Configuration

With a basic knowledge of Qt, you could change the configuration of the default Kvantum theme by following these steps:

(1) Create the folder "*~/.config/Kvantum/*" (~ is your home);

(2) Create the file "*kvantum.kvconfig*" in the above folder with this line in it:

theme=***MY_THEME***

Here, ***MY_THEME*** could be any name you choose for the new configuration;

(3) Create the folder "*~/.config/Kvantum/**MY_THEME**/*" and the file "***MY_THEME.****kvconfig*" in it;

(4) In the file "***MY_THEME.****kvconfig*", you could write down your own configuration. The easiest way is to copy/paste the contents of "*style/themeconfig/default.kvconfig*" (from the source) to it and change the values of some variables. Please note that deleting a variable often means that its value will be taken from the default configuration, so that you could keep only those sections or variables you want to change. See below for a more accurate explanation.

There are many sections (groups) and variables (keys) in the config file. That's intentional: unlike most theme engines, Kvantum is supposed to be able to control virtually all aspects of widgets.

Here are the meanings of various sections:

Sections Table

Section (Group)	Meaning
[%General]	General info on the theme and some general variables.
[PanelButtonCommand]	Panel for a button used to initiate an action, for example, a push button.
[PanelButtonTool]	Panel for a tool button.
[Dock]	A dock widget.
[DockTitle]	The title of a dock widget.
[IndicatorSpinBox]	Indicators of a spin widget.
[RadioButton]	A radio button.

[CheckBox]	A check box.
[Focus]	Generic focus indicator.
[GenericFrame]	Generic frame.
[LineEdit]	A line edit (one-line text editor).
[DropDownButton]	Indicator for a drop down button, for example, a tool button that displays a menu.
[ToolboxTab]	Redundant! Kvantum uses the default look for toolboxes.
[Tab]	The tab shape within a tab bar. Also the tear indicator of a tab bar and the close button of a tab.
[TabFrame]	The frame for tab widgets.
[TabBarFrame]	The frame that is drawn for a tab bar, ususally for a tab bar that isn't part of a tab widget.
[TreeExpander]	Indicators used to represent the branch of a tree in a tree view.
[HeaderSection]	A header section. Also its label and arrow.
[SizeGrip]	Window resize handle if it exists.
[Toolbar]	A toolbar. Also its handle and separator.
[Scrollbar]	Scrollbar increase/decrease indicators (arrows).
[ScrollbarGroove]	The groove of a scrollbar.
[ScrollbarSlider]	A scrollbar slider.
[Slider]	A slider (a classic widget for controlling a bounded value).
[SliderCursor]	The handle of a slider.
[Progressbar]	The groove and label of a progressbar.
[ProgressbarContents]	The progress indicator.
[ItemView]	An item in an item view.
[Splitter]	A splitter handle.
[Widget]	A plain QWidget.
[Menu]	The panel and frame of a menu. Also its frame shadow.
[MenuItem]	A menu item in a menu. Also the tear-off section of a menu.
[MenuBar]	The empty area of a menu bar.
[MenuBarItem]	A menu bar item, like the buttons in a menubar.
[TitleBar]	A title bar, like those used in QMdiSubWindow.
[ComboBox]	A combo box and its label.
[GroupBox]	A group box and the frame around it.
[ToolTip]	The panel for a tooltip label.
[StatusBar]	The frame of a status bar.

The following table shows the variables (keys) you could change to configure the current theme – without necessarily making a new one. These are the rules for the value inheritance:

(1) If a section (group) is not present in your configuration, its variables and their values will be taken from the default config file.

(2) If a variable is not present in a section of your configuration:
(2a) First the *"inherits"* section will be searched for it and then, If nothing is found,
(2b) its value will be taken from the same section of the default config file.

There are three logical exceptions to these rules:

Exception No.1: The *"inherits"* variable will not be taken from the default config file if it is not present.

Exception No.2: If normal, focused or pressed text colors are omitted or not valid or if a section they could belong to is not present, they will be taken from the currently used color scheme.

Exception No.3: Any variable related to compositing will be neglected if omitted. For now, there are only three compositing variables, namely, *composite*, *menu_shadow_depth* and *tooltip_shadow_depth*.

