

CSS Template Tutorial

Please take your time to visit

<http://www.FreeCSS.info>

Table of Contents

Step 1 – Setting up.....	page 3
Step 2 – Coding the basics.....	page 5
Step 3 – Coding and slicing the header.....	page 9
Step 4 – Horizontal CSS Navigation.....	page 15
Step 5 – Column Floating.....	page 20
Step 6 – Coding the content.....	page 23
Step 7 – Right column navigation.....	page 27
Step 8 – Finishing off.....	page 30

Step 1 – Setting Up

Okay welcome to part one of the tutorial. What you will be doing in this part is setting everything up to begin coding your template for your website. This tutorial is for slicing a design made in photoshop and coding it in dreamweaver. You can also access example code at various stages through the tutorial if you have any problems.

Things you will need:

- Dreamweaver or other software
- Photoshop or similar editing software
- Color to HTML - [Free download](#)
- Example PSD
 - [Without slicing already done](#) (get taught)
 - [Pre-sliced](#) (sliced ready for you)

I would also have Internet Explorer 6 installed instead of 7 and Firefox. You can then preview your template in different browsers as you go along. Another tip would be to bookmark [w3c validator](#) to check your coding for errors.

Create a new folder

The first thing to do is to set up a site folder for the design. Go to "My Pictures" and create a "tutorial" folder. This is the folder which all the images and files will be stored in. You can then simply upload the contents of the folder to your website host to get it online.

Set up a new site in dreamweaver

Open up dreamweaver and click on the site tab then new site. This should bring up a box like in the screenshot below. Type in the name of your website and click next (dont bother with URL). Then select "do not use server" and click next. Have edit locally selected and browse to find the folder you set up in for the design. On the next screen select local/ on computer and click the folder to select the folder you set up. That should be your website set up to begin the coding process.

Basically Dreamweaver makes the folder you set up your "root" folder. It makes things a lot easier when saving or linking to things.



Set up the index file

The main files which will hold the code for the template are going to be index.html and style.css. I start of my templates with just index.html and will split the CSS into an external file later. So to end the setting up process. click file, new and HTML. Save the file as index and in your folder which you set up. For example purposes I will refer to it as tutorial folder.

Step 2 – Coding the Basics

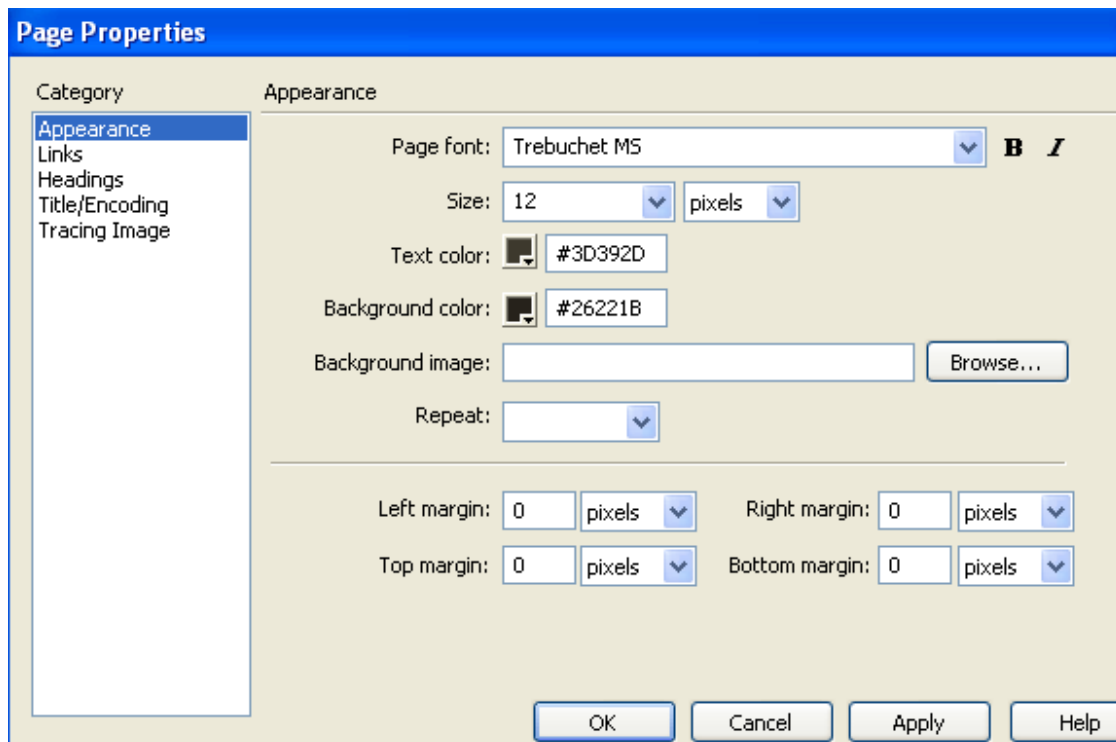
The part of the tutorial will code the very basic things. It will use CSS to define the main holding div and style the text and links for the template. **CSS code must be added between <style> and </style> in head section of webpage.**

