Content-type: text/html

# blackbox

Section: User Commands (1)
Updated: October 4th, 2000
Index Return to Main Contents

---

## NAME

blackbox - a window manager for X11

## SYNOPSIS

**blackbox** -help | -version
**blackbox** [ -rc *rcfile* ] [ -display *display* ]

## DESCRIPTION

Blackbox is yet another addition to the list of window managers for the Open Group's X Window System, Version 11 Release 6 and above. Blackbox is built with C++, sharing no common code with any other window manager (even though the graphics implementation is similar to that of Window Maker).

From the time the first line of code was written, Blackbox has evolved around one premise, minimalism. It's not meant to be Eye Candy, nor the most Featureful, nor the most Adorned for modelling the Widely acclaimed NeXT interface. It is just meant to be **fast**.

Blackbox provides configurable window decorations, a root menu to launch applications, and a toolbar that shows the current workspace name, the focused application name, and the current time. There is also a workspace menu to add or remove workspaces. The `slit' can be used to dock small applications, e.g. most of the bbtools can use the slit.

Blackbox features a special kind of icon handling: When you minimize a window, no icon appears; instead, you can view all minimized applications in the `Icons' submenu of the workspace menu. Your desktop will never get cluttered with icons. As an alternative to icons, shaded windows are provided: A double click on the titlebar of a window will shade it (i.e. the window will disappear; only the titlebar stays visible).

Blackbox uses its own graphics class to render its images on the fly. By using style files, you can determine at a great level how your desktop looks. Currently, KDE WM hints are not supported, but Blackbox is already prepared to support the new window manager specification that is now being developed for both Gnome and KDE 2.0.

## OPTIONS

Blackbox supports the following command line options:

**-help**
Display command line options and compiled-in features, then exit.

**-version**
Display version info and exit.
**-rc** *rcfile*
Use another rcfile than the default *~/.blackboxrc*.
**-display** *display*
Start Blackbox on the specified display. Programs started by Blackbox will have the **DISPLAY** environment variable set to this value, too.

# RUNNING BLACKBOX

This program is usually started by the user's startup script, most times called *~/.xinitrc*. To run blackbox, modify the script by adding

```
exec blackbox
```

as the last executed command of the script. When Blackbox terminates, the X session will terminate too.

When started, Blackbox will try to find a default menu file in */usr/local/share/Blackbox/menu*. You can provide a system-wide menu for your users here.

On exit or restart, Blackbox will save user defaults in the file *~/.blackboxrc* in the user's home directory. Some resources in this file can be edited by hand.

# USING BLACKBOX

From version 0.60.x, Blackbox does no keyboard handling by itself; instead, it relies on an external program *bbkeys*(1) for this. So, in this section, we will discuss all mouse commands.

## Root window (background):

A right click (button 3) will pop up the root menu. With this, you can launch your applications. You can customize this menu for your needs. A middle click (button 2) will pop up the workspace menu. You can add or remove a workspace, view applications running on all workspace, inspect your icons, and jump directly to any workspace or application.

Left clicking (button 1) on an application in the Workspaces menu will bring you to that workspace and raise/focus that application; middle clicking (button 2) will warp the application to the current workspace.

## Toolbar:

The toolbar consists of three fields: a workspace name, the name of the window that currently has focus, and a clock. A left click on the toolbar will bring it to the foreground, a middle click will hide it behind other windows (if AlwaysOnTop is not set), and the right button will bring up a little menu.

Using this menu, you can enter a name for the current workspace (when finished, press Enter). Also, you can choose the toolbar's position, whether or not it should be always on top (i.e. it cannot be obscured by other windows), and whether it should hide itself when the mouse moves away.

Note: In Blackbox versions below 0.60.0, a right click on the toolbar immediately enters workspace name edit mode.

### Window Titlebar and Borders:

A left click on any place of the window's border will raise it. Dragging then moves the window. Dragging the resize grips at the left-bottom and right-bottom corners resizes the window. Middle clicking on any place will immediately lower the window. Right clicking on the border or titlebar pops up the window menu, containing these commands:

**Send To...**
Send window to another workspace. When you select the workspace with the middle button, Blackbox will send you, along with the application, to the selected workspace.

**Shade**
Shade window (display titlebar only).

**Iconify**
Iconify window. The `icon' can be found in the `Icons' submenu of the workspace menu.

**Maximize**
(Un)Maximize window. When you click the middle button on this item, the window will maximize only vertically.

