Placing Elements in HTML - Positioning

Normal Flow of the Document

Normal flow is the way a web document will display in the browser or viewport. The content will flow down the page, starting with the first element in the body of the html and finishing with the last element in your document.
Block elements are formatted visually as blocks and will start a new line at the left and stack up vertically, i.e. paragraphs.
Inline elements will remain in a line horizontally going left to right until there is no more room to the right and then drop to a new line, i.e. images and links.

Box Model

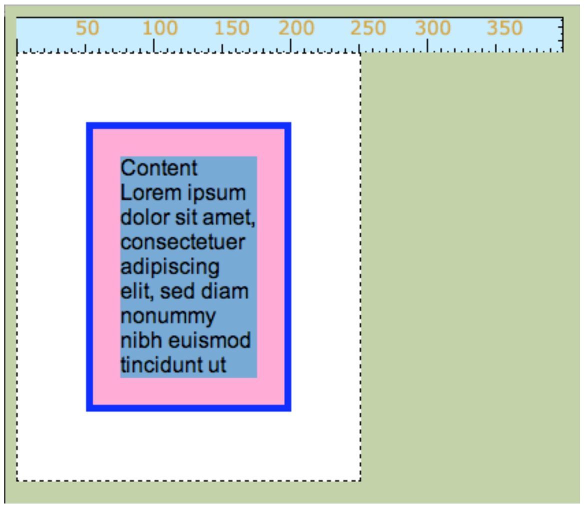
Open `box.html` in Firefox browser. Note the dimensions used to create this box. css

```
      .inside { margin: 50px;
                padding: 20px;
                width: 100px;
                border: 5px solid #0000FF;}
```

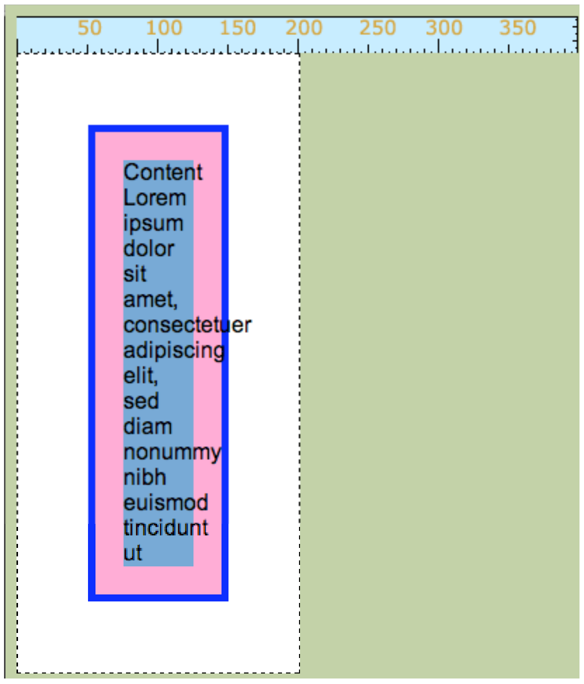
Proper Box Model
The width, the blue content area, is 100px.
The padding, borders and margins are added on to the outside of the width,
to arrive at a total of 150px for the visible bordered box dimension,
or 250px from outer (dotted) margin edge to outer margin edge.

50 + 5 + 20 + 100 + 20 + 5 + 50 = 250 pixels actually filled on screen

According to CSS specifications:


50
100
150
200
250
300
350
Content Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut

As IE 5 interprets:


50
100
150
200
250
300
350
Content Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut

IE5 Broken Box Model
In IE5, "width" was the area including the blue border and padding.

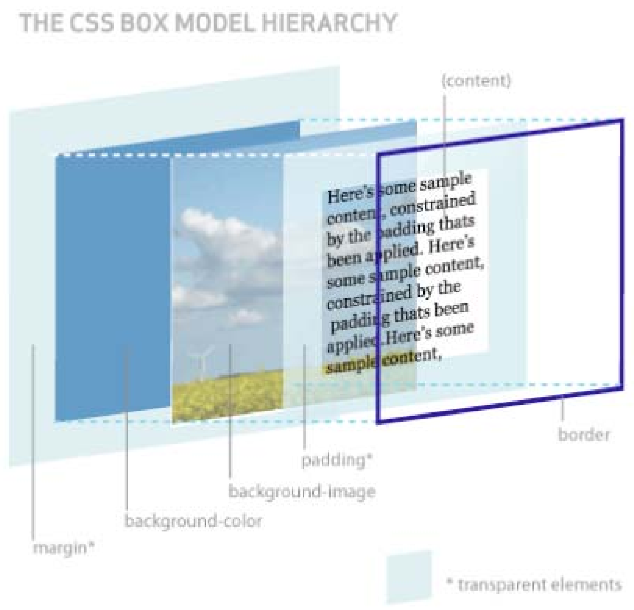
50 + 100 + 50 = 200 pixels actually filled on screen

- Missing `DOCTYPE` (Quirks mode) will invoke broken box model in IE6.