Setting up page properties

For the I use the dreamweaver input fields. For the rest of the coding process I usually type myself. To set up the properties open up your index.html file and click page properties button under the properties area at the bottom.



Once you click the button a window should appear with numerous options and input fields. The first thing to fill in is the font for the text on the design and the background. I used the following details for the tutorial template:



The design which the tutorial is for is going to have a fixed 800px width in the center of the page with a dark colour as a background. In the psd the middle content background isn't 800px wide but that doesn't really matter because we will be changing things when we code. For your template put the very background colour in the background colour field. You can get colours using the tool inside photoshop or use [colour to html](#).

Setting up the basic page links

With the properties window still open click the links tab on the left. You can put the colours and setting which you want for the links on your webpage here. This is not the navigation link eg the top menu or right side menu on the tutorial design. This is a simple link such as the colour to html one just above. For the example I put the colour as #BA8F38 and hide underline on rollover.

Setting up the centered content wrapper div

This is a div which will hold the site contents and will be in the center of the page. To do this go to code mode in dreamweaver and change your body coding (pink is the css) to:

```
body {  
  
background-color: #26221B;  
  
text-align:center;  
  
margin:0;  
  
}
```

For the css coding for the wrapper div put the following:

```
.wrapper {  
  
background-color: #F4EDDF;  
  
margin:0 auto;
```

```
width:800px;  
  
text-align:left;  
  
}
```

You can change the background colour and width to suit your design but for the tutorial example this is the parameters which will be used. If I make a design fixed width (a px width instead of %) I always make it 800px. A common screen resolution on old browsers is 800x600. If the width was greater than 800px then people would have to scroll which is very annoying.

You now need to create a div tag with the class wrapper. That means the div will take on the attributes defined for wrapper. To do so scroll down in the code mode and type between the <body> and </body> tags the following:

```
<div class="wrapper"></div>
```

Further explanation for those new to CSS and XHTML can be found below.

Very basic css explanation

.wrapper is the name for the attribute. So will be used for a wrapper attribute. The period in front of the name makes it a class. I will be doing all classes for the tutorial to make things easier.

The curly brackets open and close the definition for the attribute. You can definite what the wrapper class will be like by typing different lines of css attributes within th curly brackets.

Background color and margin and css attributes which you can define different values for. The value for the attribute is typed after a colon. Each attribute and value should be written on separate lines to make it easier to read and must have a semi colon at the end of it.

What makes the wrapper div centered?

The coding which is causing the wrapper div to be centered is a combination of (must have both) the `margin:0 auto` in the wrapper attributes and the `text-align:center` in the body coding.

What does the XHTML mean?

The snippet is the basic code for a div (basically a holder/container) with all the attributes assigned to the class wrapper. A basic div tag is written:

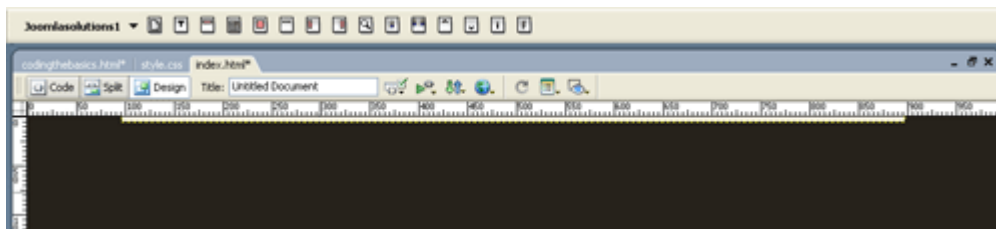
```
<div></div>
```

The `<div>` opens up the container and the `</div>` closes it. Anything inside of the opening and closing tags is going to be held inside the div container. To apply a class (set of CSS attributes) to a div container you will write:

```
<div class="classname"></div>
```

