From Design to Developer

an informal outline of “how to’s” and best practices when exporting visual assets for theme

Defining common colors, textures, and shapes

By documenting the common colors that are used in the design we can ensure that all views are consistent and that the developers can move forward with the right information. Common colors can be used for instances of text, general backgrounds, and highlights.

Defining your color palette gives the engineers a guideline when they come across something that may have been overlooked by the designer. It also helps when deciding whether something can be created with code rather than relying on a graphic.

Only if the element has a texture or styling should an image be used. If there is in a fact a texture, the swatch should be as small as possible to create the pattern of repetition. In terms of cutting up your PSD (or Gimp file), this should be done before export. There will be more on exporting later.

Side note: if we ever manage to get svg to work well in the theme, they would be an ideal method for asset creation. Though time consuming, once created they are easy to change and update, effectively automating a big part of the process.

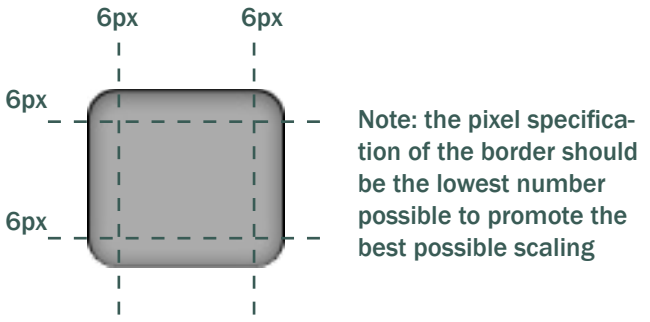
Please refer to James Ketrenos' work on scaling between device sizes and resolutions at: http://muxbox.jf.intel.com/wiki/index.php/~jketreno/scaling

Separating your graphics for the future

When an engineer is putting together a layout (in QML for our purposes), they prefer to write as little code as possible. Keep in mind that the UI you are creating assets for is likely going to be ported to a different size and resolution to run on a variety of devices. *Flexibility is important!* By following these guidelines it makes everyone's life a little easier:

1. Button icons and button backgrounds, including the highlights, need to be exported separately. By doing this, if the background and highlight shapes and/or colors are used in multiple places then the developer need only call one graphic and scale it accordingly. This also ensures that if that icon is used in other places it too need only appear once in the theme package, reducing size and improving the speed of the system.

2. If you have common shapes with rounded corners or uniquely styled sides, you may reduce the shape to the smallest possible size. QML allows the developer to specify "borders" on a png that will automatically keep the edges from distorting while the shape scales to fit the needs of the interface. The developers in fact prefer this method.



Once you have your graphic you can specify the borders in an accompanying file with the extension |.sci|. It will look like this in the text:

```
border.left:    6px
border.top      6px
border.bottom   6px
border.right:   6px
source:         slider-background.png
```

3. If you can, use the Photoshop/Gimp →QML export script described here: http://labs.qt.nokia.com/2010/10/19/exporting-qml-from-photoshop-and-gimp/ This script, if set up properly, will export the assets for you and generate a QML source file for the developer. The source file describes coordinates of the png's and opacity levels of the graphics. This will quickly get a layout set up. However this should be used with the intention of modifying any graphics that need it. I will describe more about this script later.

4. *Think before you export.* What is the best way to do this? Often you can get away with relying on the QML script exporter to do the heavy lifting. But sometimes you may need to modify a graphic before export. Sometimes the graphic already exists and needn't be worried about.

5. Do you need to edit the files after export? Sometime this is necessary if you find that a certain graphic is bigger than it needs to be. An example is if you have a toolbar with a gradient or other effects that run the width (or length if the shape is vertical) and is rectangular, then you only need a small part which the developer can then scale to whatever size they need.

Original Export

Necessary shape

6. Naming conventions. A label may work for the designer, but may not be easily located by someone else and if the label is not descriptive or common enough it can make the icon difficult to find. It's best to put your icons in an intelligible directory and follow this formula or something like it:

(intelligent directory location)-[asset name]-[local position(toolbar, top, bottom, left, right, etc.)]-[state(pressed, unpressed, selected, up, down, etc)].[png, svg, jpeg, jpg, etc.]

