(void);
extern GtkAdjustment *gtk_scrollable_get_vadjustment(GtkScrollable
*
                                scrollable);
extern
                                GtkScrollablePolicy
gtk_scrollable_get_vscroll_policy(GtkScrollable
                                * scrollable);
extern GType gtk_scrollable_policy_get_type(void);
extern void gtk_scrollable_set_hadjustment(GtkScrollable *
scrollable,
                                GtkAdjustment *);
extern void gtk_scrollable_set_hscroll_policy(GtkScrollable *
scrollable,
                                GtkScrollablePolicy);
extern void gtk_scrollable_set_vadjustment(GtkScrollable *
scrollable,
                                GtkAdjustment *);
extern void gtk_scrollable_set_vscroll_policy(GtkScrollable *
scrollable,
                                GtkScrollablePolicy);
extern GType gtk_scrollbar_get_type(void);
extern GtkWidget *gtk_scrollbar_new(GtkOrientation orientation,
                                GtkAdjustment *);
extern
                                void
gtk_scrolled_window_add_with_viewport(GtkScrolledWindow *

```

```

        scrolled_window,
        GtkWidget *);

extern gboolean
gtk_scrolled_window_get_capture_button_press(GtkScrolledWindow *
        scrolled_window);

extern
        GtkAdjustment
*gtk_scrolled_window_get_hadjustment(GtkScrolledWindow
        *
        scrolled_window);

extern
        GtkWidget
*gtk_scrolled_window_get_hscrollbar(GtkScrolledWindow *
        scrolled_window);

extern
        gboolean
gtk_scrolled_window_get_kinetic_scrolling(GtkScrolledWindow
        *
        scrolled_window);

extern
        gint
gtk_scrolled_window_get_min_content_height(GtkScrolledWindow *
        scrolled_window);

extern
        gint
gtk_scrolled_window_get_min_content_width(GtkScrolledWindow *
        scrolled_window);

extern
        GtkCornerType
gtk_scrolled_window_get_placement(GtkScrolledWindow *
        scrolled_window);

extern void gtk_scrolled_window_get_policy(GtkScrolledWindow *
        scrolled_window,
        GtkPolicyType *,
        GtkPolicyType *);

extern
        GtkShadowType
gtk_scrolled_window_get_shadow_type(GtkScrolledWindow
        *
        scrolled_window);

extern GType gtk_scrolled_window_get_type(void);

extern
        GtkAdjustment
*gtk_scrolled_window_get_vadjustment(GtkScrolledWindow
        *
        scrolled_window);

extern
        GtkWidget
*gtk_scrolled_window_get_vscrollbar(GtkScrolledWindow *
        scrolled_window);

extern GtkWidget *gtk_scrolled_window_new(GtkAdjustment *
        hadjustment,
        GtkAdjustment *);

extern
        void
gtk_scrolled_window_set_capture_button_press(GtkScrolledWindow
        * scrolled_window,
        gboolean);

extern void gtk_scrolled_window_set_hadjustment(GtkScrolledWindow
        *
        scrolled_window,
        GtkAdjustment *);

extern
        void
gtk_scrolled_window_set_kinetic_scrolling(GtkScrolledWindow *
        scrolled_window,
        gboolean);

extern
        void
gtk_scrolled_window_set_min_content_height(GtkScrolledWindow *
        scrolled_window,
        gint);

extern
        void
gtk_scrolled_window_set_min_content_width(GtkScrolledWindow *
        scrolled_window,
        gint);

extern void gtk_scrolled_window_set_placement(GtkScrolledWindow *
        scrolled_window,

```

```

                                GtkCornerType);
extern void gtk_scrolled_window_set_policy(GtkScrolledWindow *
                                scrolled_window, GtkPolicyType,
                                GtkPolicyType);
extern void gtk_scrolled_window_set_shadow_type(GtkScrolledWindow
*
                                scrolled_window,
                                GtkShadowType);
extern void gtk_scrolled_window_set_vadjustment(GtkScrolledWindow
*
                                scrolled_window,
                                GtkAdjustment *);
extern void gtk_scrolled_window_unset_placement(GtkScrolledWindow
*
                                scrolled_window);
extern GType gtk_search_entry_get_type(void);
extern GtkWidget *gtk_search_entry_new(void);
extern void gtk_selection_add_target(GtkWidget * widget, GdkAtom
selection,
                                GdkAtom target, guint info);
extern void gtk_selection_add_targets(GtkWidget * widget,
                                GdkAtom selection,
                                const GtkTargetEntry * targets,
                                guint ntargets);
extern void gtk_selection_clear_targets(GtkWidget * widget,
                                GdkAtom selection);
extern gboolean gtk_selection_convert(GtkWidget * widget,
                                GdkAtom selection, GdkAtom target,
                                guint32 time);
extern      GtkSelectionData      *gtk_selection_data_copy(const
GtkSelectionData *
                                data);
extern void gtk_selection_data_free(GtkSelectionData * data);
extern const unsigned char *gtk_selection_data_get_data(const
                                GtkSelectionData *
                                selection_data);
extern      GdkAtom      gtk_selection_data_get_data_type(const
GtkSelectionData *
                                selection_data);
extern      const      unsigned      char
*gtk_selection_data_get_data_with_length(const
                                GtkSelectionData
                                *
                                selection_data,
                                gint
                                *);
extern      GdkDisplay      *gtk_selection_data_get_display(const
GtkSelectionData *
                                selection_data);
extern gint gtk_selection_data_get_format(const GtkSelectionData *
                                selection_data);
extern gint gtk_selection_data_get_length(const GtkSelectionData *
                                selection_data);
extern      GdkPixbuf      *gtk_selection_data_get_pixbuf(const
GtkSelectionData *
                                selection_data);
extern      GdkAtom      gtk_selection_data_get_selection(const
GtkSelectionData *
                                selection_data);
extern      GdkAtom      gtk_selection_data_get_target(const
GtkSelectionData *
                                selection_data);
extern      gboolean      gtk_selection_data_get_targets(const
GtkSelectionData *

```



```

selection_data, GdkAtom * *,
gint *);
extern gchar *gtk_selection_data_get_text(const GtkSelectionData
*
selection_data);
extern GType gtk_selection_data_get_type(void);
extern gchar **gtk_selection_data_get_uris(const GtkSelectionData
*
selection_data);
extern void gtk_selection_data_set(GtkSelectionData *
selection_data,
GdkAtom type, gint format,
const gchar * data, gint length);
extern gboolean gtk_selection_data_set_pixbuf(GtkSelectionData *
selection_data, GdkPixbuf
*);
extern gboolean gtk_selection_data_set_text(GtkSelectionData *
selection_data, const char *,
gint);
extern gboolean gtk_selection_data_set_uris(GtkSelectionData *
selection_data, gchar * *);
extern gboolean gtk_selection_data_targets_include_image(const
GtkSelectionData *
selection_data,
gboolean);
extern gboolean gtk_selection_data_targets_include_rich_text(const
GtkSelectionData
*
selection_data,
GtkTextBuffer
*);
extern gboolean gtk_selection_data_targets_include_text(const
GtkSelectionData *
selection_data);
extern gboolean gtk_selection_data_targets_include_uri(const
GtkSelectionData *
selection_data);
extern GType gtk_selection_mode_get_type(void);
extern gboolean gtk_selection_owner_set(GtkWidget * widget,
GdkAtom selection, guint32 time_);
extern gboolean gtk_selection_owner_set_for_display(GdkDisplay *
display,
GtkWidget * widget,
GdkAtom selection,
guint32 time_);
extern void gtk_selection_remove_all(GtkWidget * widget);
extern GType gtk_sensitivity_type_get_type(void);
extern GType gtk_separator_get_type(void);
extern GType gtk_separator_menu_item_get_type(void);
extern GtkWidget *gtk_separator_menu_item_new(void);
extern GtkWidget *gtk_separator_new(GtkOrientation orientation);
extern gboolean
gtk_separator_tool_item_get_draw(GtkSeparatorToolItem *
item);
extern GType gtk_separator_tool_item_get_type(void);
extern GtkToolItem *gtk_separator_tool_item_new(void);
extern void gtk_separator_tool_item_set_draw(GtkSeparatorToolItem
* item,
gboolean);
extern void gtk_set_debug_flags(guint flags);
extern GtkSettings *gtk_settings_get_default(void);
extern GtkSettings *gtk_settings_get_for_screen(GdkScreen *
screen);
extern GType gtk_settings_get_type(void);
extern void gtk_settings_install_property(GParamSpec * pspec);

```