You will notice, you don't include the period before the name when you are typing it in XHTML. So `.wrapper` is just `class="wrapper"`.

You should now have a basic understanding of the way XHTML and CSS works and should have set up your font styling and links for the template. You should also have a div tags which is set width and centered on the page. If you go to design mode you should see something like:



It doesn't look like much but you should have a thin strip of colour 800px wide and centered at the top of the page. Now when you add content inside the wrapper div that colour will expand down the page as the content does.

Step 3 – Slicing + Coding the Header

Welcome to the slicing and coding tutorial for the header. In this part you will slice each image used for the header and logo and code it using CSS and XHTML. The first thing you are going to have to do is to slice the image.

Slicing the image

The first image needed is the background for the header. The background for the header I am doing has a designed pattern which repeats in the horizontal direction. Since it has a gradient whereby the colour changes in the y-axis (light at top, dark at bottom) then it cannot repeat in the vertical axis. Since its a pattern you need to slice the image carefully. You need to make it so when it is copied side by side all patterns line up to make it flow.

To slice an image you use the slice tool in **image ready**. When you have an area selected with the tool go to file save as optimized. Make sure its .gif only and saved in the tutorial folder which you created. It should automatically go into an image folder.



You can see I only selected a portion of the header background. I selected from the middle of one folder to the middle of the next. That way it should go together perfectly (pattern is floral). You will also notice I have left a section of the header (darker colour). This isn't included in the image because solid colour can be

accomplished using CSS. We can define a colour and image for a background attribute. (No matter the positioning in the line of code the image will appear on top then areas where the image background does not cover the solid colour will. It will become more clear later.

Coding the header background

The header is a div container. It is a div which will take on the attributes given to the class header. We can define its height, width and background parameters to begin with.

```
.header{  
width:800px;  
  
height:131px;  
background:#1E1B16 url(images/header_02.gif) repeat-x top;  
}
```

We get the width 800px because it is the full width of the wrapper div. Height of 131px because the image we saved is 121px with a solid colour section below of 10px. The background is defined by the colour first which will be displayed below where the image is. The image is repeated only in the x-axis (repeat-x) and at the top of the div container. If you add the XHTML code to your template you should see the background repeat.

Add this code after the wrapper opening and closing tags. Its integral it goes inside the wrapper div. everything should go inside the wrapper div:

```
<div class="header"></div>
```

Slicing the logo

The logo is the "freecss" text in the header. I will be showing you how to code this using the image replacement technique. This is an SEO (search engine optimization) friendly way of doing things with text in images. This means when you look at the code all you will see is text but in reality there will be an image there.

First thing to do is to slice the logo. To do so select the text using the slice tool and save as optimized just as before:



Try to get as little of the background as possible. That way it will flow better and the image size will be smaller (save on bandwidth and loading time).

Calculate padding values

This is a simple thing which I do to calculate any padding values. If you look at the preview of the design you will see the logo section we sliced has a gap between the top of it and the top of the header div. If there was no padding it would simply be placed at the very top left of the header div container. We therefore will need to add padding left and padding top to the header div container so the logo is placed in the right position.

To calculate the top padding value I slice the image of the header which is above the logo like so:



Save as optimized and read the height value this is what we will be using. For padding to the left side of the header we will simply guess. From the image we can see it has a height of 45px. We therefore have to alter our header container's css to this:

```
.header{
width:600px;
padding-left:200px;
padding-top:45px;
height:86px;
background:#1E1B16 url(images/header_02.gif) repeat-x top;
}
```

Why have the width and height values changed?