**Raise**
Raise window.

**Lower**
Lower window.

**Stick**
(Un)Stick window. A stuck window will always be displayed in the current workspace.

**Kill Client**
Kill (-SIGKILL) owner of window.

**Close**
Close the application cleanly.

When you double click on the titlebar of a window, it will `shade', so that only the titlebar stays visible. Another double click will redisplay the window contents.

### Window Buttons:

The button at the left upper corner of a window is the Minimize button. Clicking with any button causes the window to be iconified. The rightmost button (with the X) closes the application. The other button on the right (if present) maximizes the window in three ways: Button 1 causes full screen maximization, button 2 maximizes the window only vertically, and button 3 only horizontally.

### Any menu:

Clicking button 3 in a menu will popdown the menu. Clicking button 1 on the titlebar of any (sub)menu and then dragging it somewhere else will cause the menu to stay visible, and not disappear when you click on a menu item.

### Miscellaneous:

When you want to drag a window, but cannot see either the bottom handle or its titlebar, you can press Alt + button 1 anywhere in the window and then drag it around. You can also use Alt + button 1 to raise a partially visible window. Finally, Alt + button 2 lowers a window, and Alt + button 3 resizes the window.

## MENU FILE

A default menu file is installed in */usr/local/share/Blackbox/menu*. Of course, this system-wide menu can

be customized for all users at once. But it is also possible to create a personal menu. It is a convention to create a directory *~/.blackbox/* (or *~/blackbox/*) in your home directory, and to create a menu file, e.g. *menu* in this directory, or copy the system-wide menu file to this location. Next, we have to tell Blackbox to load our menu file instead of the default. This is accomplished by adding (or changing) a resource value in the *~/.blackboxrc* file, e.g.:

```
session.menuFile:       ~/.blackbox/menu
```

For this change to take effect, Blackbox has to be restarted. Be sure that your menu is usable, then choose `Restart' from the default Blackbox root menu.

## Menu syntax

The menu syntax is very simple and very effective. There are up to three fields in a menu line. They are of the form:

```
[tag] (label or filename) {command or filename}
```

The supported tags are as follows:

**[begin] (label for root menu)**
: This tells Blackbox to start parsing the menu file. This tag is required for Blackbox to parse your menu file. If it cannot find it, the system default menu is used instead.

**[end]**
: This tells Blackbox that it is at the end of a menu. This can either be a submenu or the main root menu. There must be at least one of these tags in your menu to correspond to the required [begin] tag.

**[exec] (label for command) {shell command}**
: Inserts a command item into the menu. When you select the menu item from the menu, Blackbox runs `shell command.'

**[exit] (label for exit)**
: Inserts an item that shuts down and exits Blackbox. Any open windows are reparented to the root window before Blackbox exits.

**[include] (filename)**
: Parses the file specified by *filename* inline with the current menu. The filename can be the full path to a file, or it can begin with ~/, which will be expanded into your home directory (e.g.

```
[include] (~/blackbox/stylesmenu)
```

will include */home/bhughes/blackbox/stylesmenu* in my menu).

**[nop] (label - optional)**
: Insert a non-operational item into the current menu. This can be used to help format the menu into blocks or sections if so desired. **[nop]** does accept a label, but it is not required, and a blank item will be used if none is supplied.

**[style] (label) {filename}**
: This tells Blackbox to insert an item that, when selected, reads the style file named *filename* and apply the new textures, colors, and fonts to the current running session.

**[stylesdir] (directory name)**
: Reads all filenames from the specified directory, assuming that they are all valid style files (directories are ignored), and creates menu items in the current menu for every filename that, when selected by the user, apply the selected style file to the current session. The labels that are created in the menu are the filenames of the style files.

**[stylesmenu] (label) {directory name}**

Creates a submenu entry with *label* (that is also the title of the new submenu), and inserts in that submenu all filenames in the specified directory, assuming that they are all valid style files (directories are ignored), in the same way as the **[stylesdir]** command does.
Both **[stylesdir]** and **[stylesmenu]** commands make it possible to install style files without editing your menu file.

**[submenu] (label) {title for menu - optional}**
This tells Blackbox to create and parse a new menu. This menu is inserted as a submenu into the parent menu. These menus are parsed recursively, so there is no limit to the number of levels or nested submenus you can have. The title for the new menu is optional; if none is supplied, the new menu's title is the same as the item label. An **[end]** tag is required to end the submenu.

**[reconfig] (label)**
When selected, this item rereads the current style and menu files and applies any changes. This is useful for creating a new style or theme, as you don't have to constantly restart Blackbox every time you save your style. However, Blackbox automagically rereads the menu whenever it changes.

**[restart] (label) {shell command - optional}**
This tells Blackbox to restart. If `shell command' is supplied, it shuts down and runs the command (which is commonly the name of another window manager). If the command is omitted, Blackbox restarts itself.

