p&k PRIX

**HELLO!
THANKS FOR
PURCHASING OUR
THEME!**

parker&kent

* For this help file we will refer to the skin_1 files.

Welcome!
Are included in this package:

psd

This folder contains all the layered psd files of the 10 skins of this template.

contents

This folder contains all the assets used within the template previews, with the exception of the style elements such icons, background images, etc. that are located inside the assets folder of each skin.

skin_1 - 10

These folders contain all the files of the skins.

Every skin folder contains a complete and different version of the template. To publish your website just upload on your server all the files and folders that you find into the skin folder that you want to use.

Every skin folder contains these files and folders:

css folder: The css folder contains the skin.css file. This file handles all the colors of the template. Check the file comments to identify the styles of every html page element.

assets folder: The assets folder contains all the images and icons used into the template.

js folder: The js folder contains all the javascript files used into the template.

php folder: The php folder contains the sendMail.php file, used by the contact form of the template.

prettyPhoto folder: The prettyPhoto folder contains all the scripts and assets of the lightbox components.

html files: This template comes with 15 html pages to present your contents. The style.css file handles the layout of the template.

2

The Logo:

To edit the logo you can simply replace the logo.png file into the assets folder, if you intend to use a png file. If you want to use another image, edit its dimensions or its path you need to edit the lines 81 and 82 of the css/skin.css file (Code 1).

```
.pk_logo, .pk_logo a{ width:100px; height:90px; }
.pk_logo{ background:transparent url("../assets/logo.png") no-repeat 0px 0px; }
```

Code 1

In addition to the logo you can add a short description editable in the html of each page. To change the text just edit the contents of the h2 tag at line 41 (Code 2).

```
<h2 class="pk_site_description">Optional site<br />description here</h2>
```

Code 2

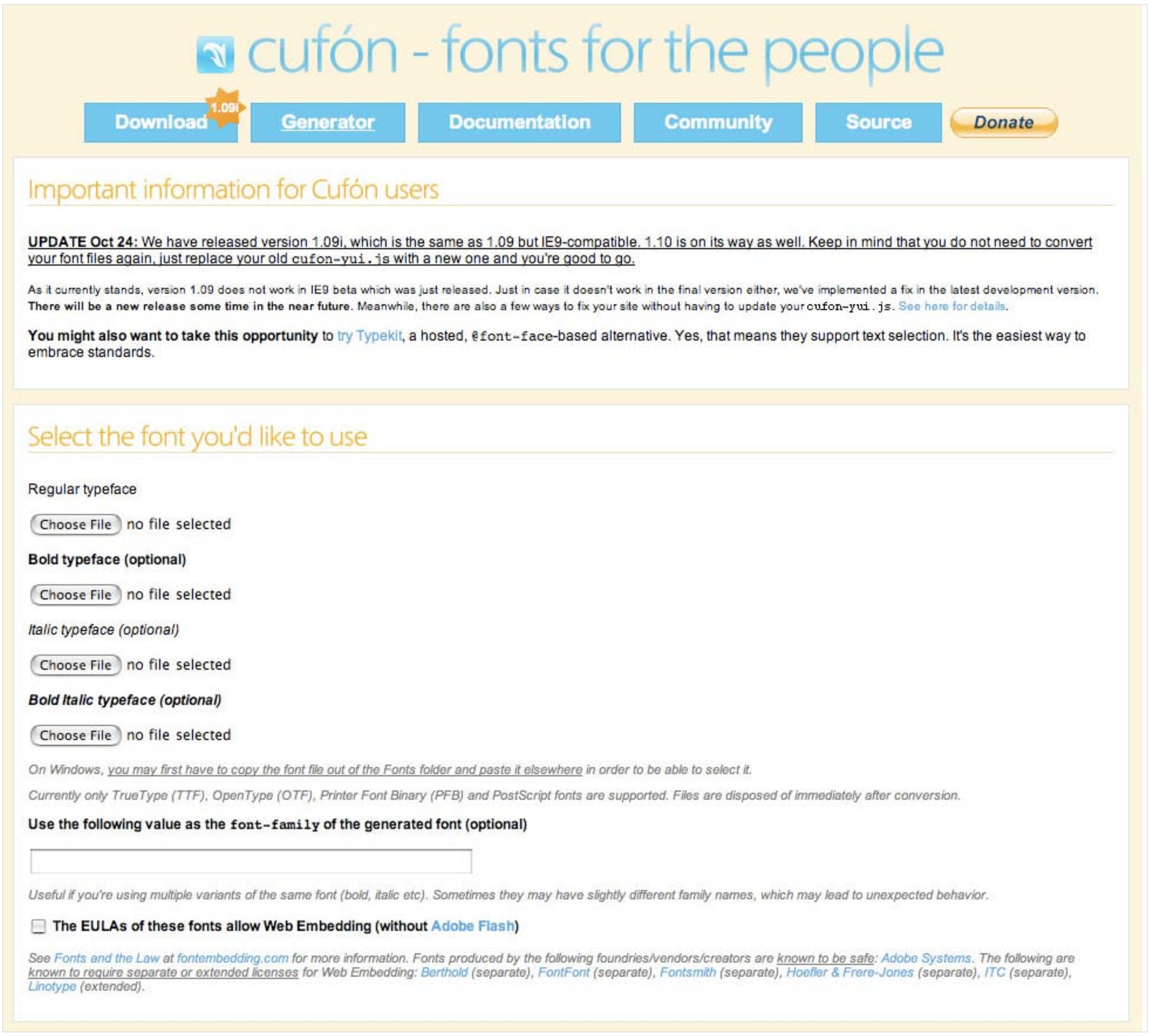
3

The Font:

For all the template’s titles, Prix uses the Cufon font replacement. To edit or change this font you need to follow these simple steps.
The first step is to chose the font that you like more. You can chose a font from your personal library or download one from the many collections of free fonts available on the web.

Once you’ve selected the font to use go to this web page:

http://cufon.shoqolate.com/generate/



Important information for Cufón users

UPDATE Oct 24: We have released version 1.09i, which is the same as 1.09 but IE9-compatible. 1.10 is on its way as well. Keep in mind that you do not need to convert your font files again, just replace your old `cufon-yui.js` with a new one and you're good to go.

As it currently stands, version 1.09 does not work in IE9 beta which was just released. Just in case it doesn't work in the final version either, we've implemented a fix in the latest development version. **There will be a new release some time in the near future.** Meanwhile, there are also a few ways to fix your site without having to update your `cufon-yui.js`. See here for details.

You might also want to take this opportunity to try Typekit, a hosted, `@font-face`-based alternative. Yes, that means they support text selection. It's the easiest way to embrace standards.

Select the font you'd like to use

Regular typeface

Choose File no file selected

Bold typeface (optional)

Choose File no file selected

Italic typeface (optional)

Choose File no file selected

Bold Italic typeface (optional)

Choose File no file selected

On Windows, you may first have to copy the font file out of the Fonts folder and paste it elsewhere in order to be able to select it.