```

extern void gtk_settings_install_property_parser(GParamSpec *
pspec,
                                GtkRcPropertyParser);
extern void gtk_settings_set_double_property(GtkSettings *
settings,
                                const char *, gdouble,
                                const char *);
extern void gtk_settings_set_long_property(GtkSettings * settings,
                                const char *, glong,
                                const char *);
extern void gtk_settings_set_property_value(GtkSettings * settings,
                                const char *,
                                const GtkSettingsValue *);
extern void gtk_settings_set_string_property(GtkSettings *
settings,
                                const char *, const char *,
                                const char *);
extern GType gtk_shadow_type_get_type(void);
extern void gtk_show_about_dialog(GtkWindow * parent, const char
*, ...);
extern gboolean gtk_show_uri(GdkScreen * screen, const char *,
guint32,
                                GError * *);
extern void gtk_size_group_add_widget(GtkSizeGroup * size_group,
                                GtkWidget *);
extern gboolean gtk_size_group_get_ignore_hidden(GtkSizeGroup *
size_group);
extern GtkSizeGroupMode gtk_size_group_get_mode(GtkSizeGroup *
size_group);
extern GType gtk_size_group_get_type(void);
extern GSList *gtk_size_group_get_widgets(GtkSizeGroup *
size_group);
extern GType gtk_size_group_mode_get_type(void);
extern GtkSizeGroup *gtk_size_group_new(GtkSizeGroupMode mode);
extern void gtk_size_group_remove_widget(GtkSizeGroup * size_group,
                                GtkWidget *);
extern void gtk_size_group_set_ignore_hidden(GtkSizeGroup *
size_group,
                                gboolean);
extern void gtk_size_group_set_mode(GtkSizeGroup * size_group,
                                GtkSizeGroupMode);
extern GType gtk_size_request_mode_get_type(void);
extern void gtk_socket_add_id(GtkSocket * socket_, Window);
extern Window gtk_socket_get_id(GtkSocket * socket_);
extern GdkWindow *gtk_socket_get_plug_window(GtkSocket * socket_);
extern GType gtk_socket_get_type(void);
extern GtkWidget *gtk_socket_new(void);
extern GType gtk_sort_type_get_type(void);
extern void gtk_spin_button_configure(GtkSpinButton * spin_button,
                                GtkAdjustment *, gdouble, guint);
extern GtkAdjustment *gtk_spin_button_get_adjustment(GtkSpinButton
*
                                spin_button);
extern guint gtk_spin_button_get_digits(GtkSpinButton *
spin_button);
extern void gtk_spin_button_get_increments(GtkSpinButton *
spin_button,
                                gdouble *, gdouble *);
extern gboolean gtk_spin_button_get_numeric(GtkSpinButton *
spin_button);
extern void gtk_spin_button_get_range(GtkSpinButton * spin_button,
                                gdouble *, gdouble *);
extern gboolean gtk_spin_button_get_snap_to_ticks(GtkSpinButton *
spin_button);
extern GType gtk_spin_button_get_type(void);
extern GtkSpinButtonUpdatePolicy

```

```

gtk_spin_button_get_update_policy(GtkSpinButton * spin_button);
extern gdouble gtk_spin_button_get_value(GtkSpinButton *
spin_button);
extern gint gtk_spin_button_get_value_as_int(GtkSpinButton *
spin_button);
extern gboolean gtk_spin_button_get_wrap(GtkSpinButton *
spin_button);
extern GtkWidget *gtk_spin_button_new(GtkAdjustment * adjustment,
gdouble,
guint);
extern GtkWidget *gtk_spin_button_new_with_range(gdouble min,
gdouble,
gdouble);
extern void gtk_spin_button_set_adjustment(GtkSpinButton *
spin_button,
GtkAdjustment *);
extern void gtk_spin_button_set_digits(GtkSpinButton * spin_button,
guint);
extern void gtk_spin_button_set_increments(GtkSpinButton *
spin_button,
gdouble, gdouble);
extern void gtk_spin_button_set_numeric(GtkSpinButton *
spin_button,
gboolean);
extern void gtk_spin_button_set_range(GtkSpinButton * spin_button,
gdouble,
gdouble);
extern void gtk_spin_button_set_snap_to_ticks(GtkSpinButton *
spin_button,
gboolean);
extern void gtk_spin_button_set_update_policy(GtkSpinButton *
spin_button,
GtkSpinButtonUpdatePolicy);
extern void gtk_spin_button_set_value(GtkSpinButton * spin_button,
gdouble);
extern void gtk_spin_button_set_wrap(GtkSpinButton * spin_button,
gboolean);
extern void gtk_spin_button_spin(GtkSpinButton * spin_button,
GtkSpinType,
gdouble);
extern void gtk_spin_button_update(GtkSpinButton * spin_button);
extern GType gtk_spin_button_update_policy_get_type(void);
extern GType gtk_spin_type_get_type(void);
extern GType gtk_spinner_get_type(void);
extern GtkWidget *gtk_spinner_new(void);
extern void gtk_spinner_start(GtkSpinner * spinner);
extern void gtk_spinner_stop(GtkSpinner * spinner);
extern GType gtk_state_flags_get_type(void);
extern GType gtk_state_type_get_type(void);
extern gboolean gtk_status_icon_get_geometry(GtkStatusIcon *
status_icon,
GdkScreen * *, GdkRectangle
*,
GtkOrientation *);
extern GIcon *gtk_status_icon_get_gicon(GtkStatusIcon *
status_icon);
extern gboolean gtk_status_icon_get_has_tooltip(GtkStatusIcon *
status_icon);
extern const char *gtk_status_icon_get_icon_name(GtkStatusIcon *
status_icon);
extern GdkPixbuf *gtk_status_icon_get_pixbuf(GtkStatusIcon *
status_icon);
extern GdkScreen *gtk_status_icon_get_screen(GtkStatusIcon *
status_icon);
extern gint gtk_status_icon_get_size(GtkStatusIcon * status_icon);

```

