Configuration

With a basic knowledge of Qt, you could change the configuration of the default Kvantum theme. That configuration can be easily copied by using **Kvantum Manager** (click *Save* button on its third page) or, manually, 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=***DefaultCopy***

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

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

(4) Copy/paste the contents of "*style/themeconfig/default.kvconfig"* (from the source) to the file "***DefaultCopy*** .kvconfig".

Now, you can change the values of variables (keys). 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.
[GeneralColors]	The most important colors used by the theme.
[Hacks]	Hacks for specific apps or widgets.
[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 frame.
[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]	Just text colors for tab labels of a toolbox.
[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.
[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.
[Window]	A window or dialog.
[WindowTranslucent]	This is used when a distinction is needed to be made between opaque and translucent windows. If it is omitted, the above section is used for all windows.

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 exceptions to these rules:

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

Exception No.2: If 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. Also font boldness/italicity will be ignored if omitted.

Exception No.3: Any variable related to compositing or hacking will be ignored if omitted.

Variables Table

Variable (Key)	Value	Meaning
The General Section		
author	string	Obvious.
comment	string	Obvious.
x11drag	true/false	Drag windows from anywhere possible?
alt_mnemonic	true/false	Show underlines only when Alt is pressed?

double_click	true/false	Activate view items on double clicking? They are activated on single clicking by default but the KDE setting has priority over the default.
vertical_spin_indicators	true/false	Draw spin indicators vertically and inside the spin line-edit? By default, they are drawn on adjacent buttons.
left_tabs	true/false	Align tabs to the left edge? Tabs are centered by default.
joined_tabs	true/false	Join tabs together? They are detached by default.
attach_active_tab	true/false	Attach the active tab to the tab widget? It is detached by default.
mirror_doc_tabs	true/false	By default, bottom and right tab shapes are mirror images of top and left ones, respectively. Setting this key to false will change that behavior if the tab widget is in the document mode or if the active tab is detached (i.e. the value of the key *attach_active_tab* is false), so that the top/left and bottom/ right tab shapes will be identical. The default value is true.
group_toolbar_buttons	true/false	Raise and group neighbor toolbar buttons? By default, they are not raised.
center_toolbar_handle	true/false	If true, the SVG element for the toolbar handle will be centered and its size will be that of the toolbar indicator. Otherwise, it will be scaled vertically with an 8-px width. False by default.
slim_toolbars	true/false	When true, the size of toolbar icons will be 16px if it is not set in the app. If false, the size will be determined by the DE or the app. True by default.
toolbutton_style	integer	Sets the toolbutton style when it is not set by the app. 0: follow, 1: icon only, 2: text only, 3: text beside icon, and 4: text under icon. The toolbutton style is 0 by default.
spread_progressbar	true/false	Spread progressbar's indicator across its whole groove and not just its interior? By default, the indicator is drawn inside progressbar's frame.
textless_progressbar	true/false	No percentage label for progressbars? They have percentage labels by default – unless the labels do not fit into the bars because of a small value for the following key.

progressbar_thickness	integer	If positive, it sets the (maximum) progressbar thickness. It is zero by default, which means there is no limit to progressbar thickness.
menubar_mouse_tracking	true/false	Enable mouse tracking in menubars? It is enabled by default.
merge_menubar_with_toolbar	true/false	Draw adjacent menu and tool bars as a whole? If true, the toolbar SVG interior and frame will be used for drawing them.
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.
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.
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.
translucent_windows	true/false	Translucent windows and dialogs? This requires a translucent SVG element too. A false value, its absence or a false value for *composite* means no translucency.
opaque	String list	A comma-separated list of executables, whose apps should not have window translucency. It has meaning only if *translucent_windows* is set to true.
blurring	true/false	Blur the screen area behind translucent windows in KDE? This needs KDE blur effect and also a graphic card that supports it. It has no effect when *composite* or *translucent_windows* is false.
popup_blurring	true/false	Blur the regions behind translucent menus and tooltips? This needs KDE blur effect and a graphic card that supports it. It will automatically be set to true if blurring is true.
splitter_width	integer	The width of splitter handles. It cannot be greater than 32px and is 7px by default.
scroll_width	integer	The thickness of scrollbars. It cannot be greater than 32px. The default value is 12px.