Variables Table

Variable (Key)	Value	Meaning
author	string	Obvious.
comment	string	Obvious.
x11drag	true/false	Drag windows from anywhere possible? *(Only in the General section.)*
alt_mnemonic	true/false	Show underlines only when Alt is pressed? *(Only in the General section.)*
left_tabs	true/false	Align tabs to the left edge? Tabs are centered by default. *(Only in the General section.)*
joined_tabs	true/false	Join tabs together? They are detached by default. *(Only in the General section.)*
attach_active_tab	true/false	Attach the active tab to the tab widget? It is detached by default. *(Only in the General section.)*
group_toolbar_buttons	true/false	Raise and group neighbor toolbar buttons? By default, they are not raised. *(Only in the General section.)*

composite	true/false	Use compositing to have translucent menus or tooltips? It is automatically set to false if no compositing is available. Its absence also means false. *(Only in the General section.)*
menu_shadow_depth	integer	The depth of the shadow menus cast. A value of zero, its absence or a false value for *composite* means no shadow. *(Only in the General section.)*
tooltip_shadow_depth	integer	The depth of the shadow tooltips cast. A value of zero, its absence or a false value for *composite* means no shadow. *(Only in the General section.)*
splitter_width	integer	The width of splitter handles. *(Only in the General section.)*
scroll_width	integer	The width of scrollbars. The default value is 12px. *(Only in the General section.)*
check_size	integer	The width and height of checkboxes and radio buttons. The default value is 13px. *(Only in the General section.)*
inherits	string	The name of a section (in the same config file and without brackets) whose configuration is also used for this one.
frame	true/false	Draw a frame around the widget?
frame.top frame.bottom frame.left frame.right	integer	The height or width of the corresponding frame part.
frame.repeat.top.patternsize frame.repeat.bottom.patternsize frame.repeat.left.patternsize frame.repeat.right.patternsize	integer	The frame pattern sizes if a pattern is used for drawing the frame.
interior	true/false	Draw an interior for the widget?
interior.repeat.x.patternsize interior.repeat.y.patternsize	integer	The interior pattern sizes. *If you use patterns, set these to >= 50 for large areas because otherwise, CPU usage might get high.*
indicator.size	integer	Some widgets, like scrollbar arrows, have indicators. This is their size.
text.margin	true/false	Put a margin around the text?
text.margin.top text.margin.bottom text.margin.left text.margin.right	integer	The sizes of the text margins if there is any.

text.normal.color	String (#RRGGBB)	The color of the normal text as #RRGGBB or with a valid name like white, black, red, etc.
text.focus.color	String (#RRGGBB)	The color of the focused (hover) text as #RRGGBB or with a valid name like white, black, red, etc.
text.press.color	String (#RRGGBB)	The color of the pressed text as #RRGGBB or with a valid name like white, black, red, etc.
text.shadow	true/false	Draw a shadow for the text?
text.shadow.xshift text.shadow.yshift	integer	The vertical/horizontal shifts of the text shadow if it exists.
text.shadow.color	string (#RRGGBB)	The color of the text shadow as #RRGGBB or with a valid name like white, black, red, etc.
text.shadow.alpha	integer (0-255)	The opacity of the text shadow. 255 means completely opaque.
text.shadow.depth	integer	The text shadow depth.
size.minwidth size.minheight	integer	Minimum/fixed width or height of a menu/menuitem, for example.

If you want to make your own theme (see the file "Theme-Making"), you'll also need to know the meanings of these variables:

Elements Table

Variable (Key)	Value	Meaning
interior.element	string	The SVG element to be used for drawing the interior of a widget.
frame.element	string	The SVG element to be used for drawing the frame of a widget.
indicator.element	string	The SVG element to be used for drawing the indicator of a widget.

Some Examples

If you don’t want menus and tooltips to be translucent or cast shadow and want the color scheme to be used for all texts, you could use a blank configuration or a very basic one with just this in it:

[%General]

You could also be more explicit:

[%General]
composite=false

[PanelButtonCommand]
text.normal.color=none
text.focus.color=none
text.press.color=none

Here “none” is not a valid color, so text colors will be taken from the currently used color scheme.

If you want to have bigger buttons without increasing your font sizes, you could use this:

[%General]
composite=true
menu_shadow_depth=6
tooltip_shadow_depth=6

[PanelButtonCommand]
text.normal.color=white
text.focus.color=#80C0FF
text.press.color=white
text.margin.top=4
text.margin.bottom=4
text.margin.left=5
text.margin.right=5

[PanelButtonTool]
inherits=PanelButtonCommand

To have black text shadows with light green focused text, use this (black text shadows are already defined but disabled in the default config file):

[%General]
composite=true
menu_shadow_depth=6
tooltip_shadow_depth=6

[PanelButtonCommand]
text.normal.color=white

text.focus.color=lightgreen
text.press.color=white
text.shadow=true

Note that, in the two examples above, the compositing values and normal/focused/pressed text colors are also added because otherwise, they would be disabled (see the exceptions above). In the previous example, customized text colors were disabled for all widgets other than push-buttons because there were no sections for them. If you want them back, you could add sections like these:

[PanelButtonTool]
inherits=PanelButtonCommand

[Tab]
inherits=PanelButtonCommand

[MenuItem]
inherits=PanelButtonCommand

And so on.