- To avoid problems with old IE 5 browsers, when possible avoid setting padding and border on an element with a stated width.
    - Add the margin to the contained element or if necessary add a wrapper div.
- Use **Web Developer toolbar** to see IE5 box model effect on your layout if you need to support IE5 browser (**CSS>Use Border Box Model**)

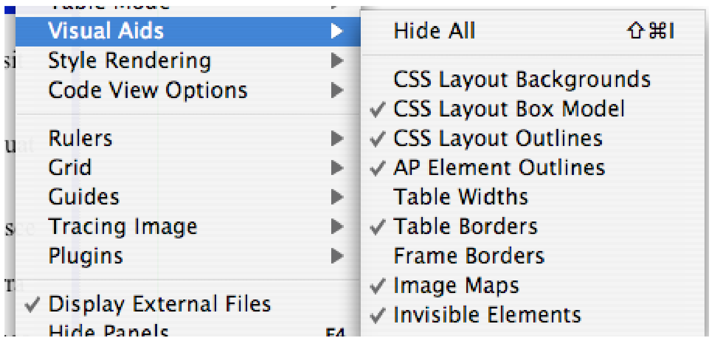
So if you are setting a width on an element to fit into a limited space, say your sidebar column is 300 pixels, you must take into account the margin, border and padding involved and subtract these dimensions from the width you set for sidebar in the css. For example, I want my sidebar to be 250 pixels wide, and I want a 1 px border and 25px margin on the right, plus padding on the left and right of 20 pixels. 20 + ? + 20 + 1 + 25 = 250 (250 – 66 = 184)

The box model also has "layers". The following diagram shows the layers and position of each element in the box. (image from http://www.hicksdesign.co.uk/boxmodel/)


THE CSS BOX MODEL HIERARCHY
(content)
Here's some sample content, constrained by the padding thats been applied. Here's some sample content, constrained by the padding thats been applied. Here's some sample content,
border
padding*
background-image
background-color
margin*
* transparent elements

Visualizing Margins in Dreamweaver

You can use the **Dreamweaver Block Visualization** and **Ruler Guides** to help you get your element boxes right. Dreamweaver shows the Visualization on block elements such as div tags, and any other page element that includes the `display:block, position:absolute, or position:relative.` It does not however display like this for simple paragraphs and inline elements, such as images. This feature is turned on from the **Toolbar Menu>View>Visual Aids**



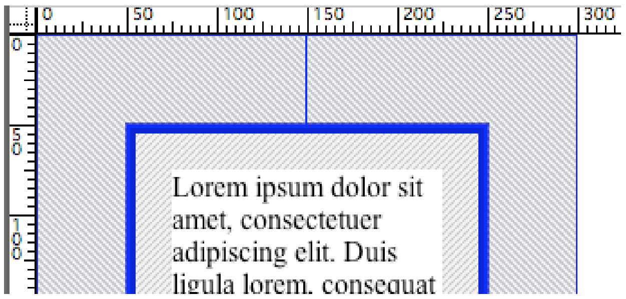
Open a new document in Dreamweaver and Save As `margin.html` in the `positioning folder`. With the Rulers on (View>Rulers>✔Show), drag out a vertical guide to 300 pixels. In the CSS panel, create a new rule for Body and set all margins to 0. Click OK. Using Loreum Ipsum, insert a paragraph of text.

In the CSS panel, create a **New CSS Rule** for the <p> tag. In the **CSS Rule Definition** dialog box, choose the category **Block**, and set **display: block**. This will get Dreamweaver to

"Visualize" the padding and margins. We are not really adding anything to the paragraph style here, because a paragraph is already a block element. You would not usually add this property, we are just doing it here to help us see how the box model works.

Now set your box properties for the <p> rule to see the results of adding margin, padding and border to the size of the box. In the **Box category** in the dialog, set the **width to 300px**, set **all margins to 50px** and set **all padding to 20px**. Switch to the **Border category** of the CSS Rule Definition dialog and set **all borders to solid, 5px and pick a blue color**. Click **OK**.

Back in your design view window, place your **cursor over the edge of the border until you see the light dotted outline turn pink and then click**. You will see the Visualization of the margins and borders. You can see that with these settings, the space that our paragraph occupies with its borders and margin and padding is far greater than the 300 pixels that we want it to take up.
We must subtract the extra margin, padding and border to get the remaining width and this is what we must use for our element's width in our css rule. So **300** (total width) **–** **100**(right and left margins) **–** **10**(right and left borders**)** **–** **40**(right and left padding) **=** **150** width to set in css rule.



