Wide Admin

Powerful backend interface

Thank you for purchasing my theme. If you have any questions that are beyond the scope of this help file, please feel free to email via my user page contact form here. Thanks so much!

Created: 11/01/2010
By: Uniq
Email: uniqcg@gmail.com

Table of contents

HTML Structure

Sidebar widgets:

For titles use <h2>. Same I used on Accordion or Datepicker.
Then add the content after closing the </h2> tag.

Navigation bar:

The menu is coming as it follows under the *<ul class="nav">*.

So, for each item we have a new list item (li).

Example:

```
<ul class="nav">
  <li><a href="">Dashboard</a></li>
  <li><a href="">Simple Link</a></li>
</ul>
```

This will only add buttons to the navbar. No dropdown here.
Don't forget to fill in the *<a href="">* with your hyperlink.

Adding a submenu:

To add a submenu to the main menu, or another submenu item, add a new menu structure, like:

```
<ul>
  <li><a href="">Submenu item 1</a></li>
  <li><a href="">Submenu item 2</a></li>
</ul>
```

before the item of the previous menu item.

Example:

```
<li><a href="">Multi Level Menu</a>
  <ul>
    <li><a href="">Menu Link 1</a></li>
    <li><a href="">Menu Link 2</a></li>
    <li><a href="">Menu Link 3</a>
      <ul>
        <li><a href="">Menu Link 1</a></li>
        <li><a href="">Menu Link 2</a></li>
        <li><a href="">Menu Link 3</a></li>
        <li><a href="">Menu Link 4</a></li>
        <li><a href="">Menu Link 5</a></li>
```

```
        <li><a href="">Menu Link 6</a></li>
      </ul>
    </li>
    <li><a href="">Menu Link 4</a></li>
    <li><a href="">Menu Link 5</a></li>
    <li><a href="">Menu Link 6</a></li>
  </ul>
</li>
```

The above code, will add a submenu to **Multi Level Menu** item on the **Navigation Bar**, also, another submenu to **Menu Link 3** of the first submenu.

Tooltips:

Every item can have a tooltip. Just add "tooltip" to the item's class. (*class="tooltip" / class="classone classtwo tooltip"*). Also, the tooltip text is taken from the title tag of the current item. So after *class="",* or before, add *title="Your tooltip description".*

Notification Messages:

This is the general structure of a notification message:

```
<div class="message">
  <h2>Message title!</h2>
  <p>Message content</p>
</div>
```

You can add **success**, **error**, **warning**, i**nformation** to the class="message". This will style up your messages.

Example:

```
<div class="message success">
```

You can also add **close** next to those two to make the notification messages closeable on click.

Dashboard icons:

The dashboard icons are the big buttons presented on the page. They are list items in one list, as below:

```
<ul class="dash">
  <li>
```

```
    <a href="#" title="Write a new article" class="tooltip">
      <img src="assets/icons/8_48x48.png" alt="" />
      <span>New article</span>
    </a>
  </li>
  <li>
    <a href="#" title="What your team posted" class="tooltip">
      <img src="assets/icons/7_48x48.png" alt="" />
      <span>Last articles</span>
    </a>
  </li>
</ul>
```

To add a new button, simply add a new list item in the **dash** ul, following the structure given above. The **title** value is the tooltip's value. Remove class="tooltip" if you do not want a tooltip for the given button. The img tag is for the icon. I've added some icons under the assets folder for your use. And the span value is the name of the button. The live demo should point you exactly to what you need!

Columns:

First of all, if you want to create a layout with columns, make sure you have included:

```
<div id="columns">

</div>
```

After you've done that, remember your layout can only have up to 4 columns.

Under the columns div, now, for each column you will add one of these:

`<div class="cols2">` - for 2 columns
`<div class="cols3">` - for 3 columns
`<div class= "cols4">` - for 4 columns

Remember that for each column you will have to reopen the div, and of course close it when the column ends.

Example:

For 2 columns we have:

```
<div class="cols2">
        Content of column one
```

```
</div>
<div class=”cols2”>
        Content of column two
</div>
```

For 3 columns, you will have to call the div three times, and for 4 columns, 4 times, of course.

Tabs:

This is the way to create tabs with Glass Admin:

First of all, we’ll go with the tabs declration:

```
<div id="tabs">
  <ul>
    <li><a href="#tabs-1">Forms Preview</a></li>
    <li><a href="#tabs-2">Tables</a></li>
    <li><a href="#tabs-3">Framework Icons &amp;amp; Buttons</a></li>
  </ul>
```

After we have declared the tabs, where the **ul** tag closes we will call each tab with it’s name then insert the content. Please take a look at the example:

```
<div id="tabs">
  <ul>
    <li><a href="#tabs-1">Forms Preview</a></li>
    <li><a href="#tabs-2">Tables</a></li>
    <li><a href="#tabs-3">Framework Icons &amp;amp; Buttons</a></li>
  </ul>
  <div id="tabs-1">
     Tab 1 content
  </div>
  <div id=”tabs-2”>
     Tab 2 content
  </div>
  <div id=”tabs-3”>
     Tab 3 content
  </div>
</div>
```

Sliders:

A slider is mostly controlled by the custom.js file.