**[config] (label)**
Inserts a Blackbox native submenu item containing numerous configuration options concerning window placement, focus style, window moving style, etc.

**[workspaces] (label)**
This tells Blackbox to insert a link to the workspaces menu directly into your menu. This is handy for those users who can't access the workspace menu directly (e.g. if you don't have a 3 button mouse, it's rather hard to middle click to show the workspace menu).

Any line that starts with a `#' is considered a comment and ignored by Blackbox. Also, in the labels/commands/filenames fields, you can escape any character like so:

```
[exec] (\(my cool\) \{XTERM\}) {xterm -T \"cool XTERM\"}
```

Using `\\' inserts a literal back-slash into the label/command/filename field.

## Menu example

Now let's put together some things. Here is a short example of a menu file:

```
# Blackbox menu file
[begin] (Blackbox 0.60.3)
  [exec] (rxvt) {rxvt -ls}
  [exec] (netscape) {netscape -install}
  [exec] (The GIMP) {gimp}
  [exec] (XV) {xv}
  [submenu] (pine)
    [exec] (inbox) {rxvt -name pine -e pine -i}
    [exec] (new message) {rxvt -name pine -e pine ""}
  [end]
  [submenu] (Window Manager)
    [exec] (Edit Menus) {nedit .blackbox/Menu}
    [submenu] (Style) {Which Style?}
      [stylesdir] (~/.blackbox/styles)
      [stylesmenu] (Blackbox Styles) {/usr/local/share/Blackbox/styles}
    [end]
    [config] (Config Options)
    [reconfig] (Reconfigure)
    [restart] (Restart)
  [end]
  [exit] (Log Out)
[end]
# end of menu file
```

# STYLES

Blackbox enables you to use specialized files that contain *X*(1) resources to specify colors, textures, and fonts, and thus the overall look of your window borders, menus, and the toolbar.

The default installation of Blackbox provides some of these style files. Usually they are put in */usr/local /share/Blackbox/styles*. You can study or edit these files to grasp how the Blackbox style mechanism works. You can use the **[style]**, **[stylesdir]** and **[stylesmenu]** menu commands in your menu file to be able to select and change between styles on the fly.

But you can also create a directory named *~/.blackbox/styles* in your home directory and put your own style files here. Of course, you may choose any name for this directory, but many downloadable themes will rely on the name *styles* (following the bb.themes.org naming scheme).

To understand how the style mechanism works, you should have a little knowledge of how X resources work.

X resources consist of a key and a value. The key is constructed of several smaller keys (sometimes referred to as children), delimited by a period (`.'). Keys may also contain a star (`*') to serve as a wildcard, which means that one line of typed text will match several keys. This is useful for styles that are based on one or two colors.

Blackbox allows you to configure its three main components: the toolbar, the menus, and the window decorations.

The little window that shows the x-y position while dragging windows borrows its style from the window's titlebar.

Here are some quick examples:

```
toolbar.clock.color:    green
```

This sets the color resource of the toolbar clock to `green'. Another example:

```
menu*color:     rgb:3/4/5
```

This sets the color resource of the menu *and all of its `children'* to `rgb:3/4/5'. (For a description of color names, see *X*(1).) So this one also applies to *menu.title.color* and *menu.frame.color*. And with

```
*font:  -b&h-lucida-medium-r-normal-*-*-140-*
```

you set the font resource for all keys to this font name all at once. (For information about the fonts installed on your system, you can use a program like *xfontsel*(1), *gtkfontsel*, or *xlsfonts*(1).)

Now what makes Blackbox just so spectacular is its ability to render textures on the fly. Texture descriptions are specified directly to the key that they should apply to, e.g.:

```
toolbar.clock:         Raised Gradient Diagonal Bevel1
toolbar.clock.color:   rgb:8/6/4
toolbar.clock.colorTo: rgb:4/3/2
```

Don't worry; we will explain right now! A texture description consists of up to five fields, which are as follows:

**Flat / Raised / Sunken**
gives the component either a flat, raised, or sunken appearance.

**Gradient / Solid**
tells Blackbox to draw either a solid color or a gradiented texture.

**Horizontal / Vertical / Diagonal / Crossdiagonal / Pipecross / Elliptic / Rectangle / Pyramid**
Select one of these texture types. They only work when **Gradient** is also specified!

**Interlaced**
tells Blackbox to interlace the texture (darken every other line). This option is most commonly used with gradiented textures, but, from Blackbox version 0.60.3 on, it also works in solid textures.