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis ligula lorem, consequat

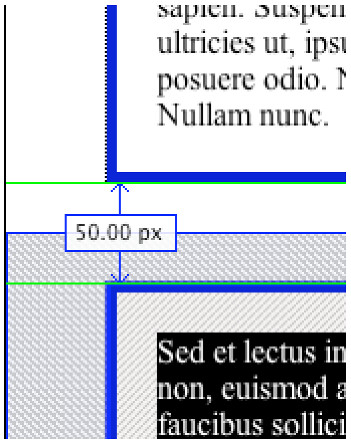
In your **CSS panel All View**, click on the p rule, and then in **properties for "p" sub-panel**, click on the **width** property and change the width to **150px**. Check your paragraph using the Visualization again and note that it now occupies the proper amount of space.

Margin collapsing

Margin collapsing happens when ***two or more vertical margins meet, they collapse to form a single margin***, equal to the height of the largest margin. When two elements with top and bottom margin are over each other, the total margin is reduced to equal only one, the largest.

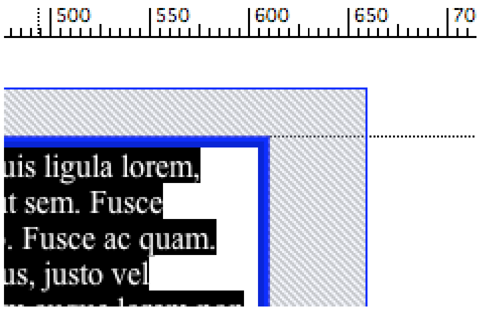
In the **CSS Panel, select the p tag rule,** then in **sub-panel properties for "p"**, change the **width** property to **500px** so we can see this more easily with our Dreamweaver visualization feature. Back in the document window, press the Enter or **Return key to add a new paragraph** and **insert some more text**. If you click on the dotted block outline to revel the margins, you can see that each paragraph is "sharing" the 50 pixels space in between the blocks.

In the **CSS panel**, **select the p rule** and using **edit** (pencil icon at the very bottom of the CSS panel), open the CSS Rule definition dialog box, choose the **Box category**. For **Margins uncheck Same for All, set the Top to 25px and leave the bottom at 50px**. In the design window, click on each paragraph block outline to see the effect of the collapsed margin. They still share the space, and it is the larger of the two margins, with the top margin of 25 pixels collapsed into the 50 pixels of the top paragraph.



You can **pull the horizontal guides down** from the ruler to the edges of each paragraph and **while holding down the CMD key**, **hover over the space in between** to see the measurement of 50 pixels.

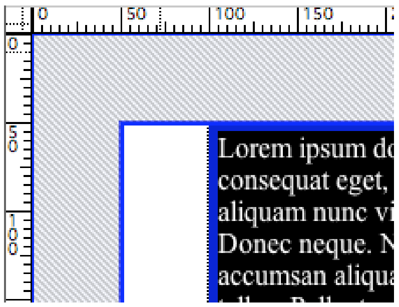
When an element with a top or bottom margin is contained within another element, without padding, their top and bottom margins will also collapse together.



In the **CSS panel, select `p` and in the properties for "p" sub-panel**, **delete the property for padding**. In the design window, select the two paragraphs to wrap them in a `<div>`. While selected choose from the **Toolbar Menu>Insert>Layout Ojects>Div Tag.** In the **CSS panel, create a New CSS rule** for the element `<div>`. In the **CSS Rule Definition** dialog choose **Box category** and **set all margins to 50px**.

Now select the dotted block outline of each paragraph to see where the top margin of the first paragraph has collapsed into the top margin of the containing `div` and the bottom margin of the second paragraph has collapsed into the bottom margin of the `div`.

Some designers choose to always set their paragraph top margins to zero and control only the bottom margins, and then setting headings to have a larger top margin than the paragraphs bottom margin so headings have more space to separate them from the previous content.



As you have probably noticed with this last exercise, the right and left margins of the division and the paragraphs did not collapse. Margin collapse only applies to vertical margins (top and bottom). Once you understand the margin collapse, you can use margins properly to control your layout.