Currently only TrueType (TTF), OpenType (OTF), Printer Font Binary (PFB) and PostScript fonts are supported. Files are disposed of immediately after conversion.

Use the following value as the `font-family` of the generated font (optional)

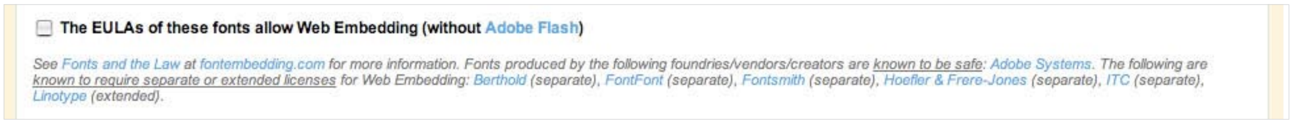
Useful if you're using multiple variants of the same font (bold, italic etc). Sometimes they may have slightly different family names, which may lead to unexpected behavior.

☐ **The EULAs of these fonts allow Web Embedding (without Adobe Flash)**

See Fonts and the Law at fontembedding.com for more information. Fonts produced by the following foundries/vendors/creators are known to be safe: Adobe Systems. The following are known to require separate or extended licenses for Web Embedding: Berthold (separate), FontFont (separate), Fontsmith (separate), Hoefler & Frere-Jones (separate), ITC (separate), Linotype (extended).

Image 1

Upload the font and then check the checkbox (Image 2):

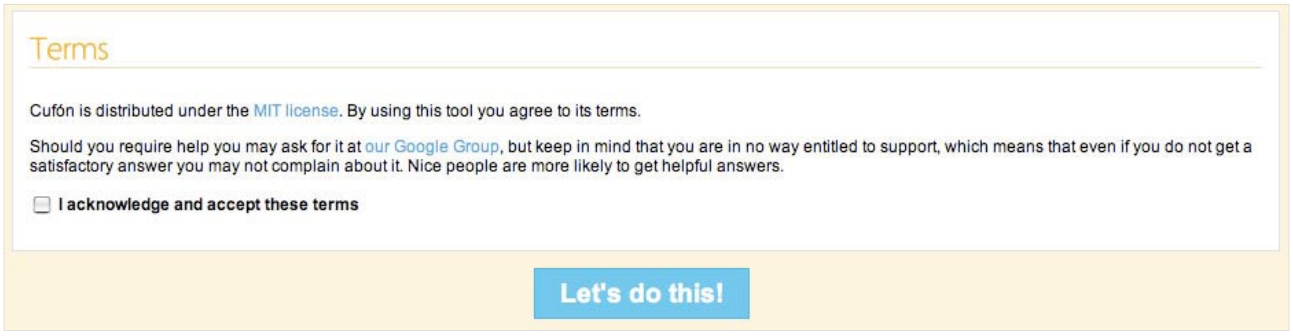


☐ **The EULAs of these fonts allow Web Embedding (without Adobe Flash)**

See Fonts and the Law at fontembedding.com for more information. Fonts produced by the following foundries/vendors/creators are known to be safe: Adobe Systems. The following are known to require separate or extended licenses for Web Embedding: Berthold (separate), FontFont (separate), Fontsmith (separate), Hoefler & Frere-Jones (separate), ITC (separate), Linotype (extended).

Image 2

Below the previous form you will find a list of checkbox. Each one represents a set of characters and punctuation, just check the right ones needed by your specific language. Once you've made your selection check the checkbox to accept the terms of use and click on the "Let's do this!" button. (Image 3)



Terms

Cufón is distributed under the MIT license. By using this tool you agree to its terms.

Should you require help you may ask for it at our Google Group, but keep in mind that you are in no way entitled to support, which means that even if you do not get a satisfactory answer you may not complain about it. Nice people are more likely to get helpful answers.

☐ **I acknowledge and accept these terms**

Image 3

Download the generated font and rename it as "font.js"; replace the original font file of the template inside the "js" folder with the new one. Enjoy your new font!

The Backgrounds:

Prix Template gives you the option to customize the skins with two backgrounds.

Main background:

```
body{ color:#848783; background-color:#fff; }
```

Code 5

To set the main background color of the template you need to edit the line 5 (the body tag style) into the css/skin.css (Code 5).

Wrapper background:

```
#pk_wrapper_all{
        background:#252b24 url("../assets/back_grounds/back_ground_pattern.jpg") repeat 0px 0px;
}

#pk_wrapper{
        background:transparent url("../assets/back_grounds/wrapper_back_ground.jpg") repeat-x 0px 0px;
}
```

Code 6

To set the wrapper background image or color of the template you need to edit lines 7 and 8 into the css/skin.css file (Code 6). You can simply overwrite the default background image inside the assets/back_grounds folder or, if you want to use an image located in a different folder, you have to edit the path of the image.

The Menu:

To edit the main menu colors you need to edit lines 89 and 90 inside the css/skin.css file (Code 7). At line 89 you can edit the color of the menu buttons labels text. At line 90 you can edit their roll over color.

```
nav a{ color:#808080; }
nav a:hover{ color:#555; }
```

Code 7

At lines 96 and 97 (Code 8) you can edit the properties of the selected menu button and its sub menu, that, into the html pages code, is identified by the class=”current”.

```
nav .current a{ color:#000; }
nav .current li a{ color:#808080; }
```

Code 8

Submenu:

```
nav ul ul{ border:solid 1px #e0e0e0; background-color:#fff; }
nav ul li li{ border-bottom:solid 1px #e0e0e0; }
nav ul li li a{ color:#808080; }
nav ul li li a:hover{ color:#555; }
```

Code 9

At line 91 (Code 9) of the css file css/skin.css you can change the border’s color and the background’s color of the submenu, at line 92 the bottom border of the submenu buttons, at line 93 the label color of the buttons and at the line 94 the label color for the rollover.