For example we could name the following graphic as such:

theme/icons/media/shuffle-dn.png
or
theme/icons/common/shuffle-pressed.png

This ties in to our next topic - - - - - -

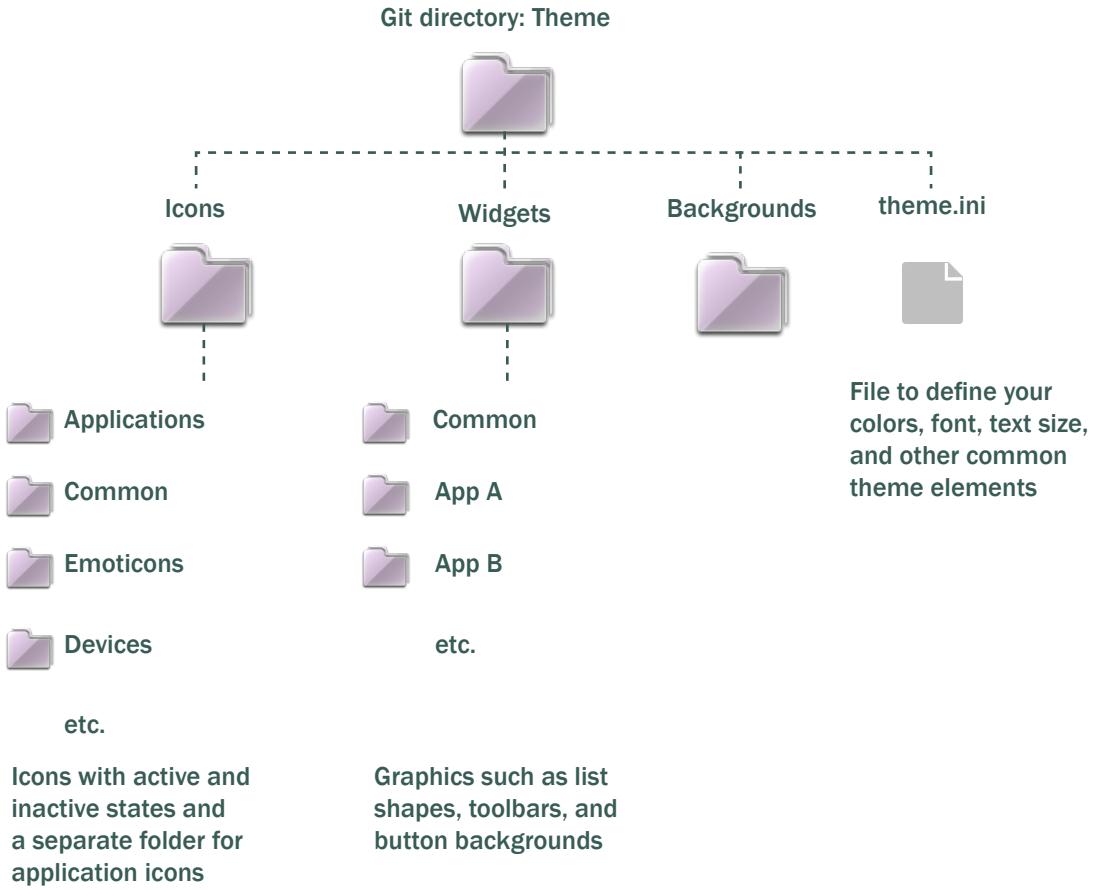
Asset management

Designate a gatekeeper. Someone should be monitoring the theme to make sure everything is consistent, up-to-date, on par, and as complete as possible.

Keep it clean. Most graphics should end up in a common folder since they will be used all over the system. If there are some application specific graphics then they should go into their own folder.

The best chance you have of keeping track of your icons is to give them their own directory as well. Whether the icons are used only once in one place or multiple times in multiple places, it's easier to keep track of what already exists and how you are using your icons if you keep them in one location. Following the naming convention above will also help you to keep track of where they belong in most cases.

You can look at the standards outlined at http://standards.freedesktop.org/icon-naming-spec/icon-naming-spec-latest.html for a clearer idea of how icons should be separated and named. However, you should be judicious. You may not need all those categories or even those labels if they don't work well.