Positioning

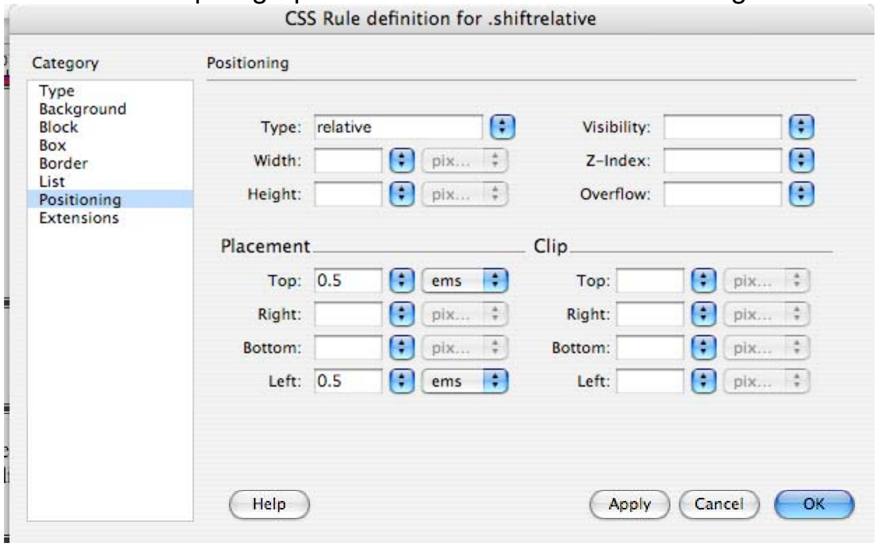
CSS provides the rule position to allow you to position elements out of their normal document flow. The values available are relative, absolute, fixed and static. Fixed is not supported in IE6 and static is not widely supported at all.

Relative Postioning

Relatively positioned elements are positioned within the normal flow and then moved. Elements that come after a relatively-positioned element behave as if the relatively-positioned element was still in its ‘normal flow’ position - leaving a gap for it.

Open `relative.html` in your positioning tutorial folder.
There are several css rules already added to this page, mostly to add borders, zero margins and make containers.

Look at the `.shiftrelative` rule. Select and open using the edit pencil icon. Choose the Positioning category. Type relative is choosen and under placement we are telling the **Top property** to move down from the top **.5em** and the **Left property** to move to the left **.5em** from the normal document flow. Some people use a period to mark the normal location of the content, as we did with the [photo here] text in our float exercise to mark where the image was inserted in our paragraph. Click **Cancel** to close the dialog box.



CSS Rule definition for .shiftrelative

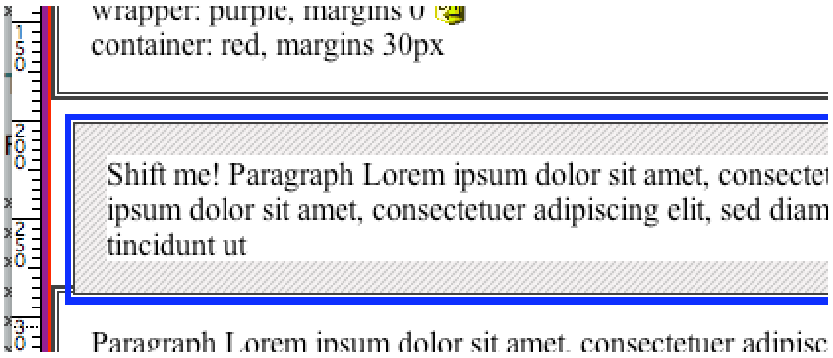
Category: Type, Background, Block, Box, Border, List, Positioning, Extensions

Positioning

Type: relative

Placement — Top: 0.5 ems; Left: 0.5 ems

Back in **design view** in the document window, place your **cursor in the third box** with the text Shift me! In the **Tag Selector bar**, select the <p> tag and **right click** to **apply the style** `.shiftrelative`. Observe the paragraph is now down and to the left of its original placement and none of the other elements have shifted before or after this paragraph.



wrapper: purple, margins 0

container: red, margins 30px

Shift me! Paragraph Lorem ipsum dolor sit amet, consecte ipsum dolor sit amet, consectetuer adipiscing elit, sed diam tincidunt ut

Paragraph Lorem ipsum dolor sit amet, consectetuer adipisc

Go to the **CSS Panel**, select `.shiftrelative` and **change the positioning properties** in the **properties for ".shiftrelative" sub-panel**. Refresh the design view or preview in the browser to see the effects. Try changing the **Left** property to a negative: **-3em** and notice that the paragraph block is now outside the viewport and the containing wrapper and container divs. Now change the **Top** property to a negative: **-3em**, and notice that the paragraph is move up from its normal location. Both top and left properties are working from the top left corner of the element to be positioned. Bottom and Right properties work with the bottom right corner of the element. Now add a property to `.shiftrelative` for **bottom: 1em** and **right: 1em** and **delete the current left and top properties**. Notice how the block has shifted up and over from the bottom left location that it normally occupied. Change the measurement for right to a negative: **right:-3em** and notice how the paragraph block moved to the right. So just remember that these properties describe moving the element **from** the location named by the property, and are not about the direction you are moving the element.