This is an important lesson to learn. When you add padding to a container its like adding a buffer space, an area which cannot have anything put into it, a sort of text indent. When you add padding you are adding to the width and height of the container. If you have padding-top:45px that means you will have a section 45px at the top of the container which is indented (content put into the container will not appear there, instead indented down). So when you add a padding value you must subtract the value from the appropriate value (height for padding-top or bottom and width for padding left or right).

Coding the logo using image replacement technique

The logo will be an h1 tag. This is the main heading of a website. If it was a newspaper it would be the title on the front page. This gives it more importance to search engine spiders. We will use the CSS image replacement technique so the image of the logo is clickable and if you read the code appears to be regular text. If you want to learn more about heading tags and how to use them check out: [heading tags explained and how to style heading tags](#).

The coding for the heading (h1) tag is quite complicated so let me type it first then go through it. The following example is only going to work if your title is linked (best to link to your homepage):

```
h1{
width:201px;
height:83px;
margin:0;
overflow: hidden;
background:url(images/logo_05.gif) no-repeat;
```

```
}  
h1 a:link, h1 a:hover, h1 a:visited, h1 a:active{  
display: block;  
width:201px;  
height:83px;  
text-indent: -100000px;  
}
```

The XHTML code you will use must be placed between the opening and closing tags of the **header div**. You can obviously change details in the code for your website.

```
<h1><a href="http://www.freecss.info">Free CSS </a></h1>
```

CSS Explained

The width and height are self explanatory. They are the values of the logo image. **Important:** all heading tags eg h1, h2 etc have margins as standard. You must put margin:0 onto these. The overflow hidden makes sure anything outside the width and height specified will not be visible (this will be the text). The background I also assume is self explanatory. The no-repeat means it will only display once.

Note: heading tags and other elements in XHTML don't require the period in front of them when you are writing there CSS.

For links you write a:link, a:hover for different linking states. In this case it is the same for all. I could write each one out separately with the same content but its easier to list them and separate with commas.

The display block makes the area (width and height specified) clickable. The text-indent is the sneaky part to hide the text we write in the XHTML. This will be shifted out side the area and since we wrote overflow hidden it is not visible.

Outcome of this part

You should have something which looks like the following in design mode:



If you press F12 to preview the design you should find the logo clickable and links to your website. If you need further explanations or help please post a comment here. I had to write this part twice since I deleted the first version by accident. I might therefore have missed a couple bits out.

Step 4 – CSS Horizontal Navigation

Welcome to the horizontal navigation tutorial. So far you should understand CSS and the way it works and you should have a header with a logo. In this part you will be coding and slicing the horizontal navigation as you can see in the preview. of the design. The tutorial is to create a "sliding door" technique navigation since it has rounded edges.

If there was not rounded corners we could simple repeat an image in the horizontal direction. Since it has rounded corners we cannot do this.

Slicing the navigation background

First we need the light background to repeat the full way across the wrapper and to hold each navigation item. To do this you first need to slice the background image like in the diagram. This will be repeated in the x-axis:



Usually I would make the selection 1px wide but since there is a pattern it must flow just like the header background had to flow together.

Coding the navigation background

This is going to be very simple. Just a div container with the full width of wrapper and of the height of the image which will be repeated on the x-axis.

```
.nav{  
width:800px;  
height:37px;  
background:url(images/navbg_12.gif) repeat-x top;  
}
```

That should be the CSS of the nav container which will hold each navigation item. We may need to adjust it with some padding but we can do that later. You then

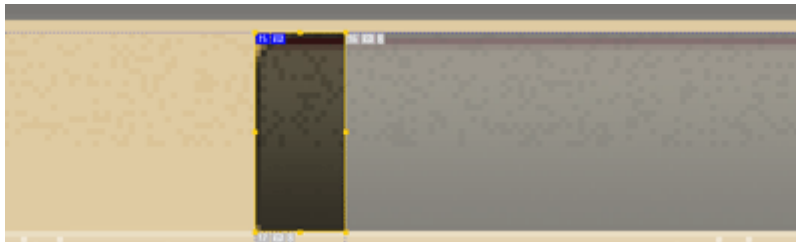
need to add the XHTML for the nav div container after the closing tag for the header:

```
<body>
<div class="wrapper">
<div class="header">
<h1><a href="#">Free CSS </a></h1>
</div>
<div class="nav"></div>
</div>
</body>
```

Slicing the images for the navigation tabs

This part is the simple part. Things you need to keep in mind. The left image is as thin as possible and the right is to be bigger than you should ever need as it will vary how much of that image is displayed. It doesn't repeat, just more of it is revealed.

