```
      button
prop      : text[S]  bgcolor[C] icon[N] toolbar_style
common    : iconpos font frame justification padding
signal    : click{}
command   : set_text(const char* text)

      check_box
prop      : text[S] plus_style
common    : iconpos frame padding font justification
signal    : click{}
command   : set_value(int V)
command   : get_value(int *V)

      dialog
prop      : title[S] pos_x[I] pos_y[I] default_width[I] default_height[I]
common    : decorations padding spacing
signal    : close{}
command   : execute()
command   : close()
      When creating the dialog, you use a function similar to
      those used for main windows:
                gui_NAME_t *K= gui_c_NAME( MAIN_WINDOW->self );
      After this, you simply execute it.
                gui_execute(*K);
      When you're done with it, you can close it by calling:
                gui_close(*K);

      dir_list
prop      : show_hidden show_files
common    : font frame
signal    : click{}
command   : set_current_file(S path)
command   : set_current_dir(S path)
command   : get_current(R, RS path, int *type)
            type is one of "FDLCBPS" File Directory Link Chardev
            Blockdev Pipe Socket

      file_list
      This widget can't navigate the file system on its own. You
      need to write signal handlers for changing directories etc.
prop      : show_hidden
prop      : no_parent
            Hides directories "." and ".." from view.
common    : font frame
signal    : click_file{clicked_path[S] clicked_name[S]}
signal    : click_dir{clicked_path[S] clicked_name[S]}
signal    : dclick_file{clicked_path[S] clicked_name[S]}
signal    : dclick_dir{clicked_path[S] clicked_name[S]}
command   : get_current(int *Rresult, const char **Rname, int *Rtype)
command   : set_current_file(const char *fn)
command   : set_directory(const char *fn)
command   : get_directory(char **fn)

      float_spinner
prop      : min[F] max[F] increment[F] ncols[I]
common    : frame padding
command   : get_value(double* RV)
command   : set_value(double V)

      group_box
prop      : title[S]
common    : spacing padding

      hbox
common    : spacing padding frame packer

      int_list
common    : font
command   : clear_selection()
command   : get_selected(int *result,int *key)
```

```
command   : select_item(int key, int sel)
command   : append(const char *text,int key)
command   : prepend(const char *text,int key)
command   : insert(int poskey, const char *text,int key)
command   : remove(int key)
command   : clear()

      label
prop      : text[S] icon[N]
common    : iconpos frame justification padding font
command   : set_text(const char *text)

      list
prop      : key[int/double/string/ptr]
common    : font
signal    : click{clicked_item[I] clicked_key[int/char*/double/void*]}
signal    : dclick{clicked_item[I] clicked_key[int/char*/double/void*]}
            clicked_item is the index of the item in the list

      main_window
prop      : class_name[D] title[S] pos_x[I] pos_y[I]
prop      : default_width[I] default_height[I]
common    : decorations padding spacing
signal    : close{}

      message_console
prop      : word_wrap no_wrap fixed_wrap text[S] no_vertical_scroller
common    : font frame justification padding
command   : print(const char *fmt, ...)
command   : append(const char* str, size_t len)
            len must be given
command   : clear()

      radio_group
      This widget only has the objid property.
command   : set_value(int V)
command   : get_value(int *R)

      radio_button
prop      : text[S] group[D] value[I] default
common    : iconpos frame justification padding font
signal    : click{}

      separator
prop      : vertical horizontal groove ridge line
common    : padding

      spinner
prop      : max[I] min[I] increment[I] ncols[I]
common    : padding frame
command   : get_value(int* RV)
command   : set_value(int V)

      tab_book
      This widget is just a container for tabs. It only accepts objid
      and layout keywords. As child widgets, you can use only the 'tab'
      widget.

      tab
      Only one child can be given for this widget
prop     : title[S]

      text_field
prop     : ncols[I] password_mode integer_mode real_mode
prop     : limit_length notify_enter_only justification frame
signal   : change{text[S]} text can be null
command  : set_text(const char *text)
command  : get_text(const char **text)

      text_viewer
```

```
prop     : text[S] fixed_wrap no_wrap word_wrap no_vertical_scroller
command  : set_text(const char *text)
command  : clear()

      toggle_button
prop     : bgcolor[C] text_on[S] text_off[S] icon_on[N] icon_off[N]
prop     : toolbar_style
common   : iconpos frame justification padding font
signal   : click{}

      vbox
common   : spacing padding frame packer
```

```
decorations
   decorations {border close maximize menu minimize
                resize shrinkable stretchable  title all none}
   no_close no_maximize etc.
   {all no_menu} = everything but menu.
   {none close} = only close.
   resize = shrinkable + stretchable.
font
   font { family[S] size[I] }
frame
    frame_none frame_sunken frame_raised frame_thick frame_ridge frame_line
iconpos
    icon_before_text icon_after_text icon_under_text icon_above_text
justification
    center_x left right hz_apart center_y top bottom vt_apart
layout
    pack_left pack_right  pack_top pack_bottom
    stick_left stick_right stick_top stick_bottom
    center_x center_y fill_x fill_y fixed_width[I] fixed_height[I]
packer
    uniform_width uniform_height
padding
    pad_right[I] pad_left[I] pad_top[I] pad_bottom[I]
spacing
    vt_spacing[I] hz_spacing[I]
```

```
define_widget {
  name ident
  is_container
  foxclass ident
  property ident type
  enum target { ident1 {value1} ident2 {value2} .. }
  library_code { code }
  widget_code { code }
  signal ident
  command ident { args } { command_code }
  common_properties { common common .. }
}

type
    int • str • ident • flag • color • font • icon
common
    iconpos • frame •justification •spacing• packer • padding• layout• objid
args
    is a list of C++ function arguments, can be empty.
command_code
    C++ code, widget is declared as $foxclass* widget.
target
    is the name of the target variable in the template without the $.
```