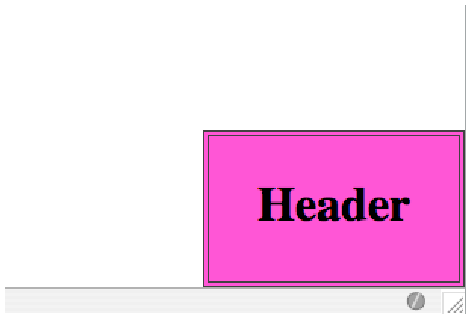
Absolute Positioning

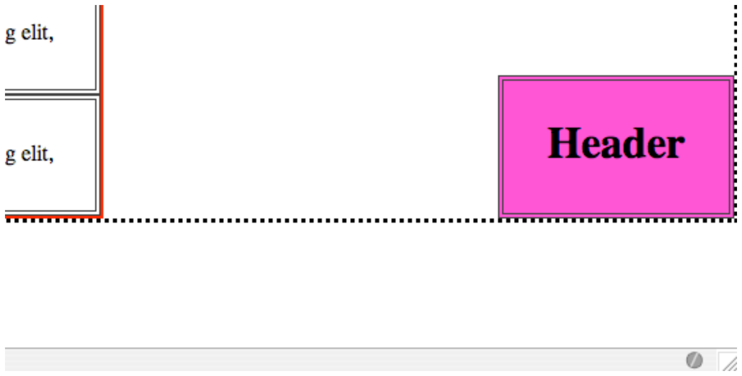
An absolute positioned box is moved out of the normal flow entirely and thus takes up no space where it is in the normal document flow, leaves no gap behind like the relative positioning did.

Again placing a text marker or a period can help a designer visualize the effect of absolute positioning.

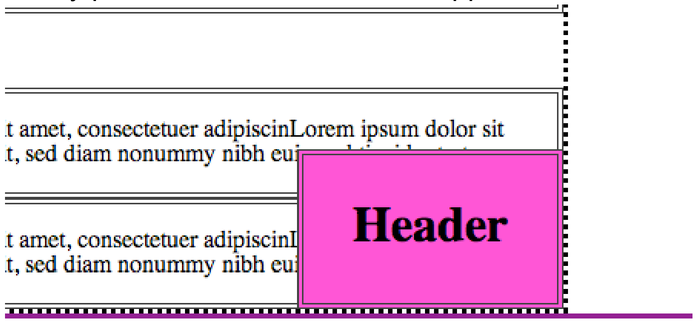
The most important aspect of the absolute positioned element is that its position is based on the document tree and its containing blocks. It is positioned in relation to its nearest positioned ancestor. If there is no positioned ancestor, then it is positioned in the viewport.

Return to `relative.html` and look at the rule `.shiftabsolute` in the CSS Panel. Click on the edit icon to open the edit dialog box and choose Positioning category. Notice that the **Type** is now **absolute**. Look at **Placement** and notice that this rule uses **Bottom and Right** to absolutely position the block at the bottom right of the last positioned ancestor, which for the header element is the viewport. Go to your document window, place your cursor in the header and from the **Tag Selector bar**, **right click** on the `<h1>` tag and apply the css style `.shiftabsolute`. Preview in the Firefox browser and notice that the header block is now at the very bottom right of the viewport and outside all of the <div> containers. If you change the size of the viewport, the header block moves with the bottom right corner. In the **CSS Panel**, with the `.shiftabsolute` rule selected change the value for the property **bottom to 3em** in the properties panel. Preview again in the browser. Notice that it still homes to the bottom right corner, but is now 3ems above the bottom edge. Again if you change the size of the browser window the header tracks with it remaining 3ems above the bottom edge. In the **CSS Panel properties for** `.shiftabsolute`, change the Placement values back to **bottom:0** and **right:0**.