Menu structure:

```
<nav>
  <ul>
    <li class=”current”><a href=”index.html” title=””>Home</a>
      <ul>
        <li><a href=”index.html” title=””>Home With Media Gallery</a></li>
        <li><a href=”index_nivo_slider.html” title=””>Home With Nivo Slider</a></li>
      </ul>
    </li>
    <li><a href=”galleries.html” title=””>Galleries</a>
      <ul>
        <li><a href=”galleries.html” title=””>Galleries Default Page</a></li>
        <li><a href=”galleries_three_columns.html” title=””>Galleries Three Columns</a></li>
        <li><a href=”galleries_four_columns.html” title=””>Galleries Four Columns</a></li>
        <li><a href=”galleries_sidebar.html” title=””>Galleries With Sidebar</a></li>
        <li><a href=”galleries_single.html” title=””>Galleries Single Page</a></li>
      </ul>
    </li>
    <li><a href=”standard_page.html” title=””>Pages</a>
      <ul>
        <li><a href=”standard_page.html” title=””>About Us</a></li>
        <li><a href=”news.html” title=””>News</a></li>
        <li><a href=”services.html” title=””>Services</a></li>
        <li><a href=”archive.html” title=””>Archive</a></li>
        <li><a href=”full_width.html” title=””>Full Width</a></li>
      </ul>
    </li>
    <li><a href=”blog.html” title=””>Blog</a>
      <ul>
        <li><a href=”blog.html” title=””>Blog</a></li>
        <li><a href=”blog_post.html” title=””>Blog Post</a></li>
      </ul>
    </li>
    <li><a href=”../skin_1” title=””>Skins</a>
      <ul>
        <li><a href=”../skin_1” title=””>Skin One</a></li>
        <li><a href=”../skin_2” title=””>Skin Two</a></li>
        <li><a href=”../skin_3” title=””>Skin Three</a></li>
        <li><a href=”../skin_4” title=””>Skin Four</a></li>
        <li><a href=”../skin_5” title=””>Skin Five</a></li>
        <li><a href=”../skin_6” title=””>Skin Six</a></li>
        <li><a href=”../skin_7” title=””>Skin Seven</a></li>
        <li><a href=”../skin_8” title=””>Skin Eight</a></li>
        <li><a href=”../skin_9” title=””>Skin Nine</a></li>
        <li><a href=”../skin_10” title=””>Skin Ten</a></li>
      </ul>
    </li>
    <li><a href=”columns.html” title=””>All Styles</a>
      <ul>
        <li><a href=”columns.html” title=””>Columns</a></li>
        <li><a href=”typography.html” title=””>Typography</a></li>
        <li><a href=”images.html” title=””>Images</a></li>
        <li><a href=”components.html” title=””>Components</a></li>
      </ul>
    </li>
    <li><a href=”contact.html” title=””>Contact</a></li>
  </ul>
</nav>
```

Code 10

In every page the menu is structured like in the previous block of code (Code 10).

The menu has a tree structure, based on the <ul></ul> and <li></li> tags.
The main menu is contained by the <nav></nav> tags.
The <ul></ul> tag can contain unlimited <li></li> items.

Every <li></li> item is a menu button, like this one:

```
<li><a href=”news.html” title=””>News</a></li>
```

Code 11

To create a sub menu you have to nest a new <ul></ul> into the <li></li> item, like this:

```
<li><a href=”standard_page.html” title=””>Pages</a>
  <ul>
    <li><a href=”standard_page.html” title=””>About Us</a></li>
    <li><a href=”news.html” title=””>News</a></li>
    <li><a href=”services.html” title=””>Services</a></li>
    <li><a href=”archive.html” title=””>Archive</a></li>
    <li><a href=”full_width.html” title=””>Full Width</a></li>
  </ul>
</li>
```

Code 12

In this example the Pages button of the main menu has one sub menu button (Code 12).

```
<li class=”current”><a href=”index.html” title=””>Home</a></li>
```

Code 13

The class=”current”, as seen before in this help, is needed only to highlight the selected menu button.

This is it, very simple. You can nest sub menus without limitation, so you can create sub menus at any level.

The Galleries:

The Prix Template gives you the option to choose from 5 different types of galleries. Every portfolio offers a different way to showcase your works and projects.

All the gallery pages, but the single gallery, are grid based, so the process for the creation of their contents is always the same.

```
<div class=”pk_one_fourth”>
  <div class=”pk_entry_grid”>
    <div class=”pk_image”>
      <div class=”pk_image_wrapper pk_zoom_icon”>
        <a href=”path/image.jpg” title=”Image title” rel=”prettyPhoto[gallery]”>
          <img src=”../contents/galleries_four_columns/01.jpg” alt=”” />
          <span class=”pk_image_button_back_ground”></span>
          <span class=”pk_image_button_icon”></span>
        </a>
      </div>
      <img src=”assets/images_shadow.png” class=”pk_image_shadow” />
    </div>
    <h3><a href=”galleries_single.html” title=””>Image title...</a></h3>
    <p>Gallery caption...</p>
    <a href=”galleries_single.html” title=”” class=”pk_small_button pk_light_button”>Read more</a>
  </div>
```

Code 14

The previous block of code (Code 14) represents a grid item of the four columns grid gallery. For every gallery the structure of the page is the same, the only thing that changes is the class of the main container div of each grid item. In this case we have “pk_one_fourth”, for the three columns grids we would have “pk_one_third” and for the two columns grids we would have “pk_one_half”.

In this example (Code 14) the item (eg thumb) is connected to the prettyPhoto plugin. Always remember that the “rel” attribute identifies a gallery, so, if all the items (eg thumbs) have the same gallery id (eg [gallery]), when prettyPhoto is open you will be able to browse all the big images of the related gallery with a prev/next navigation.

```
<a href=”path/image.jpg” title=”Image title” rel=”prettyPhoto[gallery]”>
  <img src=”../contents/galleries_four_columns/01.jpg” alt=”” />
  <span class=”pk_image_button_back_ground”></span>
  <span class=”pk_image_button_icon”></span>
</a>
```

Code 15

Adding a video, for example a YouTube one:

```
<a href=”http://www.youtube.com/watch?v=GUpSNVpjw7A&amp;hd=1” title=”Video title” rel=”prettyPhoto[gallery]”>
  <img src=”../contents/galleries_four_columns/01.jpg” alt=”” />
  <span class=”pk_image_button_back_ground”></span>
  <span class=”pk_image_button_icon”></span>
</a>
```

Code 16

or a Vimeo one:

```
<a href=”http://vimeo.com/15727210” title=”Video title” rel=”prettyPhoto[gallery]”>
  <img src=”../contents/galleries_four_columns/01.jpg” alt=”” />
  <span class=”pk_image_button_back_ground”></span>
  <span class=”pk_image_button_icon”></span>
</a>
```

Code 17

For all the other types of contents usable with prettyPhoto we suggest you to check the official website for reference:

http://www.no-margin-for-errors.com/projects/prettyphoto-jquery-lightbox-clone/

The grid can have an unlimited number of items (eg thumbs). The only important thing to remember is to apply the class=”pk_last” to the last item (eg thumb) of the row. This class basically doesn’t add the right margin to the last item.

You should apply this class in this way:

```
<div class=”pk_one_fourth pk_last”>
  <div class=”pk_entry_grid”>
    <div class=”pk_image”>
      Box content...
  </div>
</div>

<span class=”pk_clear_both”></span>
```

Code 18

After each row of thumbnails you need to add this element:
<span class=”pk_clear_both”></span> to have a perfect alignment of the grid items, in case one or more grid items have a shorter/longer description text.

