Organization of a Theme

Organization of a Theme

Let's diving into the way a theme is organized: folders, files, where they belong and how to handle them correctly

- Organization of a Theme
  - Folders
    - Thumbnail file
    - CSS and Sass
      - CSS
      - Sass / Compass
    - Font
    - Image
    - JavaScript
    - Language
    - The mobile theme
  - Files
    - Templates files
    - Style sheets
    - Image files
    - Tools

Folders

The main folders of any PrestaShop theme are those:

- The `/cache` folder contains all the temporary files that are generated and reused in order to lighten the server load. The folder is empty by default.
- The `/css` folder contains all CSS files.
  - The `/sass` folder contains all the Sass `.scss` source files, before they are compiled into the CSS files.
- The `/font` folder contains the needed font files.
- The `/img` folder contains all images.
- The `/js` folder contains all the JavaScript files.
- The `/lang` folder contains the theme's translations. Its access rights should be set at CHMOD 666 (for instance), so that the back office translation tool can read and write into it.

The following folders are not directly theme-related, but help you make sure the whole of PrestaShop's feature have a design that is consistent with your theme:

- The `/mails` folder contains the templates for the emails that PrestaShop sends (order confirmation, password request, shipping notifications, etc.).
- The `/mobile` folder contains the mobile version of the theme.
- The `/modules` folder contains the template files for many modules.
- The `/pdf` folder contains the template files for the PDF files that PrestaShop generates (invoices, delivery slip, supply orders, etc.).

The root of the folder contains TPL files only, as well as the `preview.jpg` thumbnail file.

Thumbnail file

The `preview.jpg` file at the root of the theme's folder is the thumbnail that is used by PrestaShop in its back office theme selector.

It serves as a visual reminder of what the theme is, and you should therefore make it a screenshot rather than your company's logo.

It can have any size – the default theme's is 180*445 pixels.

It must be a JPEG file.

CSS and Sass

CSS

The theme's CSS files are located in the `/css` folder.

It is recommended to have a common style sheet for global CSS rules: `global.css`.
Then, each of the controller should have its own CSS file: for instance, `product.css` for the Product page.

Sass / Compass

Sass and Compass files are optional: you do not need to use these tools to build the CSS files for your back office theme.

If you do use Sass and Compass, we strongly advise to put the source `.scss` files in the `/sass` theme, so that other developers can have access to theme and rework them more easily.

From there, you can generate the CSS files in the `/css` folders from the Sass files in the `/sass` folder!

Font

The `/font` folder is optional: it contains the fonts that you chose to use for your theme.

For instance, the default PrestaShop theme uses the Font Awesome font set (http://fortawesome.github.io/Font-Awesome/) for its responsive icons, and therefore has the following files in its `/font` folder:

- `fontawesome-webfont.eot`
- `fontawesome-webfont.svg`
- `fontawesome-webfont.ttf`
- `fontawesome-webfont.woff`

If you do not build your theme with a specific font or icon set in mind, you can skip this folder.

Image

Theme-related images are to be stored in the `/img` folder.

You can create sub-folder for a better organization. For instance, the default theme has the following subfolders:

- `/icon` for simple icons (for instance, those not available in your chosen font set).
- `/jquery` for jQuery-specific images.

You can create more if needed.

JavaScript

JavaScript files are to be stored in the `/js` folder.

Unlike CSS files, we recommend you NOT to have a common/global JavaScript file, nor should you have a single file per controller.

Language

All the translation files are to be stored in the `/lang` folder.

Files should be named after their ISO 3166-1 alpha-2 code in lowercase: for instance, `fr.php`.

These files should be generated by PrestaShop integrated translation tool (located in the Localization / Translations menu).

The mobile theme

> The default theme in PrestaShop 1.6 is fully responsive, meaning that it adapts itself to any screen size.
>
> Your own theme should be responsive too! If not, then you should build an alternative theme targeted at smaller screen – or use/adapt the one available in PrestaShop 1.5's default theme.

PrestaShop has a mobile theme option in its back office: in the Preferences / Themes page, the "Mobile" tab in the "Your current theme" section gives you the following choices: disable the option, or enable it for smartphones, tablets or both.

Once this option is enabled, the theme that is displayed to the mobile visitor is not the default desktop theme but the alternative theme that is located in the `/mobile` folder: it is better suited for small screen sizes, and therefore your customers will appreciate the difference.

In essence, the content of the `/mobile` folder is another complete PrestaShop theme: it has the same overall file structure with its own `/css`, `/img` and `/js` folders, and its own template files.

Files

Templates files

PrestaShop uses the Smarty template engine for its theme system. Smarty makes it possible to separate content (the information being presented) from presentation (the way the information is displayed). The template file mixes both in order to generate a fully-formed HTML file.