Now look at the rule `.relativeancestor` in the CSS Panel, select it and see in the properties panel, that this rule has a property of relative, but no placement properties and a dotted border to help us see the effect. We are now going to give the `<div id="wrapper">` this style. Click anywhere in the paragraph box areas, and in the Tag Selector bar, choose `<div#wrapper>` and right click to apply the `.relativeancestor` style. Save and preview in Firefox borowser. The header block is now absolutely positioned within the last relatively positioned ancestor, the wrapper div.


t amet, consectetuer adipiscinLorem ipsum dolor sit
t, sed diam nonummy nibh eu
t amet, consectetuer adipiscinL
t, sed diam nonummy nibh eu
Header

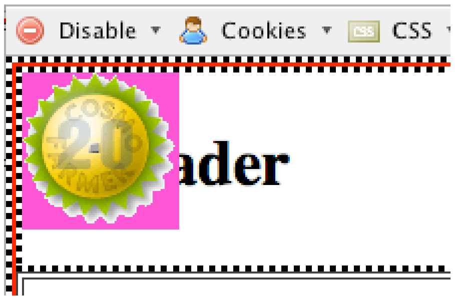
Now add `.shiftabsolute` to the `<div#container>` in the same manner and preview in browser. Now the header block is positioned at the bottom right of the container div, because it is now the nearest ancestor positioned. Notice that Dreamweaver does not do a great job in the design view of displaying the absolutely positioned header. Always preview in Firefox to be sure of the effect. Place your cursor on the header block and using the Tag Selector remove the class `.shiftabsolute` from the `<h1>` tag, by right clicking and selecting **none** from the styles

dropdown list. You should see the header block return to its normal position in the document flow. Place your cursor at the end of the header text and insert the image `cosmo_badge.gif`. Now in the Tag Selector bar, choose the `<h1>` tag and add the style `.relativeancestor`. Then select the `<img>` tag and add the style `.shiftabsolute`. In your **CSS Panel**, change the Placement Properties' values for `.shiftabsolute` to place the image in the top left corner of the header.
(Add top:0 and left:0, and delete bottom and right properties.)

The image is now covering the text. This is the problem with absolute and relative positioning. If not used carefully they can overlap other elements. We can fix this by shifting our header text over and out of the way of the image by adding **padding-left of 85px to our h1 rule**. If we tried to add margin-left the container block for <h1> would simply move over with the image still obscuring the text.

Z-index and the layer

Because absolutely positioned elements are taken out of the flow of the document, they can stack up similar to what we saw with our image and header. The default order of these "layers" is the source order. You can control the stacking order of these boxes by setting a property called the z-index. This only works when the objects you are "layering" are absolutely positioned, so this would not have fixed our header text. Open `absolute-z-index.html` in your tutorial file. Both the red (AP one) and green (AP two) blocks are absolutely positioned. Because the AP two block comes after the AP one box in the source order, it is on the top "layer" and covers part of AP one. Go to your CSS Panel and select AP one and click on the edit tool to open the CSS Rule Definition dialog box. Choose the Positioning category and for Z-index enter a value of 100. The higher the z-index, the higher up the box appears in the stack. In design view you will see that the green AP one is now on top covering part of AP two. You do not have to create z-index numbers in sequential order, just smaller to larger controls the order. Usually designers leave some gap in the numbering, using 100s or 10s, to avoid re-numbering in case another z-indexed element needs to be added.



NOTE: Always test your absolute placement elements in the browser by adjusting the View>Text size, to make sure that enlarging the text does not move things into positions that relatively or absolutely positioned element can obscure them.

There is a bug in IE5.5 and IE6. If you position something absolutely bottom right, you must have some dimension set (width or height) on the relative ancestor container that you wish to place it within, or IE will ignore the container and position it within the viewport.

Fixed Positioning

Fixed positioned elements are moved out of the normal flow entirely - relative to the viewport. This means that they don't move if the page is scrolled. Win/IE6 and below do not support this positioning method at all. Basically fixed positioning is a sub-category of absolute, where regardless of the positioned ancestors, the containing block is the viewport. This allows you to create floating elements that always stay in the same position in the window. For an example see: http://www.w3.org/Style/CSS/

Static Positioning

A statically positioned box is one that is in the normal position based on the document flow.

Floats

Floating a box will either shift if to the left or the right until its outer edge touches the edge of its container, or another floated box. It is not in the normal flow. Other block boxes behave as if the floated box wasn't there. Floats are not actually positioning rules, and are meant to provide a way to wrap content around elements. But designers have come to rely on them to get desired layout effects.

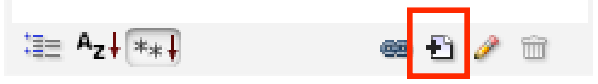
Open `float.html` in Dreamweaver.

