# RUBY/TK - FONTS, COLORS AND IMAGES

 

## Ruby/Tk Fonts

Several Tk widgets, such as the label, text, and canvas, allow you to specify the fonts used to display text, typically via a *font* configuration option.

There is already a default list of fonts, which can be used for different requirements:

| Font Name | Description |
|---|---|
| TkDefaultFont | The default for all GUI items not otherwise specified. |
| TkTextFont | Used for entry widgets, listboxes, etc. |
| TkFixedFont | A standard fixed-width font. |
| TkMenuFont | The font used for menu items. |
| TkHeadingFont | The font typically used for column headings in lists and tables. |
| TkCaptionFont | A font for window and dialog caption bars. |
| TkSmallCaptionFont | A smaller caption font for subwindows or tool dialogs |
| TkIconFont | A font for icon captions. |
| TkTooltipFont | A font for tooltips. |

You can use any of these fonts in the following way:

```
TkLabel.new(root) {text 'Attention!'; font TkCaptionFont}
```

If you are willing to create your new font using different family and font type, then here is a simple syntax to create a font:

```
TkFont.new(
  .....Standard Options....
)
```

## Standard Options:

You can specify one or more standard option separated by comma.

- **Foundry**
- **Family**
- **Weight**
- **Slant**
- **Swidth**
- **Pixel**
- **Point**
- **Xres**

- **Yres**
- **Space**
- **Avgwidth**
- **Registry**
- **Encoding**

## Ruby/Tk Colors

There are various ways to specify colors. Full details can be found in the colors command reference.

The system will provide the right colors for most things. Like with fonts, both Mac and Windows specify a large number of system-specific color names *seethereference*.

You can also specify fonts via RGB, like in HTML, e.g. "#3FF" or "#FF016A".

Finally, Tk recognizes the set of color names defined by X11; normally these are not used, except for very common ones such as "red", "black", etc.

For themed Tk widgets, colors are often used in defining styles that are applied to widgets, rather than applying the color to a widget directly.

## Examples:

```
require 'tk'

$resultsVar = TkVariable.new
root = TkRoot.new
root.title = "Window"
myFont = TkFont.new("family" => 'Helvetica',
                    "size" => 20,
                    "weight" => 'bold')
Lbl = TkLabel.new(root) do
  textvariable
  borderwidth 5
  font myFont
  foreground  "red"
  relief      "groove"
  pack("side" => "right",  "padx"=> "50", "pady"=> "50")
end

Lbl['textvariable'] = $resultsVar
$resultsVar.value = 'New value to display'

Tk.mainloop
```

This will produce the following result:


## Ruby/Tk Images

Ruby/Tk includes support for GIF and PPM/PNM images. However, there is a Tk extension library

called "Img" which adds support for many others: BMP, XBM, XPM, PNG, JPEG, TIFF, etc. Though not included directly in the Tk core, Img is usually included with other packaged distributions.

Here, we will see the basics of how to use images, displaying them in labels or buttons for example. We create an image object, usually from a file on disk.

## Examples:

```
require 'tk'

$resultsVar = TkVariable.new
root = TkRoot.new
root.title = "Window"

image = TkPhotoImage.new
image.file = "zara.gif"

label = TkLabel.new(root)
label.image = image
label.place('height' => image.height,
            'width' => image.width,
           'x' => 10, 'y' => 10)
Tk.mainloop
```

This will produce the following result:


Tk's images are actually quite powerful and sophisticated and provide a wide variety of ways to inspect and modify images. You can find out more from the image command reference and the photo command reference