```

extern const char *gtk_status_icon_get_stock(GtkStatusIcon *
status_icon);
extern GtkImageType gtk_status_icon_get_storage_type(GtkStatusIcon
*
status_icon);
extern const char *gtk_status_icon_get_title(GtkStatusIcon *
status_icon);
extern gchar *gtk_status_icon_get_tooltip_markup(GtkStatusIcon *
status_icon);
extern gchar *gtk_status_icon_get_tooltip_text(GtkStatusIcon *
status_icon);
extern GType gtk_status_icon_get_type(void);
extern gboolean gtk_status_icon_get_visible(GtkStatusIcon *
status_icon);
extern guint32 gtk_status_icon_get_x11_window_id(GtkStatusIcon *
status_icon);
extern gboolean gtk_status_icon_is_embedded(GtkStatusIcon *
status_icon);
extern GtkStatusIcon *gtk_status_icon_new(void);
extern GtkStatusIcon *gtk_status_icon_new_from_file(const char
*filename);
extern GtkStatusIcon *gtk_status_icon_new_from_gicon(GIcon * icon);
extern GtkStatusIcon *gtk_status_icon_new_from_icon_name(const
char
*icon_name);
extern GtkStatusIcon *gtk_status_icon_new_from_pixbuf(GdkPixbuf *
pixbuf);
extern GtkStatusIcon *gtk_status_icon_new_from_stock(const char
*stock_id);
extern void gtk_status_icon_position_menu(GtkMenu * menu, gint *,
gint *,
gboolean *, gpointer);
extern void gtk_status_icon_set_from_file(GtkStatusIcon *
status_icon,
const char *);
extern void gtk_status_icon_set_from_gicon(GtkStatusIcon *
status_icon,
GIcon *);
extern void gtk_status_icon_set_from_icon_name(GtkStatusIcon *
status_icon,
const char *);
extern void gtk_status_icon_set_from_pixbuf(GtkStatusIcon *
status_icon,
GdkPixbuf *);
extern void gtk_status_icon_set_from_stock(GtkStatusIcon *
status_icon,
const char *);
extern void gtk_status_icon_set_has_tooltip(GtkStatusIcon *
status_icon,
gboolean);
extern void gtk_status_icon_set_name(GtkStatusIcon * status_icon,
const char *);
extern void gtk_status_icon_set_screen(GtkStatusIcon * status_icon,
GdkScreen *);
extern void gtk_status_icon_set_title(GtkStatusIcon * status_icon,
const char *);
extern void gtk_status_icon_set_tooltip_markup(GtkStatusIcon *
status_icon,
const char *);
extern void gtk_status_icon_set_tooltip_text(GtkStatusIcon *
status_icon,
const char *);
extern void gtk_status_icon_set_visible(GtkStatusIcon *
status_icon,
gboolean);
extern guint gtk_statusbar_get_context_id(GtkStatusbar * statusbar,

```

```

                                const char *);
extern GtkWidget *gtk_statusbar_get_message_area(GtkStatusbar *
statusbar);
extern GType gtk_statusbar_get_type(void);
extern GtkWidget *gtk_statusbar_new(void);
extern void gtk_statusbar_pop(GtkStatusbar * statusbar, guint);
extern guint gtk_statusbar_push(GtkStatusbar * statusbar, guint,
                                const char *);
extern void gtk_statusbar_remove(GtkStatusbar * statusbar, guint,
guint);
extern void gtk_statusbar_remove_all(GtkStatusbar * statusbar,
guint);
extern void gtk_stock_add(const GtkStockItem * items, guint);
extern void gtk_stock_add_static(const GtkStockItem * items, guint);
extern GtkStockItem *gtk_stock_item_copy(const GtkStockItem * item);
extern void gtk_stock_item_free(GtkStockItem * item);
extern GSList *gtk_stock_list_ids(void);
extern gboolean gtk_stock_lookup(const char *stock_id, GtkStockItem
*);
extern void gtk_stock_set_translate_func(const char *domain,
                                GtkTranslateFunc, gpointer,
                                GDestroyNotify);
extern void gtk_style_context_add_class(GtkStyleContext * context,
                                const char *);
extern void gtk_style_context_add_provider(GtkStyleContext *
context,
                                GtkStyleProvider *, guint);
extern void gtk_style_context_add_provider_for_screen(GdkScreen *
screen,
                                GtkStyleProvider *,
                                guint);
extern void gtk_style_context_add_region(GtkStyleContext * context,
                                const char *, GtkRegionFlags);
extern void gtk_style_context_cancel_animations(GtkStyleContext *
context,
                                gpointer);
extern void gtk_style_context_get(GtkStyleContext * context,
GtkStateFlags,
                                ...);
extern void gtk_style_context_get_background_color(GtkStyleContext
*
                                context, GtkStateFlags,
                                GdkRGBA *);
extern void gtk_style_context_get_border(GtkStyleContext * context,
                                GtkStateFlags, GtkBorder *);
extern void gtk_style_context_get_border_color(GtkStyleContext *
context,
                                GtkStateFlags, GdkRGBA *);
extern void gtk_style_context_get_color(GtkStyleContext * context,
                                GtkStateFlags, GdkRGBA *);
extern void gtk_style_context_get_direction(GtkStyleContext *
context);
extern const PangoFontDescription
*gtk_style_context_get_font(GtkStyleContext * context,
GtkStateFlags);
extern GtkJunctionSides
gtk_style_context_get_junction_sides(GtkStyleContext * context);
extern void gtk_style_context_get_margin(GtkStyleContext * context,
                                GtkStateFlags, GtkBorder *);
extern void gtk_style_context_get_padding(GtkStyleContext *
context,
                                GtkStateFlags, GtkBorder *);
extern void gtk_style_context_get_parent(GtkStyleContext *
context);

```

```

extern          const          GtkWidgetPath
*gtk_style_context_get_path(GtkStyleContext *
                           context);
extern void      gtk_style_context_get_property(GtkStyleContext *
context,
                           const char *, GtkStateFlags,
                           GValue *);
extern GdkScreen *gtk_style_context_get_screen(GtkStyleContext *
context);
extern          GtkWidgetSection
*gtk_style_context_get_section(GtkStyleContext *
                           context, const char *);
extern GtkStateFlags gtk_style_context_get_state(GtkStyleContext *
context);
extern void      gtk_style_context_get_style(GtkStyleContext *
context, ...);
extern void      gtk_style_context_get_style_property(GtkStyleContext *
context,
                           const char *, GValue *);
extern void      gtk_style_context_get_style_valist(GtkStyleContext *
context,
                           va_list);
extern GType      gtk_style_context_get_type(void);
extern void      gtk_style_context_get_valist(GtkStyleContext * context,
                           GtkStateFlags state,
                           va_list var args);
extern gboolean   gtk_style_context_has_class(GtkStyleContext *
context,
                           const char *);
extern gboolean   gtk_style_context_has_region(GtkStyleContext *
context,
                           const char *,
                           GtkRegionFlags *);
extern void      gtk_style_context_invalidate(GtkStyleContext *
context);
extern GList      *gtk_style_context_list_classes(GtkStyleContext *
context);
extern GList      *gtk_style_context_list_regions(GtkStyleContext *
context);
extern gboolean   gtk_style_context_lookup_color(GtkStyleContext *
context,
                           const char *, GdkRGBA *);
extern          GtkWidgetIconSet
*gtk_style_context_lookup_icon_set(GtkStyleContext *
                           context,
                           const char *);
extern GtkWidget *gtk_style_context_new(void);
extern void      gtk_style_context_notify_state_change(GtkStyleContext
*
                           context, GdkWindow *,
                           gpointer, GtkStateType,
                           gboolean);
extern          void
gtk_style_context_pop_animatable_region(GtkStyleContext *
                           context);
extern          void
gtk_style_context_push_animatable_region(GtkStyleContext *
                           context, gpointer);
extern void      gtk_style_context_remove_class(GtkStyleContext *
context,
                           const char *);
extern void      gtk_style_context_remove_provider(GtkStyleContext *
context,
                           GtkWidgetProvider *);
extern void      gtk_style_context_remove_provider_for_screen(GdkScreen
*

```