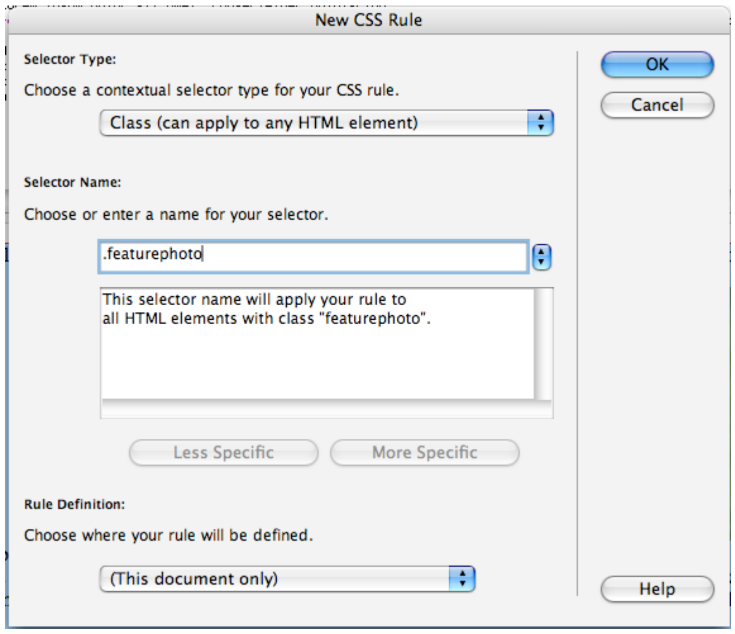
This page already has some CSS Rules embedded in the head to help us visualize the float principals: a style to outline the `p` and `h1` elements in red with a background of light blue, a style to give the strong element a red color and a style to give `img` element `margin: 1em` on all sides.

Insert the image `dandelion.png` after the marker text **[photo here]** and after the **`<strong>`** element in the first paragraph. (To make sure you are outside the <strong> tag, click inside the [photo here] text, go to the tag selector bar, select <strong> and use your right arrow key to move off it to the right. Check your code view, you should see the insert I-beam after the closing </strong> tag and then insert the image.)

The marked text **[photo here]** tells us where the image is in the source order compared to where it floats. In the CSS panel select the image and in the tag selector bar, right click on the tag `img` and in the **box category** of the CSS rule dialog add the property **`float: left`**. Notice in the design view that the image is now moved to the left and the paragraph text is *wrapping* around it, much like you might do in MS Word or InDesign.

We don't want all images we use on this page to necessarily behave this way, so we will delete the property from the `img` element style and create a custom class css rule **`.featurephoto`**. In the CSS Panel , click the new rule icon to get the New CSS Rule dialog window




New CSS Rule

Selector Type:
Choose a contextual selector type for your CSS rule.
Class (can apply to any HTML element)

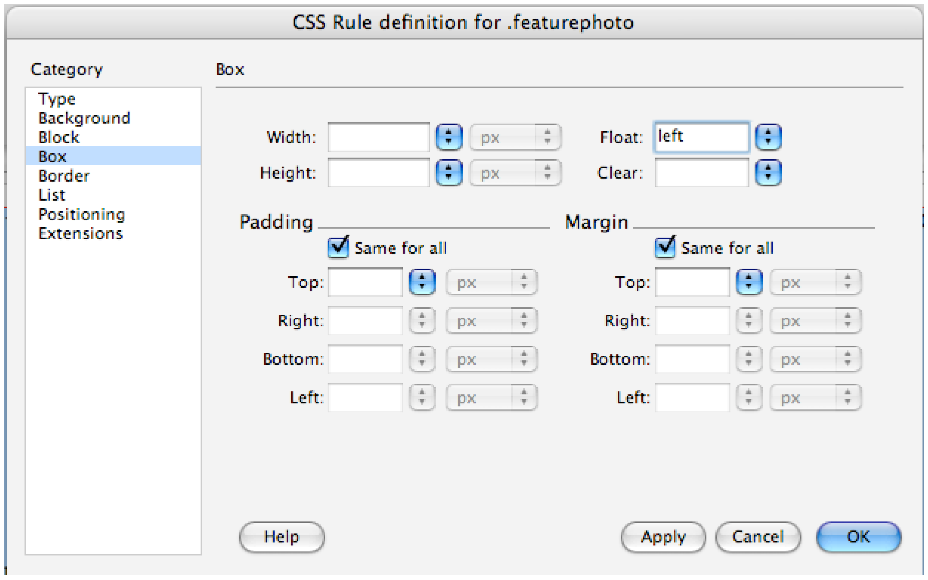
Selector Name:
Choose or enter a name for your selector.
.featurephoto

This selector name will apply your rule to all HTML elements with class "featurephoto".

Rule Definition:
Choose where your rule will be defined.
(This document only)

Making sure Class is chosen, type the new custom class name .featurephoto and make sure This document only is selected, and OK.

In the CSS Rule definition dialog, go to the **Box category** and set **`float: left`**.