The first image is the left side. To slice that image you will use the slice tool as always in image ready:



Make sure you have no text on the tab and that the right side is going to be wider than you should ever require (wider than any link will be):



Coding a navigation as a list

When coding a navigation using CSS we use the and tags. This stands for unordered list and list item. Each link is a list item which is part of an unordered

list. This means the unordered list will be held inside the nav div container we just created.

The coding for the navigation is quite complicated. I will paste the code first then go through each part so you can understand what we are doing:

```
.nav ul {
margin:0;
font-size:11px;
font-weight:bold;
padding-left:50px;
padding-top:7px;
list-style:none;
}
.nav li {
float:left;
background:url(images/navleft_15.gif) no-repeat left top;
margin-right:10px;
padding:0 0 0 15px;
border-bottom:1px solid #ccc;
}
.nav a {
float:left;
display:block;
background:url(images/navright_16.gif) no-repeat right top;
padding-top:5px;
padding-right:20px;
padding-bottom:7px;
padding-left:6px;
text-decoration:none;
color:#FFFFFF;
}
/* Commented Backslash Hack
hides rule from IE5-Mac */
.nav a {float:none;}
/* End IE5-Mac hack */
.nav a:hover {
color:#FFF;
```

```
text-decoration:underline;
}
.clear{
clear:both;
}
```

The XHTML is very simple in comparison:

```
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">News</a></li>
<li><a href="#">Products</a></li>
<li><a href="#">Categories</a></li>
<li><a href="#">Contact</a></li>
</ul>
```

The whole extract should go between the opening and closing tags for the nav container. Each li is a link.

CSS explained

The coding for the CSS looks huge and overwhelming? Don't worry, in reality its not too complicated. You don't even have to understand it to use it. This is one of the more complex types off navigations because each item consists of two images and must be able to expand.

list-style:none - This removes the bullets for each item. This is a standard with unordered lists.

Something else which you should note when using the sliding door technique. The padding on the left side (in this case 15px) must equal the width of the left image. You can see this in the second set of coding. Instead of writing it out I wrote it in one padding. The order goes top, right, bottom and then left.

In the third set of coding the right padding 20px needs to be equal to the left padding on the previous set plus the padding on the left defined in that one. In this example that means 15px plus 5px. That ensures the text is displayed in the exact center of the tab background.

The use of css hacks

You will also notice in the CSS coding a hack. This is something which is used to fix any cross browser differences. Firefox may display the code correctly but IE6 may show it incorrectly. This is a problem with some code and the hack is needed here. I don't often use hacks and tend to just copy it from other code. This is actually a really good way to learn.

Whats up with the .clear css?

The class called clear is used to clear the floats which we use in the navigation. Floats are pretty self explanatory. It floats an object either to the left or right. This allows multiple objects on the one line whereas without the float it would just appear on the left and one underneath the other. A problem with floats however is that it can break the look of your design. It can cause other div containers not to expand to hold them. So in this case the wrapper div would not expand.

To fix div containers which don't expand due to floats held inside them you simply add the clear:both attribute to the css class for clear. You can then insert a short line underneath your floats to clears them and make the wrapper or containing div expand. For this example place the following XHTML under the closing tag for the **nav container** not ul .

```
<div class="clear"></div>
```

You should now have a functioning and great looking css navigation. No? If you have some problems post a comment here and I will try to help you.



You should have something like the above. If you want here is the source code which I have thus far: [Example code](#). Also just to say to people who are familiar with CSS. I am currently keeping the css in the head section of the page instead of in an external stylesheet purely to make things easier while coding. I will switch it to an external sheet at the end and show how to link to it properly.

Step 5 – Column Floating

This is going to be a short part of the tutorial. In this part I will basically set up the left and right columns so they are ready to add the content to. This is the basic structuring of our template. First thing you will need to note is the gradient underneath the navigation on the design. This will have to be added as a background for a content div.