```

        screen,
        GtkStyleProvider
    *);
extern void gtk_style_context_remove_region(GtkStyleContext *
context,
        const char *);
extern void gtk_style_context_reset_widgets(GdkScreen * screen);
extern void gtk_style_context_restore(GtkStyleContext * context);
extern void gtk_style_context_save(GtkStyleContext * context);
extern void gtk_style_context_scroll_animations(GtkStyleContext *
context,
        GdkWindow *, gint, gint);
extern void gtk_style_context_set_background(GtkStyleContext *
context,
        GdkWindow *);
extern void gtk_style_context_set_direction(GtkStyleContext *
context,
        GtkTextDirection);
extern void gtk_style_context_set_junction_sides(GtkStyleContext *
context,
        GtkJunctionSides);
extern void gtk_style_context_set_parent(GtkStyleContext * context,
        GtkStyleContext *);
extern void gtk_style_context_set_path(GtkStyleContext * context,
        GtkWidgetPath *);
extern void gtk_style_context_set_screen(GtkStyleContext * context,
        GdkScreen *);
extern void gtk_style_context_set_state(GtkStyleContext * context,
        GtkStateFlags);
extern gboolean gtk_style_context_state_is_running(GtkStyleContext
*
        context, GtkStateType,
        gdouble *);
extern void gtk_style_properties_clear(GtkStyleProperties * props);
extern void gtk_style_properties_get(GtkStyleProperties * props,
        GtkStateFlags, ...);
extern
        gboolean
gtk_style_properties_get_property(GtkStyleProperties *
        props, const char *,
        GtkStateFlags, GValue *);
extern GType gtk_style_properties_get_type(void);
extern void gtk_style_properties_get_valist(GtkStyleProperties *
props,
        GtkStateFlags state,
        va_list var_args);
extern GtkSymbolicColor
        *gtk_style_properties_lookup_color(GtkStyleProperties * props,
        const char *);
extern gboolean gtk_style_properties_lookup_property(const char
        *property_name,
        GtkStylePropertyParser
        *, GParamSpec * *);
extern void gtk_style_properties_map_color(GtkStyleProperties *
props,
        const char *,
        GtkSymbolicColor *);
extern void gtk_style_properties_merge(GtkStyleProperties * props,
        const GtkStyleProperties *,
        gboolean);
extern GtkStyleProperties *gtk_style_properties_new(void);
extern
        void
gtk_style_properties_register_property(GtkStylePropertyParser
        parse_func,
        GParamSpec *);
extern void gtk_style_properties_set(GtkStyleProperties * props,

```

```

                                GtkStateFlags, ...);
extern void gtk_style_properties_set_property(GtkStyleProperties *
props,
                                const char *, GtkStateFlags,
                                const GValue *);
extern void gtk_style_properties_set_valist(GtkStyleProperties *
props,
                                GtkStateFlags state, va_list);
extern void gtk_style_properties_unset_property(GtkStyleProperties
* props,
                                const char *,
                                GtkStateFlags);
extern
                                GtkIconFactory
*gtk_style_provider_get_icon_factory(GtkStyleProvider
                                * provider,
                                GtkWidgetPath
                                *);
extern
                                GtkStyleProperties
*gtk_style_provider_get_style(GtkStyleProvider *
                                provider,
                                GtkWidgetPath *);
extern
                                gboolean
gtk_style_provider_get_style_property(GtkStyleProvider *
                                provider,
                                GtkWidgetPath *,
                                GtkStateFlags,
                                GParamSpec *,
                                GValue *);

extern GType gtk_style_provider_get_type(void);
extern gboolean gtk_switch_get_active(GtkSwitch * sw);
extern GType gtk_switch_get_type(void);
extern GtkWidget *gtk_switch_new(void);
extern void gtk_switch_set_active(GtkSwitch * sw, gboolean);
extern GType gtk_symbolic_color_get_type(void);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_alpha(GtkSymbolicColor *
                                color, gdouble);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_literal(const GdkRGBA *
                                color);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_mix(GtkSymbolicColor *
                                color1,
                                GtkSymbolicColor *,
                                gdouble);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_name(const char
*name);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_shade(GtkSymbolicColor *
                                color, gdouble);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_new_win32(const char
*theme_class, gint);
extern
                                GtkSymbolicColor
*gtk_symbolic_color_ref(GtkSymbolicColor *
color);
extern
                                gboolean
gtk_symbolic_color_resolve(GtkSymbolicColor *
color,
                                GtkStyleProperties *,
                                GdkRGBA *);
extern
                                char
*gtk_symbolic_color_to_string(GtkSymbolicColor *
color);
extern void gtk_symbolic_color_unref(GtkSymbolicColor * color);
extern
                                GtkTargetEntry
*gtk_target_entry_copy(GtkTargetEntry *
data);
extern void gtk_target_entry_free(GtkTargetEntry * data);
extern GType gtk_target_entry_get_type(void);

```



```

extern GtkTargetEntry *gtk_target_entry_new(const char *target,
guint,
guint);
extern GType gtk_target_flags_get_type(void);
extern void gtk_target_list_add(GtkTargetList * list, GdkAtom
target,
guint flags, guint info);
extern void gtk_target_list_add_image_targets(GtkTargetList * list,
guint,
gboolean);
extern void gtk_target_list_add_rich_text_targets(GtkTargetList *
list,
guint, gboolean,
GtkTextBuffer *);
extern void gtk_target_list_add_table(GtkTargetList * list,
const GtkTargetEntry *, guint);
extern void gtk_target_list_add_text_targets(GtkTargetList * list,
guint);
extern void gtk_target_list_add_uri_targets(GtkTargetList * list,
guint);
extern gboolean gtk_target_list_find(GtkTargetList * list, GdkAtom
target,
guint * info);
extern GType gtk_target_list_get_type(void);
extern GtkTargetList *gtk_target_list_new(const GtkTargetEntry *
targets,
guint);
extern GtkTargetList *gtk_target_list_ref(GtkTargetList * list);
extern void gtk_target_list_remove(GtkTargetList * list, GdkAtom
target);
extern void gtk_target_list_unref(GtkTargetList * list);
extern void gtk_target_table_free(GtkTargetEntry * targets, gint);
extern GtkTargetEntry
*gtk_target_table_new_from_list(GtkTargetList * list,
gint *);
extern gboolean gtk_targets_include_image(GdkAtom * targets, gint,
gboolean);
extern gboolean gtk_targets_include_rich_text(GdkAtom * targets,
gint,
GtkTextBuffer *);
extern gboolean gtk_targets_include_text(GdkAtom * targets, gint);
extern gboolean gtk_targets_include_uri(GdkAtom * targets, gint);
extern GtkWidget *gtk_test_create_simple_window(const char
>window_title,
const char *);
extern GtkWidget *gtk_test_create_widget(GType widget_type, const
char *,
...);
extern GtkWidget *gtk_test_display_button_window(const char
>window_title,
const char *, ...);
extern GtkWidget *gtk_test_find_label(GtkWidget * widget, const
char *);
extern GtkWidget *gtk_test_find_sibling(GtkWidget * base_widget,
GType);
extern GtkWidget *gtk_test_find_widget(GtkWidget * widget, const
char *,
GType);
extern void gtk_test_init(int *argc, char ***, ...);
extern const unsigned long int *gtk_test_list_all_types(guint *
n_types);
extern void gtk_test_register_all_types(void);
extern double gtk_test_slider_get_value(GtkWidget * widget);
extern void gtk_test_slider_set_perc(GtkWidget * widget, double);
extern gboolean gtk_test_spin_button_click(GtkSpinButton * spinner,
guint,

```