NOTE: To the grid thumbnails you can add the following classes: “pk_zoom_icon”, “pk_play_icon”, “pk_page_icon”, “pk_link_icon” depending on which icon type you want to show on roll over (Code 14)

The Images:

```
<div class=”pk_image pk_image_medium pk_alignleft”>
  <div class=”pk_image_wrapper pk_zoom_icon”>
    <a href=”#” title=””>
      <img src=”../contents/galleries_four_columns/01.jpg” alt=”” />
      <span class=”pk_image_button_back_ground”></span>
      <span class=”pk_image_button_icon”></span>
    </a>
  </div>
  <img src=”assets/images_shadow.png” class=”pk_image_shadow” />
</div>
```

Code 19

The previous block of code (Code 19) represents the html markup that you need to use to create a clickable image of a medium size (in this case with a left alignment).

The images can have any size. The template has four css classes that define four different default sizes. These classes should be applied to the div with the class “pk_image_wrapper”. In the previous block of code (Code 19) we have used the “pk_image_medium” class.

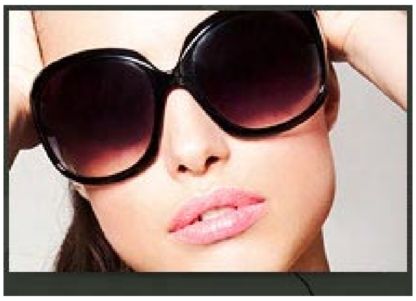
The default sizes are defined by the following css classes:

pk_image_full: 940px - for the pages without the sidebar
pk_image_large: 610px - for the pages that have the sidebar
pk_image_medium: 214px
pk_image_small: 60px - for the thumbnails

As described in the previous section of this help file, every image, if clickable, can show an icon on roll over.

These are the available icons:

simple image (no icon):

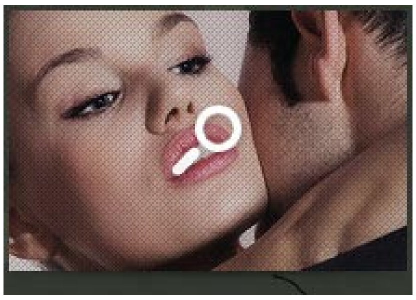


Output

```
<div class=”pk_image pk_image_medium pk_alignleft”>
  <div class=”pk_image_wrapper”>
    <img src=”../contents/galleries_four_columns/07.jpg” alt=”” />
  </div>
  <img src=”assets/images_shadow.png” class=”pk_image_shadow” />
</div>
```

Html Code

zoom icon:

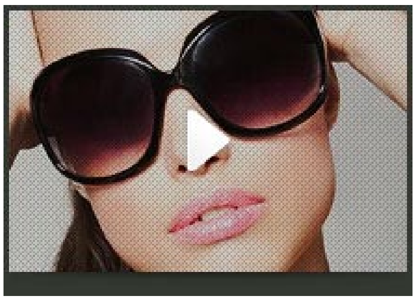


```
<div class="pk_image pk_image_medium pk_alignleft">
  <div class="pk_image_wrapper pk_zoom_icon">
    <a href=" path/image.jpg" title="" rel="prettyPhoto[id]">
      <img src="../contents/galleries_four_columns/01.jpg" alt="" />
      <span class="pk_image_button_back_ground"></span>
      <span class="pk_image_button_icon"></span>
    </a>
  </div>
  <img src="assets/images_shadow.png" class="pk_image_shadow" />
</div>
```

Output

Html Code

play icon:

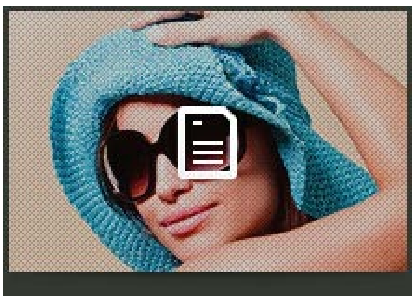


```
<div class="pk_image pk_image_medium pk_alignleft">
  <div class="pk_image_wrapper pk_play_icon">
    <a href="video_link" title="" rel="prettyPhoto[id]">
      <img src="../contents/galleries_four_columns/02.jpg" alt="" />
      <span class="pk_image_button_back_ground"></span>
      <span class="pk_image_button_icon"></span>
    </a>
  </div>
  <img src="assets/images_shadow.png" class="pk_image_shadow" />
</div>
```

Output

Html Code

page icon:



```
<div class="pk_image pk_image_medium pk_alignleft">
  <div class="pk_image_wrapper pk_page_icon">
    <a href="page_link" title="">
      <img src="../contents/galleries_four_columns/03.jpg" alt="" />
      <span class="pk_image_button_back_ground"></span>
      <span class="pk_image_button_icon"></span>
    </a>
  </div>
  <img src="assets/images_shadow.png" class="pk_image_shadow" />
</div>
```

Output

Html Code

link icon:

```
<div class="pk_image pk_image_medium pk_alignleft">
  <div class="pk_image_wrapper pk_link_icon">
    <a href="http://www.google.com" title="">
      <img src="../contents/galleries_four_columns/04.jpg" alt="" />
      <span class="pk_image_button_back_ground"></span>
      <span class="pk_image_button_icon"></span>
    </a>
  </div>
  <img src="assets/images_shadow.png" class="pk_image_shadow" />
</div>
```

Output

Html Code

NOTE: for more info about the usage of prettyPhoto check the "The Galleries" section.

8

The Components:

Prix Template includes some cool components, very useful for showcasing any type of content. You can chose from three different sliders, boxes, tabs and toggles.

a The Media Slider:

Media Slider structure:

```
<div id=”pk_media_slider” class=”pk_media_slider”>
  <div class=”pk_slider_content”>

    <!--Start media slider item-->
    <div class=”pk_slider_item”>
      <div class=”pk_slider_item_media”>
        Image or Video...
      </div>
      <div class=”pk_slider_item_info”>
        <div class=”pk_slider_item_info_background”></div>
        <div class=”pk_slider_item_info_content”>
          Info text...
        </div>
      </div>
    </div>
    <!--End media slider item-->

  </div>
  <img src=”assets/images_shadow.png” class=”pk_image_shadow” />
</div>
<script type=”text/javascript”>
  jQuery(“#pk_media_slider”).pk_media_slider({
    sliderWidth:610,
    sliderHeight:340,
    sliderPadding:0,
    sliderBackgroundColor:”#1a1f1a”,

    buttonInfoLabelColor:”#808080”,
    buttonInfoOpenLabel:”info”,
    buttonInfoCloseLabel:”close info”,

    infoTextAlign:”left”,
    infoTextColor:”#ccc”,
    infoBackgroundAlpha:0.81,
    infoBackgroundColor:”#000”,

    slideshow:true,
    slideshowAutoStart:false,
    slideshowInterval:5
  });
</script>
```

Code 20

Add image:

```
<!--Start media slider item-->
<div class="pk_slider_item">
  <div class="pk_slider_item_media">
    <div class="pk_image">
      <div class="pk_image_wrapper">
        <img src="../contents/sliders/pages/02.jpg" alt="" />
      </div>
    </div>
  </div>
  <div class="pk_slider_item_info">
    <div class="pk_slider_item_info_background"></div>
    <div class="pk_slider_item_info_content">
      <p>Consetetur sadipscing elitr,  sed diam nonumy eirmod tempor invidunt ut labore</p>
    </div>
  </div>
</div>
<!--End media slider item-->
```