Slice content background

Like always go into image ready and slice the top gradient underneath the navigation as such:



Most designs have gradients underneath the navigation and above the content or as a background for the content. In those cases make the slice 1px wide. In this case I also have a pattern. Therefore to make it flow it has to be a reasonable width as describe in previous parts of the tutorial.

CSS coding for columns and content

The idea is we we have a div called content which will hold both columns. It should be the full width of the wrapper div. Since the columns will be floated inside the content div we will need a clearing div after the columns but before the ending tag of the content div (clearing divs are explained in previous part). To work out the widths for the columns use the technique of slicing images on the design. This isn't going to work for the example since the width of the design in the psd is not 800px.

We should therefore have the following CSS coding:

```
.content{  
width:800px;
```

```
background:url(images/undernav_22.gif) repeat-x top;
}
.leftcolumn{
float:left;
width:500px;
padding:20px;
}
.rightcolumn{
width:220px;
padding-top:20px;
padding-bottom:20px;
padding-right:20px;
float:right;
}
```

To me that coding seems pretty logical, yeah? Basically the content holds the two columns. One is wide and floated on the left side with padding so the text inside it won't go right to the edges and the other is think and floated to the right (will hold further navigations and links). There is also a gap between the two columns so the are not too close and cluttered looking. To do that simply make sure the widths of the two columns (including paddings) is less that 800px. It just takes some experimenting to get the right looking dimensions. I tend to guess then change them instead of calculating it properly.

The XHTML code used is:

```
<div class="content">
<div class="leftcolumn"></div>
<div class="rightcolumn"></div>
<div class="clear"></div>
</div>
```

You should already know how XHTML works by now but if you didn't I have highlighted which closing tags go with which opening tags. It may make it easier to understand. Put this coding section after the closing tag for the nav. If the XHTML worked any differently ie the first red closing div closed the content div you would end up with overlapping element and it just wouldn't work.

If you go to the design mode on dreamweaver you should now have something like the following:



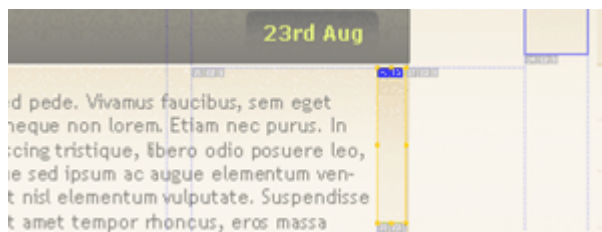
We have the basic shell which we will add the content to. We may find that we will change the width of the columns later and add some extra CSS but this will do for now.

Step 6 – Coding the content

In this part we will code the article background and the comments button. For the design we are using for the tutorial the content on the left is a post in wordpress. The content of each post is going to be held in a div with a background at the top. There is also a div tag inside that one with the background at the bottom. The comment button will then be added into the end of the inner article div into its own div.

Coding the article CSS

This part is going to be simple so it probably won't be very long. Simply slice the background pattern at the top of the article like so:



The coding is actually quite complicated because the inner div used. Basically you have the article div. This is the holder for artfoot and has the background at the top. The line-height attribute just spaces the lines a bit more. The art foot is therefore inside the article div and has the borders and paddings applied to it. The margin bottom is also added so when you have numerous posts there is a gap between them.

```
.article{
width:500px;
background:#F5EEEE0 url(images/topbg_26.gif) top repeat-x;
line-height:20px;
}
.artfoot{
width:479px;
margin-bottom:30px;

text-align:left;
```

```
padding-top:5px;
padding-left:10px;
padding-right:10px;
border-left:1px solid #CBA964;
border-right:1px solid #CBA964;
background:url(images/artfoot_30.gif) repeat-x bottom;
border-bottom:3px solid #CBA964;
}
```

You should place the XHTML for the article and artfoot divs after the closing tag for holder. When you add text inside the artfoot div you should have `<p>` before each paragraph and `</p>` at the end.

```
<div class="article">
<div class="artfoot">
<p>This is an example paragraph . </p>
<div class="info bar"> Comments (5)</div> </div>
</div>
```