From Photoshop and Gimp → QML wizardry

The QML export script works very much like the 'Export layers to files' script that Photoshop CS4 currently has. It will trim and export the layers to files and create a source file describing the location of each graphic and its opacity if it applies. To get a clean export though there is a bit of prep (at least for Photoshop) that needs to occur before you press the 'Run' button. They are as follows:

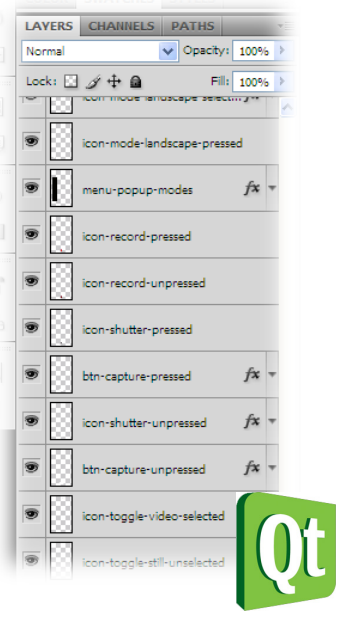
1. **Save as new:** you don't want to do this to your original file!

2. **Rasterize and condense.** Make sure all shapes and layers are in the state you would want them to appear in code. This is where you would resize shapes that are rounded or have styles to the most compact state. Anything you're not sure of that can wait until later, can wait until later. You can always re-export if you have to.

3. **Name your layers** according to the convention mentioned above and be sure to use dashes **NOT** underscores or spaces. For some reason underscores interfere with the initial run of the source file in QT Creator and spaces are just bad form. A developer can batch change later if it's really necessary.

4. **Remove all grouping.** It will cause an error in the exporting process and stop. You should be left with only layers.

5. **Save.** You don't want to do this all over again if something goes wrong and your computer chokes. You may also need to export again, so best to be safe.



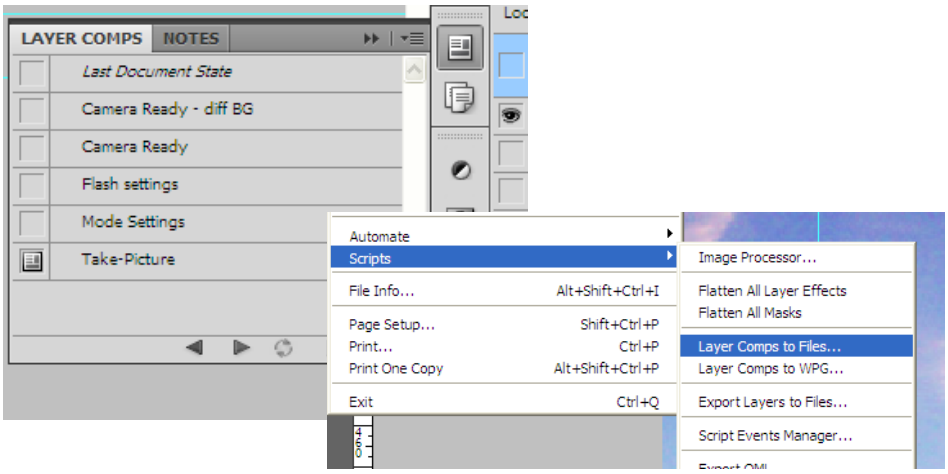
For Cyrene's Peace of Mind

Cyrene would love to see these things happen:

1. Keeping all the icons in one vector file. Finding and pulling icons out of a Photoshop file is all well and good until you need to change something or duplicate a shape. Having a master file helps me keep better tabs on what is and is not done, keeps things organized, and makes the visual designer happy.

2. Include down states in your Photoshop files for buttons or any other touchable object. I don't want to guess on what was intended. This leads back to properly defining common colors or styles for actions.

3. Use Layer Comps in your Photoshop file(s). It's really easy and may reduce the amount of PSD's you generate. When it comes time to export full-screen mockups all you have to do is run a script, "Layer Comps to files," and the deed is done.



If you require more information on any part of this please email me.
Cyrene Domogalla | cyrene.domogalla@intel.com