Custom GRUB Theme Tutorial

This tutorial combines information compiled over time. While I have contributed to its content, many anonymous sources have also provided valuable insights and instructions. Although I cannot specifically credit all contributors, I extend my appreciation to everyone who has shared their knowledge.

Create a GRUB2 Theme

Create a Directory

First, create an empty directory for the new theme files. It is recommended to create a subdirectory in `/boot/grub2/themes/`.

Create the directory:

```
sudo mkdir /boot/grub2/themes/test/
```

Create the Theme File

1. **Create `theme.txt`:**

   It is recommended to name the file `theme.txt`.

   ```
   sudo nano /boot/grub2/themes/test/theme.txt
   ```

2. **Change GRUB Parameters:**

   Edit the GRUB configuration file:

   ```
   sudo nano /etc/default/grub
   ```

   Find the `GRUB_THEME` parameter in `/etc/default/grub`.

   To switch to your theme, set the parameter `GRUB_THEME` in the file `/etc/default/grub`.

   Add the following line, or modify it if it already exists:

   ```
   GRUB_THEME=/boot/grub2/themes/test/theme.txt
   ```

   Notes:

   - After the sign `"="`, write the absolute path to the main theme file.
   - If the line `GRUB_THEME=...` already exists in `/etc/default/grub`, rewrite it to match `GRUB_THEME=/boot/grub2/themes/test/theme.txt`.
   - There should not be two lines for the same parameter.

3. **Update Bootloader Configuration File:**

Changes in this file won't be applied until you regenerate the bootloader configuration file.

- Usually, there is a script `update-grub2` in the system. It should be run with root rights.

```
sudo update-grub2
```

- If `update-grub2` is not available, use the following command:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

Notes:

- The command `update-grub2` does the same as `grub2-mkconfig`.
- `update-grub2` should be called each time you make changes to `/etc/default/grub`.

GRUB2 Theme Options

Most GRUB2 theme options are described in `theme.txt`. GRUB2 parses this file each time during the boot process. Reboot to see changes made to `theme.txt` or the theme directory.

Common Concepts

Errors in global options and component names will show during boot. Errors in component options will not.

Types of Elements

- **Root Element:** Top level of `theme.txt`, sets global options.
- **Boot Menu (boot_menu):** Sets boot menu options.
- **Progress Bar (progress_bar):** Horizontal timeout indicator.
- **Circular Progress (circular_progress):** Circular timeout indicator.
- **Label (label):** Outputs a text string.
- **Image (image):** Outputs an image.
- **Vertical Container (vbox):** Outputs elements vertically.
- **Horizontal Container (hbox):** Outputs elements horizontally.
- **Canvas (canvas):** Coordinates set for each child element.

Root element is a canvas stretched to screen dimensions.

Images

Images are stretched to defined sizes, or their real sizes if smaller. Be mindful of different display aspect ratios (4:3 vs 16:9).

- **Logotype Images:** Use absolute width and height to avoid distortions.
- **Gradients:** Use a 1px width gradient image for various effects.

Graphical Decoration

Graphical decoration of an element consists of 9 zones: central, south, south-west, west, etc.

- **Non-Stretched Zones:** 4 diagonal zones.
- **Stretched Zones:** East/west zones by vertical, north/south zones by horizontal, central zone by both.

Ensure consistency in zone sizes to avoid distortions.

Fonts

Create fonts using the `grub2-mkfont` utility. Refer to the "Font creating guide" for details.

Test Scheme Description

- **Increased 5x for clarity, alpha-channel at 75%.**
- **Dark red background for clarity.**

Graphical elements include: - **Inbox:** `inbox_*.png` - **Button:** `button_*.png` - **Menu:** `menu_*.png` - **Scrollbar:** `scrollbar_frame_*.png`, `scrollbar_thumb_*.png` - **Progress Bar:** `progress_*.png` - **Circular Progress Center Image:** `center.png` - **Circular Progress Tick Image:** `tick.png`

Common Properties

- **Coordinates:** `left, top, width, height` - rectangle containing an element.
- **ID:** Special value affects element display.

Root Element

Contains all theme elements and global options. - **Global Options:** Use "-" for names (e.g., "terminal-box"), ":" as a divider (e.g., "title_text:"GNU Grub""). - **Element Options:** Use"_" for names (e.g., "item_height"), "=" as a divider (e.g., "item_height = 20").