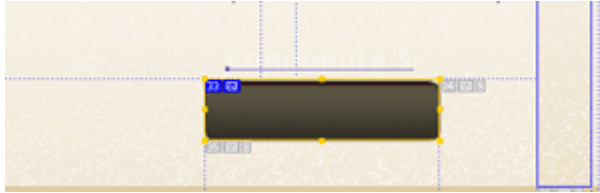
Coding the comments button

As you can see from the above code I have put the comments button into an info bar div. So lets first define the CSS for the info bar then change the coding for the actual comments button. The coding for the info bar should be simple. Some padding at the bottom, full width, padding to the right and text aligned to the right. The coding of the info bar is just to get the comments button in the right place.

```
.info bar{
width:438px;
padding-right:40px;
text-align:right;
padding-bottom:10px;
}
```

Note: Since the artfoot has a 1px border on either side and 10px padding on either side the width available for the info bar is a maximum of $500 - (2 \times 20) = 478$ px.

That should be all the coding we need for the info bar. The actual comments link is going to be more difficult. I am going to make it easier by making the button a fixed width meaning we just need to slice the button background and no need for repeats.



CSS for comments link

I have referred to the comments link as a button as it appears like a button. Please don't confuse this with a form button. It is simply a link with CSS used to define the background image and area for the link. The coding for this is actually pretty complicated. I decided to make it a list so other links could be added as well such as read more. If you want me to explain it please post a comment. Otherwise you can just use my code and try to adapt it if you need to:

```
.info bar ul{
list-style:none;
text-align:right;
margin:0;
}
.info bar ul li{
width:109px;
height:24px;
padding-top:2px;
background:no-repeat top center url(images/button_33.gif);
text-align:center;
color:#F4EDDF;
float:right;
font-weight:bold;
font-size:11px;
font-family:Verdana;
}
.info bar ul li a:link, .info bar ul li a:visited, .info bar ul li a:hover{
display:block;
```

```
text-decoration:none;  
color:#F4EDDF;  
}
```

The XHTML has therefore also changed and a clear has been added since the li items are floated to the right. You should end up with XHTML like:

```
<div class="article">  
<div class="artfoot">  
<p>Example paragraph in a post </p>  
<div class="info" ><ul><li><a href="#">Comments (5)</a></li></ul><div  
class="clear"></div></div>  
</div>  
</div>
```

You should now have completed the left column. All that is left to do is the links on the right and a really simple footer.

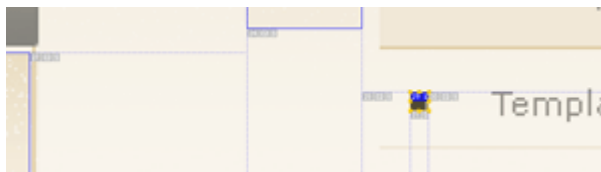
Step 7 – Right Column Links

The tutorial is almost over and the template is almost finished. In this part I will show you how to create the simple navigation which is on the right column in the design preview. It is going to be a navigation created using unordered lists just like other ones.

If we look at the design we can see each list item is going to need a border at the bottom and a custom bullet on the left.

Slicing the bullet image

As usual I slice the images needed first. We are just going to need the one image for this navigation. Simply slice the square bullet in image ready and save as usual. Remember to zoom in to make it easier.



Coding the vertical navigation in CSS

Right now we have the images needed. All we have to do now is to code it. The code isn't very difficult so I will just present it to you. You should be able to see each thing is created using the CSS. If you want something explained just ask.

```
.rightcolumn ul
{
list-style-type: none;
padding-left:0;
margin:0;
text-align: left;
width:100%;
font-family:Verdana, Arial, Helvetica, sans-serif;
padding-top:5px;
}
.rightcolumn ul li{
```

```
padding-top:6px;
padding-bottom:6px;
border-bottom:1px solid #E8D9BD;
margin:0;
}
.rightcolumn ul li a
{
background:url(images/nullet_29.gif) left no-repeat;
text-align: left;
padding-left: 20px;
font-size:11px;
text-decoration: none;
color: #4E4739;
}

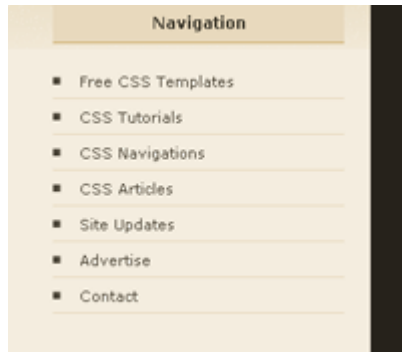
.rightcolumn ul li a:hover
{
background:url(images/nullet_29.gif) left no-repeat;
color: #4E4739;
padding-left: 20px;
text-decoration:underline;
}
```

A couple things to note from the above example. List navigations such as this can be given paddings by default. That is the reason for the padding-left:0 for the ul. The list-type:none removes the regular bullet and the background attribute is used for our bullet image.