It's handled by jQuery UI, so this is how it goes:

The jQuery UI Slider plugin makes selected elements into sliders. There are various options such as multiple handles, and ranges. The handle can be moved with the mouse or the arrow keys.
All callbacks receive two arguments: The original browser event and a prepared ui object, view below for a documentation of this object (if you name your second argument 'ui'):
ui.handle: DOMElement - the current focused handle
ui.value: Integer - the current handle's value

To call a slider to a div, simple give an id to the div, and then under the custom.js file, use the following synthax:

```
$('#slider').slider({
	range: true,
	values: [20, 70]
});
```

Where #slider stands for the div ID you used above. And the values are the values you might want to use.

Modals:

Declare the button for the modal you want to use. This is how i've done it.
What it matters is the id mostly, the other classes are for styling it up. For example this will make a button.

```
<p><a href="#" id="dialog_link" class="ui-state-default ui-corner-all"><span class="ui-icon ui-icon-newwin"></span>Open Dialog</a></p>
```

And here is the content of the dialog you are going to use.

```
	<!-- Dialog -->
	<div id="dialog" title="Welcome message!">
		<p>Welcome to <b>Wide Admin</b>!</p>
		<p>Wide Admin is one powerful customizable backend user interface. Check the demo to see what it can do!</p>
		<p>And this is a custom message text that you can modify to fit your needs.</p>
	</div>
	<!-- End of Dialog -->
```

Remember that these are also controlled by the custom.js file:

```
$('#dialog').dialog({
        autoOpen: false,
        width: 500,
        buttons: {
                "Ok": function() {
                        $(this).dialog("close");
                },
                "Cancel": function() {
                        $(this).dialog("close");
                }
        }
});

// Dialog Link
$('#dialog_link').click(function(){
        $('#dialog').dialog('open');
        return false;
});
```

WYSIWYG editor:

This is extremely easy.
Simple call the wysiwyg class to any textarea, and there you go! A complete wysiwyg editor!

```
<textarea class="wysiwyg"></textarea>
```

This does the trick! ☺

This is all about the HTML section. If I forgot to mention anything, or you need help with something, please feel free to drop me an e-mail on my themeforest profile page, or drop a comment on the item's page and I will assist you with the problem as soon as possible.

Also, I'd be pleased if you would let me know if there are any bugs or if you have any suggestion on how to improve the item! Same as above, use on of the two contact methods!

CSS Structure

There are 6 css files used in this theme, which are placed under the **css** folder.

login.css
layout.css
core.css
skin_clean.css
plugins.css
css3.css

The first file, **login.css** is used for the login page.

The background is handled by the **body** tag.
The rest of the content is placed under the **#container.**

Not that much styling here, as it's only 1 form and logo placed.

The most important part of the theme, is the **layout.css** file, which is made from the other 4 css files, and I mean: **core.css** where the skeleton of the site is created, **skin_clean.css** which is the skin file for this theme, **plugins.css** required by some jquery elements to run, and **css3.css** for rounded corners and styling.

Remember that the file uses css3 selectors to give the aspect and the looks.

Same as above, the **#container** does all the trick.

Background, except the footer is declared in the body tag.

So, we have the following structure:

<table>
<tr><td colspan="2">#header</td></tr>
<tr><td>#sidebar</td><td>#content</td></tr>
<tr><td colspan="2">#footer</td></tr>
</table>

In **#header** we have 4 elements:

.logo, .meta, #navbar (ul.nav and #search)

.logo is where the logo stands.

.meta is the meta information in right side of the screen, where the welcome message is.

ul.nav is the main navigation bar.

#search is the searchbar.

This is for the header part.
The sidebar is as it is, nothing too special, I've talked about it in the HTML structure section.

Under the **#content** we have all the elements.
Refer to any element as declared in the HTML to see it's CSS syntax.

And in the **#footer** we only have two paragraphs, declared **.mid** so they go align to the center of the screen.

Javascript structure

All of the files are placed under the **js** folder of the theme.

The files are:

easyTooltip.js
hoverIntent.js
jquery.wysiwyg.js
jquery-1.3.2.min.js
jquery-ui-1.7.2.custom.min.js
superfish.js
custom.js

These are practically the required files for the theme to run as expected.

easyTooltip.js is the tooltip for html elements. Please refer to the HTML elements.
superfish.js is dropdown plugin for the navigation bar.
jquery-1.3.2.min.js is the jQuery plugin.
jquery-ui-1.7.2.custom.min.js jQuery UI
hoverIntent.js is a required plugin for superfish.js. Just let it live in the folder so everything will work as it should!
jquery.wysiwyg.js is the WYSIWYG editor plugin for jquery, explained in the HTML structure.

And finally **custom.js.** This is a special file created by me where all js / jquery functions are executed to give functionality to the theme. This is what you should edit if you want to edit / add anything.