A template file is built with two types of block of code:

- Code that does not change throughout the HTML rendering process: mostly design sections, and some immutable content (logo, menu, links, etc.).
- Code that does change depending on the context of the rendered page: variables in the code are replaced with the actual content that is expected by the visitor in this context.

Note that you can generate more than just HTML pages with Smarty: XML files, text files, email file, etc.

See for instance the constant and variable blocks in 404.tpl, the template file displayed when PrestaShop needs to send a 404 File Not Found error message:

```
<div>
        <div>
            <img src="{$img_dir}/img-404.jpg" alt="{l s='Page not found'}" />
    </div>

        <h1>{l s='This page is not available'}</h1>
        <p>
                {l s='We\'re sorry, but the Web address you\'ve entered is no longer available.'}
        </p>

        <h3>{l s='To find a product, please type its name in the field below.'}</h3>
        <form action="{$link->getPageLink('search')}" method="post">
                <div>
                        <label for="search_query">{l s='Search our product catalog:'}</label>
                        <input id="search_query" name="search_query" type="text" />
                    <button type="submit" name="Submit" value="OK">{l s='Ok'}</button>
                </div>
        </form>

        <div>
                <a href="{$base_dir}" title="{l s='Home'}">{l s='Home page'}</a>
        </div>
</div>
```

(this is a simplified version of the real template file, which you can find here: https://github.com/PrestaShop/PrestaShop/blob/1.6/themes/default-bootstrap/404.tpl)

People familiar with HTML (which you should be if you intend to build a PrestaShop theme) will immediately notice some `{$tag_name}` tags in the regular HTML content. These are PrestaShop's Smarty variables.

There are already a few interesting variables here:

- `{$img_dir}` returns the absolute file path for the `/img` folder.
- `{l s='My text'}` is a special method for strings that need to be translated. Every string should be encapsulated in a `{l s='...'}` tag.
- `{$link->getPageLink('search')}` returns the absolute file path to another template file, in this case the `search.tpl` file.
- `{$base_dir}` returns the absolute file path to the root of PrestaShop's folder – and therefore, to the home page.

PrestaShop uses the Smarty 3 engine. You can learn more about Smarty and its syntax here: http://www.smarty.net/docs/en/smarty.for.designers.tpl.

Style sheets

Template files render into HTML files, with no styling (except for inline styles, if any), which means that the blocks of content are displayed as-is, bare bones, one block after the other. This is where style sheets (CSS files) are useful: they are here to redefine the way the blocks of content are displayed, sometimes even rearranging whole portions of the page in order to make it look better. Font, margin, columns and many other design aspects can be recomposed using CSS.

You can create and edit CSS files any way you want, making sure that they are stored in the `/css` folder.
It is recommended to have a common style sheet for global CSS rules: `global.css`.
Then, each of the controller should have its own CSS file: for instance, `product.css` for the Product page.

If you are starting from the default theme's style sheet files, you should rather edit the corresponding Sass files in the `/sass` folder, then generate the new CSS files and store it in the `/css` folder. This ensures consistency between CSS and Sass files.

Here is an example of a Sass file:

Filename: /sass/product.scss

```
.primary_block {
  margin-bottom: 40px;
}
.top-hr{
  background: $top-line-color;
  height: 5px;
  margin: 2px 0 31px;
}
```

...which gets rendered into these CSS lines:

Filename: /css/product.css

```
/* line 6, ../sass/product.scss */
.primary_block {
  margin-bottom: 40px;
}
/* line 9, ../sass/product.scss */
.top-hr {
  background: #c4c4c4;
  height: 5px;
  margin: 2px 0 31px;
}
```

As you can see, the `$top-line-color` variable in the Sass file turns into the `#c4c4c4` value in the rendered CSS file. Sass variables in the default PrestaShop 1.6 theme are stored in the `_theme_variables.scss` file.

Image files

The images used by the theme should be stored in its `/img` folder (and subfolders for specific cases, for instance `/img/icon` for Gif icons and `/img/jquery` for jQuery-specific images).
You can use pretty much any kind of image you wish when creating your design.

In terms of icons, PrestaShop uses the Font Awesome font set, stores in the `/font` folder. Using a font for icons has many advantages:

- A single file for many different icons.
- Many possible variations: size, color, shades, rotation, etc.
- Display equally great on all screen sizes and resolutions: PC, TV screen, Retina, etc.

Tools

As an aside, you need a solid IDE (and a good knowledge of it) in order to quickly locate the needed file with a grep-like tool.

You also need to get acquainted with pre-compilation tools, to make your life easier.