Code 21

Add image with prettyphoto:

```
<!--Start media slider item-->
<div class="pk_slider_item">
  <div class="pk_slider_item_media">
    <div class="pk_image">
      <div class="pk_image_wrapper pk_zoom_icon">
        <a href=" path/image.jpg" title="Image title" rel="prettyPhoto[id]">
          <img src="../contents/sliders/pages/02.jpg" alt="" />
          <span class="pk_image_button_back_ground"></span>
          <span class="pk_image_button_icon"></span>
        </a>
      </div>
    </div>
  </div>
  <div class="pk_slider_item_info">
    <div class="pk_slider_item_info_background"></div>
    <div class="pk_slider_item_info_content">
      <p>Consetetur sadipscing elitr,  sed diam nonumy eirmod tempor invidunt ut labore</p>
    </div>
  </div>
</div>
<!--End media slider item-->
```

Code 22

Add item (image or video) without info:

```
<!--Start media slider item-->
<div class="pk_slider_item">
  <div class="pk_slider_item_media">
    <div class="pk_image">
      <div class="pk_image_wrapper">
        <img src="../contents/sliders/pages/02.jpg" alt="" />
      </div>
    </div>
  </div>
</div>
<!--End media slider item-->
```

Code 23

Add video:

```
<!--Start media slider item-->
<div class=”pk_slider_item”>
  <div class=”pk_slider_item_media”>
    <div class=”pk_video”>
      <div id=”vimeo”></div>
      <script type=”text/javascript”>
        var params = {
          allowfullscreen: “true”,
          allowscriptaccess: “always”,
          wmode: “transparent”
        };
        swfobject.embedSWF(“http://www.vimeo.com/moogaloop.swf?clip_id=15727210”, “vimeo”, “610”, “340”,
“9.0.0”, “swf/expressInstall.swf”, “”, params);
      </script>
    </div>
  </div>
  <div class=”pk_slider_item_info”>
    <div class=”pk_slider_item_info_background”></div>
    <div class=”pk_slider_item_info_content”>
      <p>Consetetur sadipscing elitr,  sed diam nonumy eirmod tempor invidunt ut labore</p>
    </div>
  </div>
</div>
<!--End media slider item-->
```

Code 24

Always remember that, to avoid conflicts, the ids of the videos divs must be unique (in the previous block of code <div id=”vimeo”></div>). So, if you intend to use on the same page more videos from YouTube, Vimeo, etc., you need to set unique ids to their container divs.

A good practice is to use sequential ids:

<div id=”vimeo_1”></div>, <div id=”vimeo_2”></div>, etc.

Parameters:

```
<script type=”text/javascript”>
  jQuery(“#pk_media_slider”).pk_media_slider({
    sliderWidth:610,
    sliderHeight:340,
    sliderPadding:0,
    sliderBackgroundColor:”#1a1f1a”,

    buttonInfoLabelColor:”#808080”,
    buttonInfoOpenLabel:”info”,
    buttonInfoCloseLabel:”close info”,

    infoTextAlign:”left”,
    infoTextColor:”#ccc”,
    infoBackgroundAlpha:0.81,
    infoBackgroundColor:”#000”,

    slideshow:true,
    slideshowAutoStart:false,
    slideshowInterval:5
  });
</script>
```

Code 25

The previous block of code (Code 25) is necessary to init the slider and to pass to it all the parameters for its customization.
Also for the sliders divs, to avoid conflicts, is necessary to use sequential and unique ids (as we've seen with the previous vimeo videos example). Check the yellow highlights in the block of code (Code 20) to see the slider id that needs to be unique.

b The Nivo Slider:

Nivo Slider structure:

```
<div class="slider-wrapper">
  <div id="nivo_slider" class="nivoSlider">
    <img src="../contents/sliders/pages/01.jpg" alt="" />
    <img src="../contents/sliders/pages/02.jpg" alt="" title="This is the title" />
    <img src="../contents/sliders/pages/03.jpg" alt="" />
    <img src="../contents/sliders/pages/04.jpg" alt="" title="#htmlcaption" />
  </div>
  <img src="assets/images_shadow.png" class="pk_image_shadow" />
  <div id="htmlcaption" class="nivo-html-caption">
    <strong>This</strong> is an example of a <em>HTML</em> caption with <a href="#">a link</a>.
  </div>
  <script type="text/javascript">
    //jQuery(window).load(function() {
      jQuery('#nivo_slider').nivoSlider();
    //});
  </script>
</div>
```

Code 26

The Nivo slider supports only images. To add images to the Nivo slider you simply have to add <img> tags within its main container div.

```
<img src="../contents/sliders/pages/02.jpg" alt="" title="This is the title" />
```

Code 27

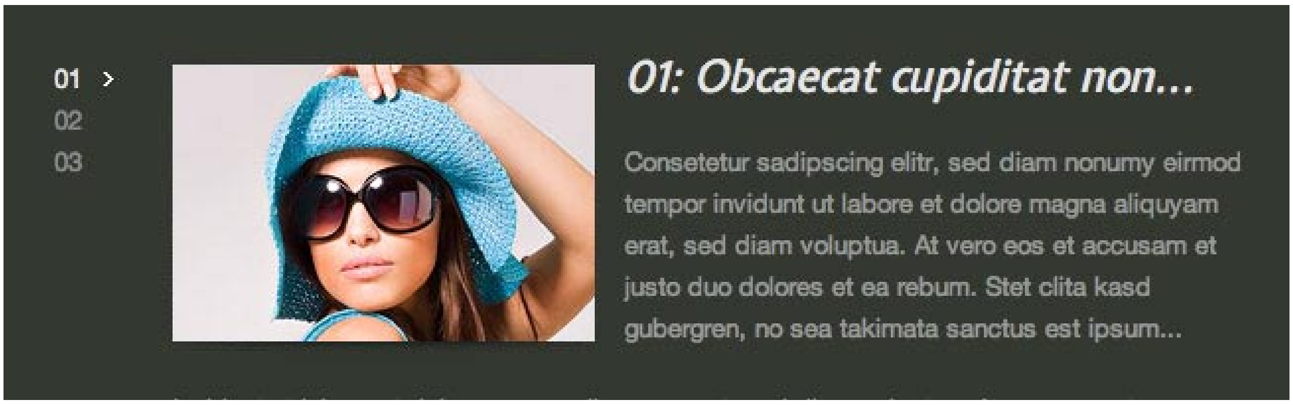
To edit the size of the Nivo slider open the style.css file and edit its width and height at the lines 597 and 599 (Code 28).

```
.slider-wrapper{ width:610px; height:370px; margin-bottom:20px; }
.slider-wrapper .pk_image_shadow{ display:block; margin-top:0px; }
.nivoSlider{ position:relative; width:610px; height:340px; background:transparent
url('assets/nivo_slider/loading.gif') no-repeat 50% 50%; }
```

Code 28

NOTE: In order to have enough space for the slider navigation, the "slider-wrapper" div must be at least 30px higher than the Nivo slider div.