Desktop Color and Image

- **desktop-color:** Background color.
- **desktop-image:** Background image, placed in the theme's directory and scaled to screen resolution.

Title Settings

- **Title Text, Color, Font:**
  - **Hide Title:** Set `title-text=""`.
  - **Center Aligned:** Baseline at 40 + ascent from the top.

Example:

```
title-text: "Title Text"
title-color: "#FFFFFF"
desktop-image: "background.png"
```

Console Decoration

- **Decoration:** Outside of console rectangle with a 3px padding. Default background is black, changeable via `/etc/default/grub`.

Console Font

- **Font:** Name of the console font. After adding a new font, run `update-grub2`.

Example:

```
terminal-font: "Droid Sans Mono Regular 11"
terminal-box: "inbox_*.png"
```

Boot Menu

- **Main Element:** Lists installed operating systems.
- **Coordinates:** `left, top, width, height` - rectangle coordinates.
- **Font and Color:** Use for selected and non-active menu items.

Example:

```
+ boot_menu {
    left = 50%-300
    top = 50%-100
    width = 600
    height = 300
    selected_item_color = "#0000FF"
}
```

Item Decoration

- **Selected Item Decoration:** `selected_item_pixmap_style`
- **Boot Menu Decoration:** `menu_pixmap_style`

Example:

```
+ boot_menu {
    selected_item_pixmap_style = "button_*.png"
    menu_pixmap_style = "inbox_*.png"
}
```

Padding and Spacing

- **Item Padding:** Printing area padding from each side.
- **Height Calculation:** Use the given formula for height.

Example:

```
item_padding = 20
item_spacing = 20
item_height = 30
```

Icons and Icon Space

- **Default Values:**
  - `icon_width = 32`
  - `icon_height = 32`
  - `item_icon_space = 4`
- **Free Space:** Indented for icons regardless of their presence.

Example:

```
icon_width = 0
item_icon_space = 4
```

For a complete understanding, refer to the GRUB2 Theme Reference.

Using Icons

To use icons, create a directory named `image` within your theme directory. Place your icons in PNG format into this directory. The icon filenames should correspond to specific predefined classes. The default classes are:

- `windows` > `os`
- `gnu-linux` > `gnu` > `os`
- `osx` > `darwin` > `os`
- `hurd` > `gnu` > `os`

The most specific class takes precedence. For example, if `windows.png` is present, it will be used, and `os.png` will not. The main system class (which is the name of the system where the `update-grub2` script was called) is also available and takes precedence over `gnu-linux`. For instance, on a ROSA system, this class will be “rosa”. You can find classes in the `/boot/grub2/grub.cfg` file.

Scrollbar Settings

For detailed scrollbar settings, consider the following points:

- Do not use slices other than north, south, and center for `scrollbar_thumb`.
- Use all 9 slices for `scrollbar_frame`.
- Increase padding by adding transparent pixels to the corresponding slice.
- The scrollbar is drawn to the right of the center slice, and it is drawn only if needed.
- The width of `scrollbar_frame` and `scrollbar_thumb` can differ.
- The `scrollbar_width` parameter specifies the width of the scrollbar’s central part. If the combined width of `scrollbar_width` and the east and west slices of `scrollbar_frame` is less than the width of the east slice of the menu’s graphical decoration, the scrollbar won’t be fully drawn.
- Set both `scrollbar_frame` and `scrollbar_thumb`. If one is missing, the scrollbar won’t be visible.

Here’s an example of a correctly displayed scrollbar:

```
+ boot_menu {
       left = 50%-300
       top = 50%-100
       width = 600
       height = 289

       selected_item_color = "#0000FF


"
       scrollbar_thumb = "scrollbar_thumb_*.png"
       scrollbar_frame = "scrollbar_frame_*.png"
}
```

Remember that the content width of the vertical part of the menu should be at least the sum of the scrollbar width and the east slice of the `menu_pixmap_style`.

In the next section, we will explore the Progress Bar and Circular Progress indicators, which are crucial for enhancing your GRUB theme.

Progress Bar

A progress bar is a horizontal indicator showing the boot progress or timeout. Here are the essential properties and an example:

1. **Basic Settings:**

   - **Coordinates:** Set the position and size of the progress bar.
   - **Pixmaps:** Specify the images for the progress bar elements.

2. **Element Properties:**

   - **left, top, width, height:** Define the rectangle containing the progress bar.
   - **pixmap:** Image file used for the progress bar background.
   - **foreground:** Image file used for the progress bar foreground.