**Bevel1 / Bevel2**
tells Blackbox which type of bevel to use. Bevel1 is the default bevel. The shading is placed on the edge of the image. Bevel2 is an alternative. The shading is placed one pixel in from the edge of the image.

Instead of a texture description, the option **ParentRelative** is also available, which makes the component appear as a part of its parent, i.e. totally transparent.

All gradiented textures are composed of two color values: the *color* and *colorTo* resources. When **Interlaced** is used in **Solid** mode, the *colorTo* resource is used to find the interlacing color.

Well, here is the complete component list; also, all components together with which kind of value they can contain. Comments are preceded with an exclamation sign (!), which is also used for comments in Blackbox style c.q. X resource files.

```
! The toolbar itself.
toolbar:                        Texture
toolbar.color:                  Color
toolbar.colorTo:                Color

! The buttons on the toolbar.
toolbar.button:                 Texture or ParentRelative
toolbar.button.color:           Color
toolbar.button.colorTo:         Color

! Color of the button arrows.
toolbar.button.picColor:        Color

! Buttons in pressed state.
toolbar.button.pressed:         Texture (e.g. Sunken) or ParentRelative
toolbar.button.pressed.color:   Color
toolbar.button.pressed.colorTo: Color

! Color of pressed button arrows.
toolbar.button.pressed.picColor:Color

! The toolbar workspace label.
toolbar.label:                  Texture or ParentRelative
toolbar.label.color:            Color
toolbar.label.colorTo:          Color
toolbar.label.textColor:        Color

! The toolbar window label.
toolbar.windowLabel:            Texture or ParentRelative
toolbar.windowLabel.color:      Color
toolbar.windowLabel.colorTo:    Color
toolbar.windowLabel.textColor:  Color

! The toolbar clock.
toolbar.clock:                  Texture or ParentRelative
toolbar.clock.color:            Color
toolbar.clock.colorTo:          Color
toolbar.clock.textColor:        Color

! How the toolbar's text should be justified.
toolbar.justify:                center, left, or right
```

```
! Font to be used for all toolbar components.
toolbar.font:                   Font (e.g. -*-helvetica-medium-r-normal-*-*-100-*)

! The menu titlebar.
menu.title:                     Texture
menu.title.color:               Color
menu.title.colorTo:             Color
menu.title.textColor:           Color
menu.title.font:                Font
menu.title.justify:             center, left, or right

! The menu frame.
menu.frame:                     Texture
menu.frame.color:               Color
menu.frame.colorTo:             Color
menu.frame.textColor:           Color
menu.frame.disableColor:        Color
menu.frame.font:                Font
menu.frame.justify:             center, left, or right

! Bullets for submenu items.
menu.bullet:                    empty, triangle, square, or diamond
menu.bullet.position:           right or left

! The highlighted menu item.
menu.hilite:                    Texture (e.g. Raised)
menu.hilite.color:              Color
menu.hilite.colorTo:            Color
menu.hilite.textColor:          Color

! A focused window.
window.title.focus:             Texture
window.title.focus.color:       Color
window.title.focus.colorTo:     Color

! An unfocused window.
window.title.unfocus:           Texture
window.title.unfocus.color:     Color
window.title.unfocus.colorTo:   Color

! Window label.
window.label.focus:             Texture or ParentRelative
window.label.focus.color:       Color
window.label.focus.colorTo:     Color
window.label.focus.textColor:   Color

window.label.unfocus:           Texture or ParentRelative
window.label.unfocus.color:     Color
window.label.unfocus.colorTo:   Color
window.label.unfocus.textColor: Color

! Handlebar.
window.handle.focus:            Texture
window.handle.focus.color:      Color
window.handle.focus.colorTo:    Color

window.handle.unfocus:          Texture
window.handle.unfocus.color:    Color
window.handle.unfocus.colorTo:  Color

! Resize grips.
window.grip.focus:              Texture
window.grip.focus.color:        Color
window.grip.focus.colorTo:      Color

window.grip.unfocus:            Texture
window.grip.unfocus.color:      Color
window.grip.unfocus.colorTo:    Color

! Window buttons.
window.button.focus:            Texture or ParentRelative
window.button.focus.color:      Color
window.button.focus.colorTo:    Color
```

```
window.button.focus.picColor:   Color

window.button.unfocus:          Texture or ParentRelative
window.button.unfocus.color:    Color
window.button.unfocus.colorTo:  Color
window.button.unfocus.picColor: Color

window.button.pressed:          Texture (e.g. Sunken)
window.button.pressed.color:    Color
window.button.pressed.colorTo:  Color

! Frame around window.
window.frame.focusColor:        Color
window.frame.unfocusColor:      Color

! Font and justification for window labels.
window.font:                    Font
window.justify:                 center, left, or right

! Miscellaneous resources.

! A border can be drawn around all components.
borderWidth:                    a number of pixels, e.g. 1
borderColor:                    Color

bevelWidth:                     a number of pixels > 0
handleWidth:                    a number of pixels > 0

! Width of the window frame (from version 0.61 on).
! When not specified, frameWidth defaults to the value of bevelWidth.
frameWidth:                     a number of pixels >= 0


! This command is executed whenever this style is selected.
! Typically it sets the root window to a nice picture.
rootCommand:                    Shell command, e.g. bsetroot -mod 4 4 -fg rgb:5/6/6 -bg grey20

! Some of the bbtools read these old 0.51 resources.
menuFont:                       Font
titleFont:                      Font
```