scroll_min_extent	integer	The minimum height of a vertical scrollbar slider and the minimum width of a horizontal one. It cannot be greater that 100px or less than 16px. The default value is 36px.
scroll_arrows	true/false	Draw scrollbar add-line and sub-line arrows? True by default. If set to false, it will remove scroll arrows as far as possible but some apps might still force scroll arrows.
slider_width	integer	The width of sliders. It cannot be greater than 48px and is 8px by default.
slider_handle_width slider_handle_length	integer	The width and the height of slider handles. They cannot be greater than 48px. The default values are 16px.
check_size	integer	The width and height of checkboxes and radio buttons. The default value is 13px.
small_icon_size large_icon_size button_icon_size toolbar_icon_size	integer	These affect menu-items/headers, icon-views, buttons/tabbars/listviews, and toolbars respectively. KDE setting will have priority over these values if it exists.
fill_rubberband	true/false	Always fill the rubber-band rectangle with the highlight color? By default, drop rectangles for movable toolbars and dock widgets are hollow.
The GeneralColors Section		
window.color	String (#RRGGBB)	A general background color as #RRGGBB or with a valid name like white, black, red, etc.
base.color	String (#RRGGBB)	Used mostly as the background color for text entry widgets.
alt.base.color	String (#RRGGBB)	Used as the alternate background color in views with alternating row colors.
button.color	String (#RRGGBB)	The general button background color.
light.color	String (#RRGGBB)	Lighter than *button.color* (used mostly for 3D bevels).
mid.light.color	String (#RRGGBB)	Between *button.color* and *light.color* (used mostly for 3D bevels).
dark.color	String (#RRGGBB)	Darker than *button.color* (used mostly for 3D bevels).

mid.color	String (#RRGGBB)	Between *button.color* and *dark.color* (used mostly for 3D bevels).
shadow.color	String (#RRGGBB)	A very dark color. By default, it is black. (used mostly for 3D bevels).
highlight.color	String (#RRGGBB)	A color for text selection.
inactive.highlight.color	String (#RRGGBB)	Like *highlight.color* but when the text widget does not have focus.
tooltip.base.color	String (#RRGGBB)	Tooltip background color (used in "WhatsThis" tooltips).
text.color	String (#RRGGBB)	The foreground color used with *base.color*.
window.text.color	String (#RRGGBB)	A general foreground color.
button.text.color	String (#RRGGBB)	Obvious.
disabled.text.color	String (#RRGGBB)	Obvious.
tooltip.text.color	String (#RRGGBB)	Obvious.
highlight.text.color	String (#RRGGBB)	The color of selected text.
link.color	String (#RRGGBB)	Obvious.
link.visited.color	String (#RRGGBB)	Obvious.
progress.indicator.text.color	String (#RRGGBB)	The color of that part of the progress text, which is inside the progress indicator. Useful when the progress text does not have enough contrast with the progress indicator.
The Hacks Section		
transparent_dolphin_view	true/false	No background or frame for Dolphin's view (Dolphin is the file manager of KDE)?
blur_konsole	true/false	Blur the region behind Konsole's transparent background if possible?
transparent_ktitle_label	true/false	No background for the label of KtitleWidget (a KDE widget with a heading label)?

transparent_menutitle	true/false	No background for (KDE) menu titles?
kcapacitybar_as_progressbar	true/false	Draw KCapacityBar as progressbar? KCapacityBar has its hard-coded style by default.
respect_darkness	true/false	Some apps don't respect dark themes. Fix that as far as possible?
force_size_grip	true/false	Show the size grips of dialogs and statusbars as far as possible?
Other Sections		
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.expansion	integer	A positive value will expand the frames until the corner frames meet each other either vertically or horizontally, *provided that at least the height or the width of the widget is not greater than it*. With appropriate SVG images, this key can be used for making corners completely rounded. Its value is zero by default. Read the file *Theme-Making.pdf* for an explanation.
interior	true/false	Draw an interior for the widget?
interior.x.patternsize interior.y.patternsize	integer	The interior pattern sizes. Used for tiling the interior of a widget with a pattern. A value of zero means no tiling in the corresponding direction. Their absence also means no pattern. *Some widget types may never accept patterns.*
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. It may override the text colors defined under the GeneralColors section. **Note:** State-specific text colors do not have meaning for *Window*, *Dock*, *LineEdit* and frame widgets (namely *GenericFrame*, *TabFrame* and *TabBarFrame*).
text.focus.color	String (#RRGGBB)	The color of the focused (hover) text.
text.press.color	String (#RRGGBB)	The color of the pressed text.
text.toggle.color	String (#RRGGBB)	The color of the toggled text.
text.bold	true/false	Bold font for text? The font is not bold by default.
text.italic	true/false	Italic font for text? The font is not italic by default.
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
text.toggle.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.toggle.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.toggle.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.