```

        gboolean);
extern gchar *gtk_test_text_get(GtkWidget * widget);
extern void gtk_test_text_set(GtkWidget * widget, const char *);
extern gboolean gtk_test_widget_click(GtkWidget * widget, guint
button,
        GdkModifierType modifiers);
extern gboolean gtk_test_widget_send_key(GtkWidget * widget, guint
keyval,
        GdkModifierType modifiers);
extern
        GtkTextAttributes
*gtk_test_attributes_copy(GtkTextAttributes *
        src);
extern void gtk_test_attributes_copy_values(GtkTextAttributes *
src,
        GtkTextAttributes *);
extern GType gtk_test_attributes_get_type(void);
extern GtkTextAttributes *gtk_test_attributes_new(void);
extern
        GtkTextAttributes
*gtk_test_attributes_ref(GtkTextAttributes *
        values);
extern void gtk_test_attributes_unref(GtkTextAttributes * values);
extern void gtk_test_buffer_add_mark(GtkTextBuffer * buffer,
GtkTextMark *,
        const GtkTextIter *);
extern void gtk_test_buffer_add_selection_clipboard(GtkTextBuffer
* buffer,
        GtkClipboard *);
extern void gtk_test_buffer_apply_tag(GtkTextBuffer * buffer,
GtkTextTag *,
        const GtkTextIter *,
        const GtkTextIter *);
extern void gtk_test_buffer_apply_tag_by_name(GtkTextBuffer *
buffer,
        const char *,
        const GtkTextIter *,
        const GtkTextIter *);
extern gboolean gtk_test_buffer_backspace(GtkTextBuffer * buffer,
GtkTextIter *, gboolean,
        gboolean);
extern void gtk_test_buffer_begin_user_action(GtkTextBuffer *
buffer);
extern void gtk_test_buffer_copy_clipboard(GtkTextBuffer * buffer,
GtkClipboard *);
extern GtkTextChildAnchor
*gtk_test_buffer_create_child_anchor(GtkTextBuffer * buffer,
        GtkTextIter *);
extern GtkTextMark *gtk_test_buffer_create_mark(GtkTextBuffer *
buffer,
        const char *,
        const GtkTextIter *,
        gboolean);
extern GtkTextTag *gtk_test_buffer_create_tag(GtkTextBuffer *
buffer,
        const char *, const char *,
        ...);
extern void gtk_test_buffer_cut_clipboard(GtkTextBuffer * buffer,
GtkClipboard *, gboolean);
extern void gtk_test_buffer_delete(GtkTextBuffer * buffer,
GtkTextIter *,
        GtkTextIter *);
extern gboolean gtk_test_buffer_delete_interactive(GtkTextBuffer *
buffer,
        GtkTextIter *,
        GtkTextIter *,
        gboolean);
extern void gtk_test_buffer_delete_mark(GtkTextBuffer * buffer,

```

```

        GtkTextMark *);
extern void gtk_text_buffer_delete_mark_by_name(GtkTextBuffer *
buffer,
        const char *);
extern gboolean gtk_text_buffer_delete_selection(GtkTextBuffer *
buffer,
        gboolean, gboolean);
extern gboolean gtk_text_buffer_deserialize(GtkTextBuffer *
        register_buffer,
        GtkTextBuffer *, GdkAtom,
        GtkTextIter *,
        const unsigned char *, gsize,
        GError * *);
extern gboolean
gtk_text_buffer_deserialize_get_can_create_tags(GtkTextBuffer *
buffer,
        GdkAtom format);
extern void
gtk_text_buffer_deserialize_set_can_create_tags(GtkTextBuffer *
        buffer,
        GdkAtom format,
        gboolean

can_create_tags);
extern void gtk_text_buffer_end_user_action(GtkTextBuffer *
buffer);
extern void gtk_text_buffer_get_bounds(GtkTextBuffer * buffer,
        GtkTextIter *, GtkTextIter *);
extern gint gtk_text_buffer_get_char_count(GtkTextBuffer * buffer);
extern GtkTargetList
*gtk_text_buffer_get_copy_target_list(GtkTextBuffer *
        buffer);
extern GdkAtom
*gtk_text_buffer_get_deserialize_formats(GtkTextBuffer *
        buffer, gint *);
extern void gtk_text_buffer_get_end_iter(GtkTextBuffer * buffer,
        GtkTextIter *);
extern gboolean gtk_text_buffer_get_has_selection(GtkTextBuffer *
buffer);
extern GtkTextMark *gtk_text_buffer_get_insert(GtkTextBuffer *
buffer);
extern void gtk_text_buffer_get_iter_at_child_anchor(GtkTextBuffer
*
        buffer, GtkTextIter *,
        GtkTextChildAnchor *);
extern void gtk_text_buffer_get_iter_at_line(GtkTextBuffer *
buffer,
        GtkTextIter *, gint);
extern void gtk_text_buffer_get_iter_at_line_index(GtkTextBuffer *
buffer,
        GtkTextIter *, gint,
        gint);
extern void gtk_text_buffer_get_iter_at_line_offset(GtkTextBuffer
* buffer,
        GtkTextIter *, gint,
        gint);
extern void gtk_text_buffer_get_iter_at_mark(GtkTextBuffer *
buffer,
        GtkTextIter *, GtkTextMark *);
extern void gtk_text_buffer_get_iter_at_offset(GtkTextBuffer *
buffer,
        GtkTextIter *, gint);
extern gint gtk_text_buffer_get_line_count(GtkTextBuffer * buffer);
extern GtkTextMark *gtk_text_buffer_get_mark(GtkTextBuffer *
buffer,
        const char *);

```

```

extern gboolean gtk_text_buffer_get_modified(GtkTextBuffer *
buffer);
extern GtkTargetList
*gtk_text_buffer_get_paste_target_list(GtkTextBuffer *
buffer);
extern GtkTextMark
*gtk_text_buffer_get_selection_bound(GtkTextBuffer *
buffer);
extern gboolean gtk_text_buffer_get_selection_bounds(GtkTextBuffer
*
buffer, GtkTextIter *,
GtkTextIter *);
extern GdkAtom
*gtk_text_buffer_get_serialize_formats(GtkTextBuffer *
buffer, gint *);
extern gchar *gtk_text_buffer_get_slice(GtkTextBuffer * buffer,
const GtkTextIter *,
const GtkTextIter *, gboolean);
extern void gtk_text_buffer_get_start_iter(GtkTextBuffer * buffer,
GtkTextIter *);
extern GtkTextTagTable
*gtk_text_buffer_get_tag_table(GtkTextBuffer *
buffer);
extern gchar *gtk_text_buffer_get_text(GtkTextBuffer * buffer,
const GtkTextIter *,
const GtkTextIter *, gboolean);
extern GType gtk_text_buffer_get_type(void);
extern void gtk_text_buffer_insert(GtkTextBuffer * buffer,
GtkTextIter *,
const char *, gint);
extern void gtk_text_buffer_insert_at_cursor(GtkTextBuffer *
buffer,
const char *, gint);
extern void gtk_text_buffer_insert_child_anchor(GtkTextBuffer *
buffer,
GtkTextIter *,
GtkTextChildAnchor *);
extern gboolean gtk_text_buffer_insert_interactive(GtkTextBuffer *
buffer,
GtkTextIter *,
const char *, gint,
gboolean);
extern gboolean
gtk_text_buffer_insert_interactive_at_cursor(GtkTextBuffer
* buffer,
const char *,
gint,
gboolean);
extern void gtk_text_buffer_insert_pixbuf(GtkTextBuffer * buffer,
GtkTextIter *, GdkPixbuf *);
extern void gtk_text_buffer_insert_range(GtkTextBuffer * buffer,
GtkTextIter *,
const GtkTextIter *,
const GtkTextIter *);
extern gboolean
gtk_text_buffer_insert_range_interactive(GtkTextBuffer *
buffer,
GtkTextIter *,
const GtkTextIter
*,
const GtkTextIter
*, gboolean);
extern void gtk_text_buffer_insert_with_tags(GtkTextBuffer *
buffer,
GtkTextIter *, const char *,
gint, GtkTextTag *, ...);

```