The XHTML for the navigation is really simple. The navigations like this can only be shown on the right column but numerous can be added. The XHTML should be:

```
<ul>
<li><a href="http://www.yoururl.com">Home</a></li>
</ul>
```

After you have added this you should see something like below when you preview:



Something wrong?

As in other parts if you have done something wrong or need further explanation please post a comment. You can also download the example source code thus for from here: [example code](#).

In the next part

We will finish off the template by adding a footer and cleaning it up a little. We will also move the CSS into a separate file.

Step 8 – Finishing Off

Welcome to the final part of the tutorial. So far you should have picked up all the techniques and expertise to code a template. I will show you in this part how to finish the template off with a footer and just make our coding a little neater. I will also give a few tips for optimizing you website.

Coding the footer using CSS

I reckon you should probably be able to do this on your own by now but if not don't worry. I will go through making the footer for the template. Basically its going to be within the wrapper div (holding the website content) and below the ending tags for the content. The attributes needed for our example will be width 100% and with a solid background colour.

```
.footer{
width:100%;
padding-top:15px;
padding-bottom:15px;
color:#fff;
text-align:center;
background:#4E4739;
}
```

All pretty self explanatory I would say. No need for anything complicated for this one. For the XHTML simple place the following just before the final div end tag (the one for wrapper div):

```
<div class="footer">Template by <a
href="http://www.freecss.info">freecss.info</a></div>
```

So thats its for the CSS and XHTML. You are completed. Now time to just clean a couple things up.

CSS in external stylesheet

I believe you can do this in more than one way but I will just do it the way I am used to. Its quite simple. Press file then new and select CSS. Go to code mode of

your current template and copy everything after `<style type="text/css">` and before `</style>`. Past this into the code mode for the CSS file. If want you can only copy it into notepad. Next save the file as style.css and in your tutorial folder.

You have now created your style sheet. Next thing to do is to remove from `<style type="text/css">` to `</style>` on your index.html. Also remove the tags themselves. The template now has no CSS defined. If you switch to design mode you will see what your website would look like with CSS turned off- Just as well CSS exists 😊

Now you will need to link in your index.html to you style sheet. This is simple. Just add the following code you to your index.html's head section (after title tag probably and before the `</head>` tag).

```
<link rel="stylesheet" type="text/css" href="style.css"/>
```

If you check in design mode you should now see your template with CSS defined.

Make sure everything validates

I always do this with my websites but sometimes it doesn't really make much of a difference. Its best to have proper compliant code which validates. To check your CSS and HTML use [w3c validator](#). You can choose various options to validate your code there. It will tell you any problems which you should be able to then go and fix.

After checking my coding I found no errors. That just comes after many years of coding. At first I would make something like 20 different errors on my first draft of coding so to speak. If you have problems with the way your website is displaying its often worth checking to see what the errors if any for your coding are. It can often show up spelling mistakes and typos.

Some search engine optimization tips

A few tips for coding templates. First off try to keep any text in your templates as html text. Try not to contain text in images. If you do this its best to use the image replacement technique to make it look as though the image is text. Its also best to use alt tags for you images that you do have. This helps with accessibility. Have

your website title or keyword in your title tag. This is probably the biggest on page seo factor so use it!

You have reached the end

Well if I was to say it was great fun 😊 I would be lying (deleted parts and had to rewrite them. Lots of stumbling blocks and time was spent on this tutorial so please if you find any problems with it post them here and I will try to sort them out. I hope you found the tutorial interesting and informative. You can hopefully learn something from what I have written.

[Download Finished Template](#)

[Preview Finished Template](#)