C The Tour:



01: Obcaecat cupiditat non...

Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est ipsum...

Tour structure:

```
<div id="pk_tour" class="pk_tour">
  <ul class="pk_tour_navigation">
    <li><a href="#" title="">01</a></li>
    <li><a href="#" title="">02</a></li>
    <li><a href="#" title="">03</a></li>
  </ul>
  <div class="pk_tour_content">
    <ul class="pk_tour_content_items">

      <!--Start tour item-->
      <li class="pk_tour_content_item">
        <p>Content...</p>
      </li>
      <!--End tour item-->

      <!--Start tour item-->
      <li class="pk_tour_content_item">
        <p>Content...</p>
      </li>
      <!--End tour item-->

      <!--Start tour item-->
      <li class="pk_tour_content_item">
        <p>Content...</p>
      </li>
      <!--End tour item-->

    </ul>
  </div>
</div>
<script type="text/javascript">
  jQuery("#pk_tour").pk_tour({
    tour_width:610,
    navigation_width:30,
    navigation_align:"left",
    navigation_margin_left:0,
    navigation_margin_right:30
  });
</script>
```

Code 29

The structure of this component is based on two lists, the first one is used by the navigation controls, the second one for the contents.

This is the markup for each navigation button:

```
<li><a href=”#” title=””>01</a></li>
```

Code 30

This is the markup for each content:

```
<!--Start tour item-->
<li class=”pk_tour_content_item”>
  <p>Content...</p>
</li>
<!--End tour item-->
```

Code 31

The list of the buttons and the list of the contents must have the same number of elements and the same order. The first button in the list opens the first content of the second list.

Parameters:

```
<script type=”text/javascript”>
  jQuery(“#pk_tour”).pk_tour({
    tour_width:610,
    navigation_width:30,
    navigation_align:”left”, // left or right
    navigation_margin_left:0,
    navigation_margin_right:30
  });
</script>
```

Code 32

The previous block of code (Code 32) is necessary to init the component and to pass to it all the parameters needed for its customization. Also in this case, to avoid conflicts, the id of each component div must be unique and sequential if you want to use it multiple times in the same page (Code 29 - yellow highlights).

```
.pk_tour .pk_tour_navigation a{ color:#777; }
.pk_tour .pk_tour_navigation a.pk_left_current{ color:#e0e0e0; }
.pk_tour .pk_tour_navigation a.pk_right_current{ color:#e0e0e0; }
```

Code 33

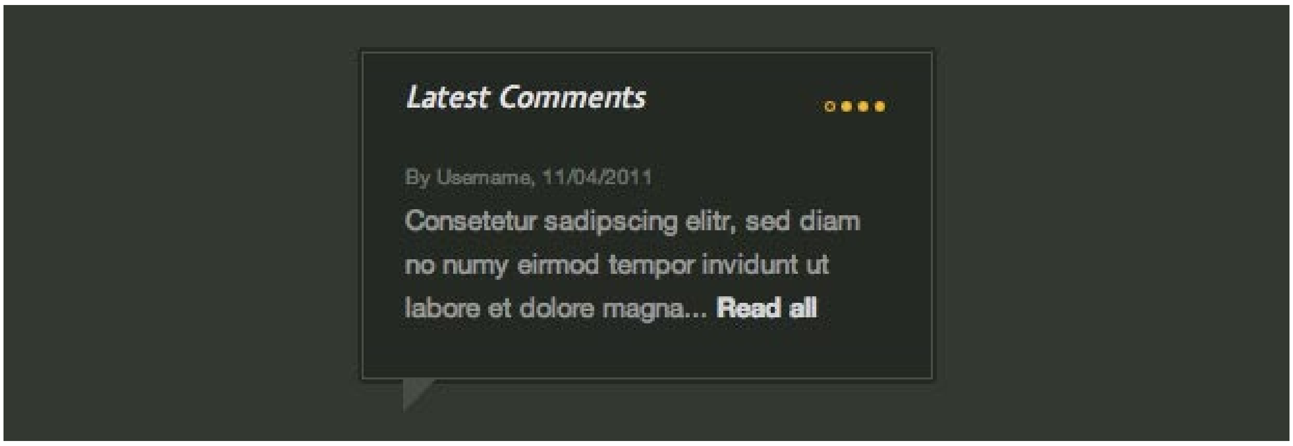
From line 259 to line 261 of the css/skin.css file (Code 33) you can edit the colors of the component elements. At line 259 is defined the color of the navigation buttons and at the lines 260-261 the colors of the selected button.

NOTE: The Tour component is used as slider for the images of the single gallery template. Considering that the component container adapts its height to the height of the selected content, it's necessary to set the height of each content item, like in the following example (Code 34):

```
<li class="pk_tour_content_item">
  <img src="../contents/single_gallery/01.jpg" alt="" width="656" height="705" />
  <p class="pk_image_title">Title...</p>
</li>
```

Code 34

d The Mini Slider:


Latest Comments
By Usemame, 11/04/2011
Consetetur sadipscing elitr, sed diam no numy eirmod tempor invidunt ut labore et dolore magna... Read all

Mini Slider structure:

```
<div id="pk_mini_slider" class="pk_mini_slider pk_dark_skin">
  <div class="pk_mini_slider_content">
    <h4>Title...</h4>
    <ul>
      <li>content...</li>
      <li>Content...</li>
    </ul>
  </div>
  <span class="pk_arrow"></span>
</div>
<script type="text/javascript">
  jQuery("#pk_mini_slider").pk_mini_slider({
    slider_width:275, // Number
    slider_height:156, // Number or "auto"
  });
</script>
```

Code 35

The Mini Slider is a simple but very useful component to showcase small contents like: news, latest works, etc. To add a content to it you just need to add a list item (Code 36):

```
<li>Content...</li>
```

Code 36

Parameters:

```
<script type=”text/javascript”>
  jQuery(“#pk_mini_slider”).pk_mini_slider({
    slider_width:275, // Number
    slider_height:156, // Number or “auto“
  });
</script>
```

Code 37

The previous block of code (Code 37) is necessary to init the Mini Slider and to pass to it the parameters for its customization. Also in this case, to avoid conflicts, the id of each component div must be unique and sequential if you want to use it multiple times in the same page (Code 35 - yellow highlights).