Now, this seems a long list, but, remember, when you create your own style, you can easily set lots of keys with a single command, e.g.

```
*color:             slategrey
*colorTo:           darkslategrey
*unfocus.color:     darkslategrey
*unfocus.colorTo:   black
*textColor:         white
*unfocus.textColor: lightgrey
*font:              lucidasans-10
```

This sets already nice defaults for many components.

# THE SLIT

The slit is a special Blackbox window frame that can contain dockable applications, e.g. the `bbtools.' When applications are run in the slit, they have no window borders of their own; instead, they are framed in the slit, and they are always visible in the current workspace. You can click button 3 on the edge of the slit window to get a menu to determine its position, whether its contained applications should be grouped horizontally or vertically and whether it should hide itself when the mouse moves away.

Most dockable applications use the **-w** option to run in the slit. For example, you could put in your *~/.xinitrc*:

```
bbmail -w &
bbpager -w &
exec blackbox
```

Of course, to use the slit, you must have slit support compiled in.

# RESOURCE FILE

Usually the *~/.blackboxrc* resource file is created and maintained by Blackbox itself. All options from the **[config]** menu (from 0.60.x on), the last selected style file, your workspace names and so on are saved into this file. However, there are some resources in it you might want to edit yourself:

**session.menuFile:**
: This tells Blackbox where to look for its menu file.

**session.screen0.toolbar.widthPercent:**
: This determines the amount (in %) of space the toolbar will take. Default value is: *66*.

**session.screen0.strftimeFormat:**
: This adjusts the way the current time is displayed in the toolbar. The *strftime*(3) format is used. Default value is: *%I:%M %p*.

**session.autoRaiseDelay:**
: This adjusts the delay (in ms) before focused windows will raise when using the Auto Raise option. Default value is: *250*.

**session.doubleClickInterval:**
: This adjusts the delay (in ms) between mouse clicks for Blackbox to consider a double click. Default value is: *250*.

**session.screen0.edgeSnapThreshold:**
: When moving a window across your screen, Blackbox is able to have it `snap' to the edges of the screen for easy placement. This variable tells Blackbox the distance (in pixels) at which the window will jump to the edge. Default value is: *0*.

**session.cacheMax:**
: This tells Blackbox how much memory (in Kb) it may use to store cached pixmaps on the X server. If your machine runs short of memory, you may lower this value. Default value is: *200*.

**session.cacheLife:**
: This tells Blackbox how long (in minutes) unused pixmaps may stay in the X server's memory. Default value is: *5*.

**session.colorsPerChannel:**
: This tells Blackbox how many colors to take from the X server on pseudocolor displays. A channel would be red, green, or blue. Blackbox will allocate this variable ^ 3 colors and make them always available. This value must be between 2 and 6. When you run Blackbox on an 8-bit display, you must set this resource to 4. Default value is: *4*.

When running Blackbox in a multiple desktop environment, the **screen0** key can also be **screen1, 2** etc. for any appropriate desktop.

# ENVIRONMENT

**HOME**
: Blackbox uses $**HOME** to find its *.blackboxrc* file, and to resolve style file and directory names.

**DISPLAY**
: When no other display was given on the command line, Blackbox will start on the display specified by this variable.

# AUTHOR and CREDITS

All of the code is written and maintained by Brad Hughes (blackbox@alug.org) and Jeff Raven (jraven@psu.edu), with contributions and patches merged from many individuals around the world.

The official Blackbox website: **http://blackbox.alug.org/**
Many themes and other contributions: **http://bb.themes.org/**

This manpage was put together by Wilbert Berendsen (wbsoft@xs4all.nl). Numerous other languages will be available.

# SEE ALSO

*bsetroot*(1), *bbkeys*(1)

---

# Index



---

This document was created by man2html, using the manual pages.
Time: 21:16:29 GMT, March 31, 2001