```

extern void gtk_text_buffer_insert_with_tags_by_name(GtkTextBuffer
*
buffer, GtkTextIter *,
const char *, gint,
const char *, ...);
extern void gtk_text_buffer_move_mark(GtkTextBuffer * buffer,
GtkTextMark *, const GtkTextIter *);
extern void gtk_text_buffer_move_mark_by_name(GtkTextBuffer *
buffer,
const char *,
const GtkTextIter *);
extern GtkTextBuffer *gtk_text_buffer_new(GtkTextTagTable * table);
extern void gtk_text_buffer_paste_clipboard(GtkTextBuffer * buffer,
GtkClipboard *, GtkTextIter *,
gboolean);
extern void gtk_text_buffer_place_cursor(GtkTextBuffer * buffer,
const GtkTextIter *);
extern void gtk_text_buffer_register_deserialize_format(GtkTextBuffer *
buffer,
const char *,
GtkTextBufferDeserializeFunc,
gpointer,
GDestroyNotify);
extern void gtk_text_buffer_register_deserialize_tagset(GtkTextBuffer *
buffer,
const char *,
GdkAtom);
extern void gtk_text_buffer_register_serialize_format(GtkTextBuffer *
buffer,
const char *,
GtkTextBufferSerializeFunc,
gpointer,
GDestroyNotify);
extern void gtk_text_buffer_register_serialize_tagset(GtkTextBuffer *
buffer,
const char *,
GdkAtom);
extern void gtk_text_buffer_remove_all_tags(GtkTextBuffer * buffer,
const GtkTextIter *,
const GtkTextIter *);
extern void gtk_text_buffer_remove_selection_clipboard(GtkTextBuffer *
buffer,
GtkClipboard *);
extern void gtk_text_buffer_remove_tag(GtkTextBuffer * buffer,
GtkTextTag *, const GtkTextIter *,
const GtkTextIter *);
extern void gtk_text_buffer_remove_tag_by_name(GtkTextBuffer *
buffer,
const char *,
const GtkTextIter *,
const GtkTextIter *);
extern void gtk_text_buffer_select_range(GtkTextBuffer * buffer,
const GtkTextIter *,
const GtkTextIter *);
extern void gtk_text_buffer_serialize(GtkTextBuffer *
register_buffer,
GtkTextBuffer *, GdkAtom,
const GtkTextIter *,
const GtkTextIter *, gsize *);
extern void gtk_text_buffer_set_modified(GtkTextBuffer * buffer,
gboolean);

```

```

extern void gtk_text_buffer_set_text(GtkTextBuffer * buffer, const
char *,
                                gint);
extern GType gtk_text_buffer_target_info_get_type(void);
extern
gtk_text_buffer_unregister_deserialize_format(GtkTextBuffer *
                                buffer,
                                GdkAtom format);
extern
gtk_text_buffer_unregister_serialize_format(GtkTextBuffer *
                                buffer,
                                GdkAtom format);
extern
gtk_text_child_anchor_get_deleted(GtkTextChildAnchor *
                                anchor);
extern GType gtk_text_child_anchor_get_type(void);
extern GList *gtk_text_child_anchor_get_widgets(GtkTextChildAnchor
*
                                anchor);
extern GtkTextChildAnchor *gtk_text_child_anchor_new(void);
extern GType gtk_text_direction_get_type(void);
extern void gtk_text_iter_assign(GtkTextIter * iter, const
GtkTextIter *);
extern gboolean gtk_text_iter_backward_char(GtkTextIter * iter);
extern gboolean gtk_text_iter_backward_chars(GtkTextIter * iter,
gint);
extern gboolean gtk_text_iter_backward_cursor_position(GtkTextIter
* iter);
extern
gtk_text_iter_backward_cursor_positions(GtkTextIter * iter,
                                gint);
extern gboolean gtk_text_iter_backward_find_char(GtkTextIter *
iter,
                                GtkTextCharPredicate,
                                gpointer,
                                const GtkTextIter *);
extern gboolean gtk_text_iter_backward_line(GtkTextIter * iter);
extern gboolean gtk_text_iter_backward_lines(GtkTextIter * iter,
gint);
extern gboolean gtk_text_iter_backward_search(const GtkTextIter *
iter,
                                const char *,
                                GtkTextSearchFlags,
                                GtkTextIter *, GtkTextIter *,
                                const GtkTextIter *);
extern gboolean gtk_text_iter_backward_sentence_start(GtkTextIter
* iter);
extern gboolean gtk_text_iter_backward_sentence_starts(GtkTextIter
* iter,
                                gint);
extern gboolean gtk_text_iter_backward_to_tag_toggle(GtkTextIter *
iter,
                                GtkTextTag *);
extern
gtk_text_iter_backward_visible_cursor_position(GtkTextIter
* iter);
extern
gtk_text_iter_backward_visible_cursor_positions(GtkTextIter
* iter,
                                gint);
extern gboolean gtk_text_iter_backward_visible_line(GtkTextIter *
iter);
extern gboolean gtk_text_iter_backward_visible_lines(GtkTextIter *
iter,
                                gint);

```

```

extern                                     gboolean
gtk_text_iter_backward_visible_word_start(GtkTextIter *
                                           iter);

extern                                     gboolean
gtk_text_iter_backward_visible_word_starts(GtkTextIter *
                                           iter, gint);

extern  gboolean  gtk_text_iter_backward_word_start(GtkTextIter *
iter);
extern  gboolean  gtk_text_iter_backward_word_starts(GtkTextIter *
iter,
                                           gint);

extern gboolean gtk_text_iter_begins_tag(const GtkTextIter * iter,
                                         GtkTextTag *);
extern gboolean gtk_text_iter_can_insert(const GtkTextIter * iter,
                                         gboolean);
extern gint gtk_text_iter_compare(const GtkTextIter * lhs,
                                  const GtkTextIter *);
extern GtkTextIter *gtk_text_iter_copy(const GtkTextIter * iter);
extern gboolean gtk_text_iter_editable(const GtkTextIter * iter,
gboolean);
extern gboolean gtk_text_iter_ends_line(const GtkTextIter * iter);
extern gboolean gtk_text_iter_ends_sentence(const GtkTextIter *
iter);
extern gboolean gtk_text_iter_ends_tag(const GtkTextIter * iter,
                                       GtkTextTag *);
extern gboolean gtk_text_iter_ends_word(const GtkTextIter * iter);
extern gboolean gtk_text_iter_equal(const GtkTextIter * lhs,
                                    const GtkTextIter *);
extern gboolean gtk_text_iter_forward_char(GtkTextIter * iter);
extern gboolean gtk_text_iter_forward_chars(GtkTextIter * iter,
gint);
extern gboolean gtk_text_iter_forward_cursor_position(GtkTextIter
* iter);
extern gboolean gtk_text_iter_forward_cursor_positions(GtkTextIter
* iter,
                                           gint);
extern gboolean gtk_text_iter_forward_find_char(GtkTextIter * iter,
                                                GtkTextCharPredicate,
                                                gpointer,
                                                const GtkTextIter *);
extern gboolean gtk_text_iter_forward_line(GtkTextIter * iter);
extern gboolean gtk_text_iter_forward_lines(GtkTextIter * iter,
gint);
extern gboolean gtk_text_iter_forward_search(const GtkTextIter *
iter,
                                           const char *,
                                           GtkTextSearchFlags,
                                           GtkTextIter *, GtkTextIter *,
                                           const GtkTextIter *);
extern gboolean gtk_text_iter_forward_sentence_end(GtkTextIter *
iter);
extern gboolean gtk_text_iter_forward_sentence_ends(GtkTextIter *
iter,
                                           gint);
extern void gtk_text_iter_forward_to_end(GtkTextIter * iter);
extern gboolean gtk_text_iter_forward_to_line_end(GtkTextIter *
iter);
extern gboolean gtk_text_iter_forward_to_tag_toggle(GtkTextIter *
iter,
                                           GtkTextTag *);
extern                                     gboolean
gtk_text_iter_forward_visible_cursor_position(GtkTextIter *
iter);
extern                                     gboolean
gtk_text_iter_forward_visible_cursor_positions(GtkTextIter
* iter,

```