The Mini Slider is available in two versions: light and dark. You can set its look by simply adding the classes “pk_light_skin” and “pk_dark_skin” to its container div (Code 38):

```
<div id=”pk_mini_slider” class=”pk_mini_slider pk_dark_skin”>
```

Code 38

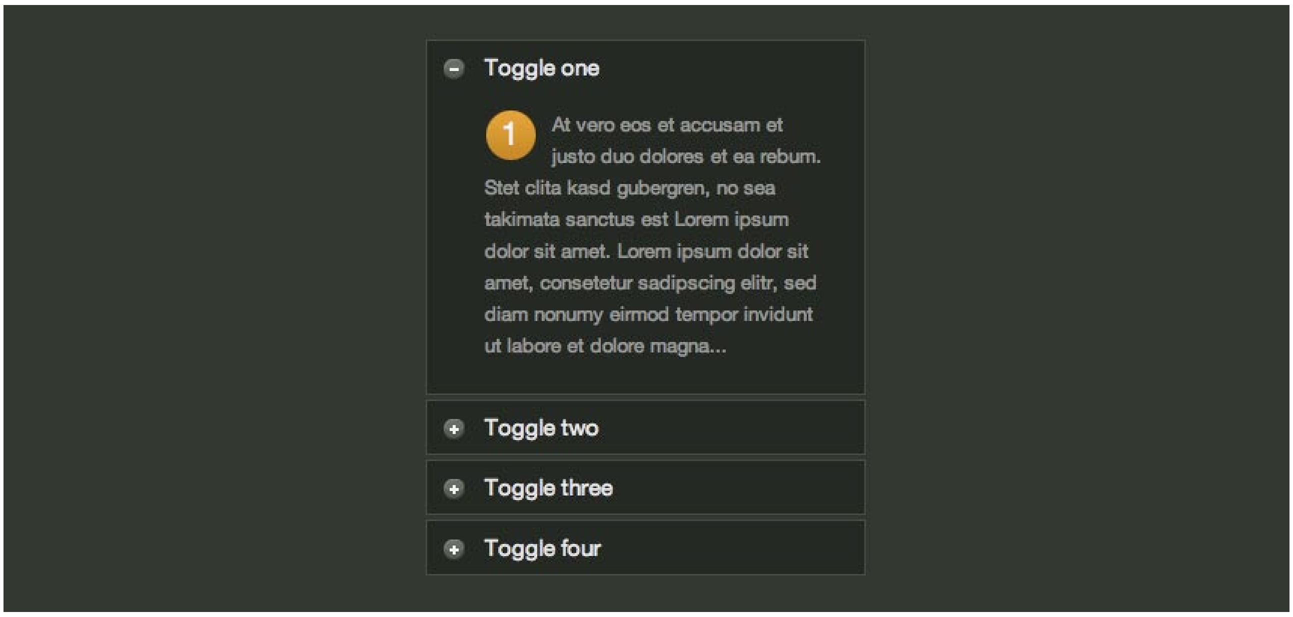
```
.pk_mini_slider.pk_light_skin{ color:#846833; background-color:#ffa800; }
.pk_mini_slider.pk_light_skin a{ color:#353d33; }
.pk_mini_slider.pk_light_skin a:hover{ color:#000; }
.pk_mini_slider.pk_light_skin h4{ color:#000; }
.pk_mini_slider.pk_light_skin .pk_mini_slider_content{ border:solid 1px #353d33; }
.pk_mini_slider.pk_light_skin .pk_mini_slider_content p.pk_meta{ color:#999; }
.pk_mini_slider.pk_light_skin .pk_arrow{ border-top-color:#353d33; }

.pk_mini_slider.pk_dark_skin{ background-color:#1a1f1a; }
.pk_mini_slider.pk_dark_skin a{ color:#ccc; }
.pk_mini_slider.pk_dark_skin a:hover{ color:#e0e0e0; }
.pk_mini_slider.pk_dark_skin h4{ color:#e0e0e0; }
.pk_mini_slider.pk_dark_skin .pk_mini_slider_content{ border:solid 1px #353d33; }
.pk_mini_slider.pk_dark_skin .pk_mini_slider_content p.pk_meta{ color:#575c55; }
.pk_mini_slider.pk_dark_skin .pk_arrow{ border-top-color:#353d33; }
```

Code 39

From line 269 to line 283 of the css/skin.css file you can edit the colors of the two skins (Code 39).

e The Toggles:



At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna...

Toggles structure:

```
<div class=”pk_toggles pk_dark_icon”>
  <div class=”pk_toggle”>
    <p class=”pk_toggle_button”>Button Label</p>
    <div class=”pk_toggle_content_wrapper” style=”width:292px;”>
      <div class=”pk_toggle_content”>Content...</div>
    </div>
  </div>

  <div class=”pk_toggle”>
    <p class=”pk_toggle_button”>Button Label</p>
    <div class=”pk_toggle_content_wrapper” style=”width:292px;”>
      <div class=”pk_toggle_content”>Content...</div>
    </div>
  </div>

  <div class=”pk_toggle”>
    <p class=”pk_toggle_button”>Button Label</p>
    <div class=”pk_toggle_content_wrapper” style=”width:292px;”>
      <div class=”pk_toggle_content”>Content...</div>
    </div>
  </div>

  <div class=”pk_toggle”>
    <p class=”pk_toggle_button”>Button Label</p>
    <div class=”pk_toggle_content_wrapper” style=”width:292px;”>
      <div class=”pk_toggle_content”>Content...</div>
    </div>
  </div>
</div>
```

Code 40

Adding/Removing an element:

```
<div class=”pk_toggle”>
  <p class=”pk_toggle_button”>Button Label</p>
  <div class=”pk_toggle_content_wrapper” style=”width:292px;”>
    <div class=”pk_toggle_content”>Content...</div>
  </div>
</div>
```

Code 41

The previous block of code (Code 41) represents a toggle element. In this case we have set a width to the toggle wrapper; setting a width isn’t strictly necessary, but we suggest to always set a width.

```
<div class=”pk_toggles pk_dark_icon”>
```

Code 42

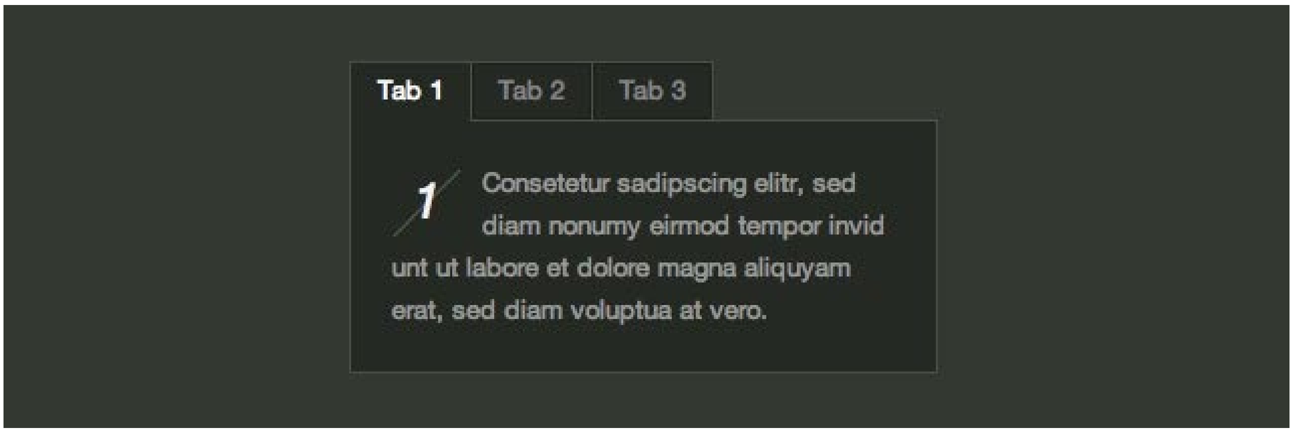
To the toggles icons can have two styles: light and dark. To set the icon style just set the “pk_light_icon” or the “pk_dark_icon” to the toggles container (Code 42).