3. **Example:**

   ```
   + progress_bar {
       left = 50%-150
       top = 50%+150
       width = 300
       height = 30
       pixmap = "progress_bg.png"
       foreground = "progress_fg.png"
   }
   ```

Circular Progress

A circular progress indicator shows the boot progress or timeout in a circular format. Here are the essential properties and an example:

1. **Basic Settings:**

    - **Coordinates:** Set the position and size of the circular progress indicator.
    - **Center Image:** Image displayed at the center of the circle.
    - **Tick Image:** Image used for the progress ticks.

2. **Element Properties:**

    - **left, top, width, height:** Define the rectangle containing the circular progress indicator.
    - **center_image:** Image file used for the center of the circular progress.
    - **tick_image:** Image file used for the ticks.

3. **Example:**

```
+ circular_progress {
    left = 50%-50
    top = 50%+50
    width = 100
    height = 100
    center_image = "center.png"
    tick_image = "tick.png"
}
```

Labels

Labels are used to display text strings in your GRUB theme. Here are the essential properties and an example:

1. **Basic Settings:**

    - **Coordinates:** Set the position and size of the label.
    - **Text, Font, Color:** Specify the text, font, and color of the label.

2. **Element Properties:**

    - **left, top, width, height:** Define the rectangle containing the label.
    - **text:** Text string to be displayed.
    - **font:** Font used for the text.
    - **color:** Color of the text.

3. **Example:**

```
+ label {
    left = 50%-200
    top = 50%-50
    width = 400
    height = 100
    text = "Welcome to GRUB"
    font = "Droid Sans Mono Regular 16"
    color = "#FFFFFF"
}
```

Images

Images are used to display graphical elements in your GRUB theme. Here are the essential properties and an example:

1. **Basic Settings:**
   - **Coordinates:** Set the position and size of the image.
   - **File:** Specify the image file.
2. **Element Properties:**
   - **left, top, width, height:** Define the rectangle containing the image.
   - **file:** Image file to be displayed.
3. **Example:**

```
+ image {
    left = 50%-100
    top = 50%-100
    width = 200
    height = 200
    file = "logo.png"
}
```

Containers

Containers are used to group elements vertically (vbox) or horizontally (hbox). Here are the essential properties and an example for each type:

VBox (Vertical Container)

1. **Basic Settings:**
   - **Coordinates:** Set the position and size of the container.
   - **Elements:** List the elements contained within the vbox.
2. **Element Properties:**
   - **left, top, width, height:** Define the rectangle containing the vbox.
   - **elements:** List of elements within the vbox.
3. **Example:**

```
+ vbox {
    left = 50%-150
    top = 50%-150
    width = 300
    height = 400
    elements = {
        + label {
            text = "Option 1"
            font = "Droid Sans Mono Regular 16"
            color = "#FFFFFF"
        }
        + label {
            text = "Option 2"
            font = "Droid Sans Mono Regular 16"
            color = "#FFFFFF"
        }
    }
}
```

HBox (Horizontal Container)

1. **Basic Settings:**

   - **Coordinates:** Set the position and size of the container.
   - **Elements:** List the elements contained within the hbox.

2. **Element Properties:**

   - **left, top, width, height:** Define the rectangle containing the hbox.
   - **elements:** List of elements within the hbox.

3. **Example:**

```
+ hbox {
    left = 50%-150
    top = 50%-50
    width = 300
    height = 100
    elements = {
        + label {
            text = "Option 1"
            font = "Droid Sans Mono Regular 16"
            color = "#FFFFFF"
        }
        + label {
            text = "Option 2"
            font = "Droid Sans Mono Regular 16"
            color = "#FFFFFF"
        }
    }
}
```

Final Steps

Once you have created and configured your theme elements in the `theme.txt` file, you need to:

1. **Save the `theme.txt` file:** Ensure it is located in `/boot/grub2/themes/test/`.

2. **Update GRUB Configuration:** Run the following command to apply your new theme:

```
sudo update-grub2
```

3. **Reboot:** Restart your system to see the changes in effect.

Conclusion

Creating a custom GRUB theme involves organizing graphical elements, setting their properties, and configuring the overall look and feel of the bootloader. By following the examples and guidelines provided in this tutorial, you can create a unique and visually appealing GRUB theme that enhances your system’s boot experience.

For further customization, refer to the GRUB2 Theme Reference for detailed information on additional properties and options.