```

                                gint);
extern gboolean gtk_text_iter_forward_visible_line(GtkTextIter *
iter);
extern gboolean gtk_text_iter_forward_visible_lines(GtkTextIter *
iter,
                                gint);
extern gboolean gtk_text_iter_forward_visible_word_end(GtkTextIter
* iter);
extern
                                gboolean
gtk_text_iter_forward_visible_word_ends(GtkTextIter * iter,
                                gint);
extern gboolean gtk_text_iter_forward_word_end(GtkTextIter * iter);
extern gboolean gtk_text_iter_forward_word_ends(GtkTextIter * iter,
gint);
extern void gtk_text_iter_free(GtkTextIter * iter);
extern gboolean gtk_text_iter_get_attributes(const GtkTextIter *
iter,
                                GtkTextAttributes *);
extern GtkTextBuffer *gtk_text_iter_get_buffer(const GtkTextIter *
iter);
extern gint gtk_text_iter_get_bytes_in_line(const GtkTextIter *
iter);
extern gunichar gtk_text_iter_get_char(const GtkTextIter * iter);
extern gint gtk_text_iter_get_chars_in_line(const GtkTextIter *
iter);
extern GtkTextChildAnchor *gtk_text_iter_get_child_anchor(const
GtkTextIter
                                * iter);
extern PangoLanguage *gtk_text_iter_get_language(const GtkTextIter
* iter);
extern gint gtk_text_iter_get_line(const GtkTextIter * iter);
extern gint gtk_text_iter_get_line_index(const GtkTextIter * iter);
extern gint gtk_text_iter_get_line_offset(const GtkTextIter * iter);
extern GSList *gtk_text_iter_get_marks(const GtkTextIter * iter);
extern gint gtk_text_iter_get_offset(const GtkTextIter * iter);
extern GdkPixbuf *gtk_text_iter_get_pixbuf(const GtkTextIter *
iter);
extern gchar *gtk_text_iter_get_slice(const GtkTextIter * start,
const GtkTextIter *);
extern GSList *gtk_text_iter_get_tags(const GtkTextIter * iter);
extern gchar *gtk_text_iter_get_text(const GtkTextIter * start,
const GtkTextIter *);
extern GSList *gtk_text_iter_get_toggled_tags(const GtkTextIter *
iter,
                                gboolean);
extern GType gtk_text_iter_get_type(void);
extern gint gtk_text_iter_get_visible_line_index(const GtkTextIter
* iter);
extern
gint
gtk_text_iter_get_visible_line_offset(const
GtkTextIter *
                                iter);
extern gchar *gtk_text_iter_get_visible_slice(const GtkTextIter *
start,
const GtkTextIter *);
extern
gchar
*gtk_text_iter_get_visible_text(const GtkTextIter *
start,
const GtkTextIter *);
extern gboolean gtk_text_iter_has_tag(const GtkTextIter * iter,
GtkTextTag *);
extern gboolean gtk_text_iter_in_range(const GtkTextIter * iter,
const GtkTextIter *,
const GtkTextIter *);
extern gboolean gtk_text_iter_inside_sentence(const GtkTextIter *
iter);
extern gboolean gtk_text_iter_inside_word(const GtkTextIter * iter);

```



```

extern gboolean gtk_text_iter_is_cursor_position(const GtkTextIter
* iter);
extern gboolean gtk_text_iter_is_end(const GtkTextIter * iter);
extern gboolean gtk_text_iter_is_start(const GtkTextIter * iter);
extern void gtk_text_iter_order(GtkTextIter * first, GtkTextIter
*);
extern void gtk_text_iter_set_line(GtkTextIter * iter, gint);
extern void gtk_text_iter_set_line_index(GtkTextIter * iter, gint);
extern void gtk_text_iter_set_line_offset(GtkTextIter * iter, gint);
extern void gtk_text_iter_set_offset(GtkTextIter * iter, gint);
extern void gtk_text_iter_set_visible_line_index(GtkTextIter *
iter, gint);
extern void gtk_text_iter_set_visible_line_offset(GtkTextIter *
iter,
gint);
extern gboolean gtk_text_iter_starts_line(const GtkTextIter * iter);
extern gboolean gtk_text_iter_starts_sentence(const GtkTextIter *
iter);
extern gboolean gtk_text_iter_starts_word(const GtkTextIter * iter);
extern gboolean gtk_text_iter_toggles_tag(const GtkTextIter * iter,
GtkTextTag *);
extern GtkTextBuffer *gtk_text_mark_get_buffer(GtkTextMark * mark);
extern gboolean gtk_text_mark_get_deleted(GtkTextMark * mark);
extern gboolean gtk_text_mark_get_left_gravity(GtkTextMark * mark);
extern const char *gtk_text_mark_get_name(GtkTextMark * mark);
extern GType gtk_text_mark_get_type(void);
extern gboolean gtk_text_mark_get_visible(GtkTextMark * mark);
extern GtkTextMark *gtk_text_mark_new(const char *name, gboolean);
extern void gtk_text_mark_set_visible(GtkTextMark * mark, gboolean);
extern GType gtk_text_search_flags_get_type(void);
extern gboolean gtk_text_tag_event(GtkTextTag * tag, GObject *,
GdkEvent *,
const GtkTextIter *);
extern gint gtk_text_tag_get_priority(GtkTextTag * tag);
extern GType gtk_text_tag_get_type(void);
extern GtkTextTag *gtk_text_tag_new(const char *name);
extern void gtk_text_tag_set_priority(GtkTextTag * tag, gint);
extern void gtk_text_tag_table_add(GtkTextTagTable * table,
GtkTextTag *);
extern void gtk_text_tag_table_foreach(GtkTextTagTable * table,
GtkTextTagTableForeach, gpointer);
extern gint gtk_text_tag_table_get_size(GtkTextTagTable * table);
extern GType gtk_text_tag_table_get_type(void);
extern GtkTextTag *gtk_text_tag_table_lookup(GtkTextTagTable *
table,
const char *);
extern GtkTextTagTable *gtk_text_tag_table_new(void);
extern void gtk_text_tag_table_remove(GtkTextTagTable * table,
GtkTextTag *);
extern void gtk_text_view_add_child_at_anchor(GtkTextView *
text_view,
GtkWidget *,
GtkTextChildAnchor *);
extern void gtk_text_view_add_child_in_window(GtkTextView *
text_view,
GtkWidget *,
GtkTextWindowType, gint,
gint);
extern gboolean gtk_text_view_backward_display_line(GtkTextView *
text_view,
GtkTextIter *);
extern
gboolean
gtk_text_view_backward_display_line_start(GtkTextView *
text_view,
GtkTextIter *);

```