To edit the toggles colors, edit the lines 326 and 327 of the css/skin.css file. At line 326 you can edit the border color and the background color, at line 327 you can edit the label color of the toggles buttons (Code 43).

```
.pk_toggles .pk_toggle{ border-color:#353d33; background-color:#1a1f1a; }
.pk_toggles .pk_toggle_button{ color:#e0e0e0; }
```

Code 43

f The Tabs:


Tab 1
Tab 2
Tab 3
1
Consetetur sadipscing elitr, sed diam nonumy eirmod tempor invid unt ut labore et dolore magna aliquyam erat, sed diam voluptua at vero.

Tabs structure:

```
<div class=”pk_boxed_tabs”>
  <ul class=”pk_tabs_navigation”>
    <li><a href=”#” title=””>Tab 1</a></li>
    <li><a href=”#” title=””>Tab 2</a></li>
    <li><a href=”#” title=””>Tab 3</a></li>
  </ul>
  <div class=”pk_tabs”>
    <div class=”pk_tab”>
      Content...
    </div>
    <div class=”pk_tab”>
      Content...
    </div>
    <div class=”pk_tab”>
      Content...
    </div>
  </div>
</div>
```

Code 44

To add a tab you need to add a list element to the tabs navigation list (Code 44). This is the markup (Code 45):

```
<li><a href=”#” title=””>Tab 1</a></li>
```

Code 45

Then you need to add the tab div within the tabs div (Code 46):

```
<div class=”pk_tab”>
  Content...
</div>
```

Code 46

From line 334 to line 338 of the css/skin.css file you can edit the colors of the tabs (Code 47).

```
.pk_boxed_tabs .pk_tabs_navigation li{ border-color:#353d33; background-color:#1a1f1a; }
.pk_boxed_tabs .pk_tabs_navigation a{ color:#707070; border-color:#151a15; }
.pk_boxed_tabs .pk_tabs_navigation li.pk_active_tab{ background-color:#1a1f1a; }
.pk_boxed_tabs .pk_tabs_navigation li.pk_active_tab a{ color:#fff; border-color:#1a1f1a; }
.pk_boxed_tabs .pk_tabs{ border:solid 1px #353d33; background-color:#1a1f1a; }
```

Code 47

The Contact Forms:

```
<section class="pk_contact_form">
  <h3 class="pk_heading_3">Email Us</h3>
  <form id="pk_contact_form" action="#">
    <label for="name">Your Name</label><input type="text" id="name" name="name" tabindex="1" />
    <label for="email">Your Email</label><input type="email" id="email" name="email" tabindex="2" />
    <label for="message">Your Message</label><textarea id="message" name="message" tabindex="3"></textarea>
    <input type="submit" name="submit_comment" value="Send email" class="pk_small_button pk_light_button" tabindex="4" />
  </form>
  <script type="text/javascript">
    jQuery("#pk_contact_form").pk_contact_form();
  </script>
</section>
```

Code 48

```
<section class="pk_widget">
  <h5>Email Us</h5>
  <div class="pk_contact_form pk_small_form">
    <form id="pk_contact_form_1" action="#">
      <label for="name">Your Name</label><input type="text" id="name" name="name" tabindex="1" />
      <label for="email">Your Email</label><input type="email" id="email" name="email" tabindex="2" />
      <label for="message">Your Message</label><textarea id="message" name="message" tabindex="3"></textarea>
      <input type="submit" name="submit_comment" value="Send email" class="pk_small_button pk_light_button" tabindex="4" />
    </form>
    <script type="text/javascript">
      jQuery("#pk_contact_form_1").pk_contact_form();
    </script>
  </div>
</section>
```

Code 49

The previous blocks of code (Code 48, 49) are needed to add a contact form to the html pages. Using the markup of the first block (Code 48) you will have a standard contact form, like the one present in the contact.html page of the template preview. Using the markup of the second block (Code 49) you will have a small contact form, like the one present in the sidebar of the standard_page.html page of the template preview.

Is possible to use more than one contact form in each page by simply using different ids as highlighted in yellow in the previous blocks of code (Code 48, 49).

```
input[type=submit], button{ color:#000; }
input[type=text], input[type=password], input[type=email], textarea{ color:#848783; background-color:#1f261f; }
```

Code 50

At lines 33 and 34 of the css/skin.css file (Code 50) you can edit the label color of the send button (line 33) and the background color of the input fields of the forms (line 34).

```
footer input[type=text], footer input[type=password], footer input[type=email], footer textarea{ background-color:#f0f0f0; }
```

Code 51

At line 190 of the css/skin.css file (Code 51) you can edit the background color of the input fields when the contact form is placed in the footer of the page.

```
.pk_contact_form textarea:focus,.pk_contact_form input[type=text]:focus, .pk_contact_form input[type=email]:focus{
color:#222922; background-color:#ffa800; }
```

Code 52

At line 245 of the css/skin.css file (Code 52) you can edit the background color and the text color of the input fields for the “focus in” status.

Button Send:

the “send mail” button of the contact forms can have two styles: light and dark. Just add the class “pk_light_button” or “pk_dark_button” to it (Code 53).

```
<input type=”submit” name=”submit_comment” value=”Send email” class=”pk_small_button pk_light_button” tabindex=”4” />
```

Code 53

Both the contact forms use the php/sendMail.php file to send the emails. The contact forms also use javascript to show the error messages if a required field of the contact form is not correctly filled.

Let’s see how to set up your contact form:

First of all, open with a simple text editor the php/sendMail.php file.
Into this file you must set your email address, at line 15:

```
$adminEmail = “mail@yourdomain.com”;
```

Code 54

Just type the email address where you want to receive the emails coming from the contact form of the site. Always into the php/sendMail.php file, you can edit the error messages of every field of the contact form. After you make any change to this file, be sure to save it with utf-8 encoding. To do so, if you’re editing it with notepad, just chose the “save as” option and chose the utf-8 encoding from the list.

The success message that appears when an email is successfully sent is defined into the php/sendMail.php file at line 156 (Code 55):

```
<p class=”pk_form_success”>Your message has been sent. Thanks.</p>
```

Code 55

Enjoy it!

p&k