```

extern void gtk_text_view_buffer_to_window_coords(GtkTextView *
text_view,
                                                GtkTextWindowType, gint,
                                                gint, gint *, gint *);
extern gboolean gtk_text_view_forward_display_line(GtkTextView *
text_view,
                                                GtkTextIter *);
extern gboolean gtk_text_view_forward_display_line_end(GtkTextView
*
text_view,
                                                GtkTextIter *);
extern gboolean gtk_text_view_get_accepts_tab(GtkTextView *
text_view);
extern gint gtk_text_view_get_border_window_size(GtkTextView *
text_view,
                                                GtkTextWindowType);
extern GtkTextBuffer *gtk_text_view_get_buffer(GtkTextView *
text_view);
extern void gtk_text_view_get_cursor_locations(GtkTextView *
text_view,
                                                const GtkTextIter *,
                                                GdkRectangle *,
                                                GdkRectangle *);
extern gboolean gtk_text_view_get_cursor_visible(GtkTextView *
text_view);
extern GtkTextAttributes *gtk_text_view_get_default_attributes(GtkTextView
*
text_view);
extern gboolean gtk_text_view_get_editable(GtkTextView *
text_view);
extern GtkAdjustment *gtk_text_view_get_hadjustment(GtkTextView *
text_view);
extern gint gtk_text_view_get_indent(GtkTextView * text_view);
extern GtkInputHints gtk_text_view_get_input_hints(GtkTextView *
text_view);
extern GtkInputPurpose gtk_text_view_get_input_purpose(GtkTextView
*
text_view);
extern void gtk_text_view_get_iter_at_location(GtkTextView *
text_view,
                                                GtkTextIter *, gint, gint);
extern void gtk_text_view_get_iter_at_position(GtkTextView *
text_view,
                                                GtkTextIter *, gint *, gint,
                                                gint);
extern void gtk_text_view_get_iter_location(GtkTextView *
text_view,
                                                const GtkTextIter *,
                                                GdkRectangle *);
extern GtkJustification gtk_text_view_get_justification(GtkTextView *
text_view);
extern gint gtk_text_view_get_left_margin(GtkTextView * text_view);
extern void gtk_text_view_get_line_at_y(GtkTextView * text_view,
                                                GtkTextIter *, gint, gint *);
extern void gtk_text_view_get_line_yrange(GtkTextView * text_view,
                                                const GtkTextIter *, gint *,
                                                gint *);
extern gboolean gtk_text_view_get_overwrite(GtkTextView *
text_view);
extern gint gtk_text_view_get_pixels_above_lines(GtkTextView *
text_view);
extern gint gtk_text_view_get_pixels_below_lines(GtkTextView *
text_view);

```

```

extern gint gtk_text_view_get_pixels_inside_wrap(GtkTextView *
text_view);
extern gint gtk_text_view_get_right_margin(GtkTextView *
text_view);
extern PangoTabArray *gtk_text_view_get_tabs(GtkTextView *
text_view);
extern GType gtk_text_view_get_type(void);
extern GtkAdjustment *gtk_text_view_get_vadjustment(GtkTextView *
text_view);
extern void gtk_text_view_get_visible_rect(GtkTextView * text_view,
GdkRectangle *);
extern GdkWindow *gtk_text_view_get_window(GtkTextView * text_view,
GtkTextWindowType);
extern GtkTextWindowType gtk_text_view_get_window_type(GtkTextView
*
text_view,
GdkWindow *);
extern GtkWrapMode gtk_text_view_get_wrap_mode(GtkTextView *
text_view);
extern gboolean
gtk_text_view_im_context_filter_keypress(GtkTextView *
text_view,
GdkEventKey *);
extern void gtk_text_view_move_child(GtkTextView * text_view,
GtkWidget *,
gint, gint);
extern gboolean gtk_text_view_move_mark_onscreen(GtkTextView *
text_view,
GtkTextMark *);
extern gboolean gtk_text_view_move_visually(GtkTextView *
text_view,
GtkTextIter *, gint);
extern GtkWidget *gtk_text_view_new(void);
extern GtkWidget *gtk_text_view_new_with_buffer(GtkTextBuffer *
buffer);
extern gboolean gtk_text_view_place_cursor_onscreen(GtkTextView *
text_view);
extern void gtk_text_view_reset_im_context(GtkTextView *
text_view);
extern void gtk_text_view_scroll_mark_onscreen(GtkTextView *
text_view,
GtkTextMark *);
extern gboolean gtk_text_view_scroll_to_iter(GtkTextView *
text_view,
GtkTextIter *, gdouble,
gboolean, gdouble, gdouble);
extern void gtk_text_view_scroll_to_mark(GtkTextView * text_view,
GtkTextMark *, gdouble, gboolean,
gdouble, gdouble);
extern void gtk_text_view_set_accepts_tab(GtkTextView * text_view,
gboolean);
extern void gtk_text_view_set_border_window_size(GtkTextView *
text_view,
GtkTextWindowType, gint);
extern void gtk_text_view_set_buffer(GtkTextView * text_view,
GtkTextBuffer *);
extern void gtk_text_view_set_cursor_visible(GtkTextView *
text_view,
gboolean);
extern void gtk_text_view_set_editable(GtkTextView * text_view,
gboolean);
extern void gtk_text_view_set_indent(GtkTextView * text_view, gint);
extern void gtk_text_view_set_input_hints(GtkTextView * text_view,
GtkInputHints);
extern void gtk_text_view_set_input_purpose(GtkTextView *
text_view,

```

```

                                GtkInputPurpose);
extern void gtk_text_view_set_justification(GtkTextView *
text_view,
                                GtkJustification);
extern void gtk_text_view_set_left_margin(GtkTextView * text_view,
gint);
extern void gtk_text_view_set_overwrite(GtkTextView * text_view,
gboolean);
extern void gtk_text_view_set_pixels_above_lines(GtkTextView *
text_view,
                                gint);
extern void gtk_text_view_set_pixels_below_lines(GtkTextView *
text_view,
                                gint);
extern void gtk_text_view_set_pixels_inside_wrap(GtkTextView *
text_view,
                                gint);
extern void gtk_text_view_set_right_margin(GtkTextView * text_view,
gint);
extern void gtk_text_view_set_tabs(GtkTextView * text_view,
PangoTabArray *);
extern void gtk_text_view_set_wrap_mode(GtkTextView * text_view,
GtkWrapMode);
extern gboolean gtk_text_view_starts_display_line(GtkTextView *
text_view,
                                const GtkTextIter *);
extern void gtk_text_view_window_to_buffer_coords(GtkTextView *
text_view,
                                GtkTextWindowType, gint,
                                gint, gint *, gint *);
extern GType gtk_text_window_type_get_type(void);
extern void gtk_theming_engine_get(GtkThemingEngine * engine,
GtkStateFlags, ...);
extern void gtk_theming_engine_get_background_color(GtkThemingEngine *
engine, GtkStateFlags,
GdkRGBA *);
extern void gtk_theming_engine_get_border(GtkThemingEngine *
engine,
                                GtkStateFlags, GtkBorder *);
extern void gtk_theming_engine_get_border_color(GtkThemingEngine *
engine,
                                GtkStateFlags, GdkRGBA *);
extern void gtk_theming_engine_get_color(GtkThemingEngine * engine,
GtkStateFlags, GdkRGBA *);
extern void gtk_theming_engine_get_direction(GtkThemingEngine *
engine);
extern const PangoFontDescription
*gtk_theming_engine_get_font(GtkThemingEngine * engine,
GtkStateFlags);
extern GtkJunctionSides
gtk_theming_engine_get_junction_sides(GtkThemingEngine * engine);
extern void gtk_theming_engine_get_margin(GtkThemingEngine *
engine,
                                GtkStateFlags, GtkBorder *);
extern void gtk_theming_engine_get_padding(GtkThemingEngine *
engine,
                                GtkStateFlags, GtkBorder *);
extern const GtkWidgetPath
*gtk_theming_engine_get_path(GtkThemingEngine *
engine);
extern void gtk_theming_engine_get_property(GtkThemingEngine *
engine,
                                const char *, GtkStateFlags,
                                GValue *);

```