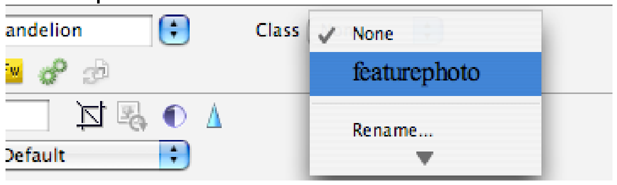


Width: px
Height: px
Float: left
Clear:

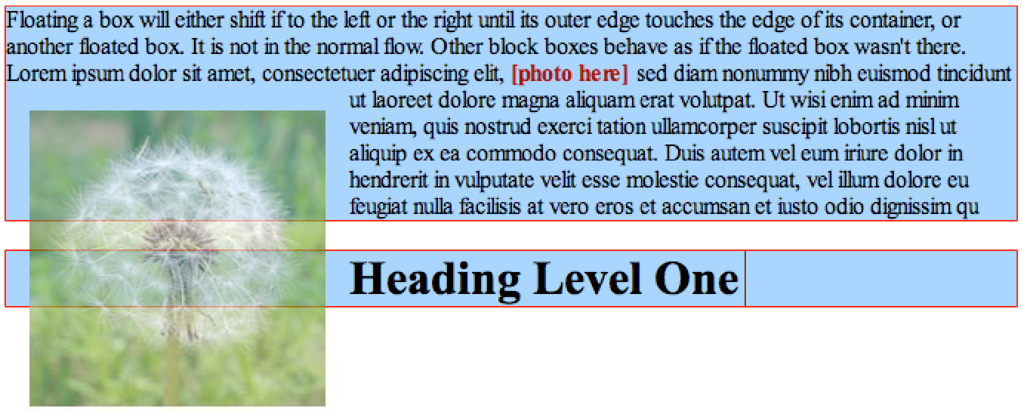
Padding
[✓ Same for all]
Top: px
Right: px
Bottom: px
Left: px

Margin
[✓ Same for all]
Top: px
Right: px
Bottom: px
Left: px

Back in the design view, select the image and in the properties panel apply the css rule .featurephoto.



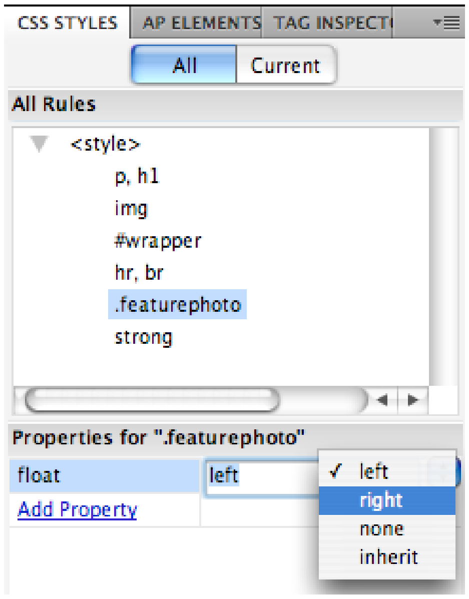
You can see that the image has now floated all the way to the left edge of the body. Notice that it has been removed from the normal document flow, where we have the marker text **[photo here]**. Also note that the block elements', paragraph and h1, borders and backgrounds boxes extend behind the floated image, and the text is wrapping around the image with 1em margin.



Floating a box will either shift if to the left or the right until its outer edge touches the edge of its container, or another floated box. It is not in the normal flow. Other block boxes behave as if the floated box wasn't there. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, **[photo here]** sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qu

Heading Level One

Let's edit the css rule `.featurephoto` to float the image to the right. In the CSS Panel, select **.featurephoto** from the list of rules, go down to the properties section of the panel and click on the float property value and change it to right.



See how the image now floats all the way to the right and the text is wrapping around the left side. In the CSS Panel, selecting .featurephoto from the list of rules, go down to the properties section of the panel and click on the float property value of right and change it back to **left**.

Even though the `h1` element is a block element, it will not drop below the floated element and start a new block. In order to force an element to **"clear" the float and start a new block,** we must use the clear property. There are three values, **`clear: left, clear: right, clear: both`**. The **`clear: left`** is what we need here to force the heading to "clear" or drop past the left floated floating image and start a new block. Place your cursor in the h1 element and in the CSS panel, create a **new CSS Rule** for the just the tag element `h1`. In the CSS Rule definition, select the **Box** Category and set **`clear: left`**. Now we see that the heading has dropped below the image and starts at the left. In the CSS Panel All view, delete the last `h1` rule that we made and let the heading return to the wrapping position. Now select and drag and drop the image to the end of the heading text. Even though it still has the float property, it does not affect the elements that come before it, only elements that follow in the document flow. Undo the image move (CMD or CTRL Z).

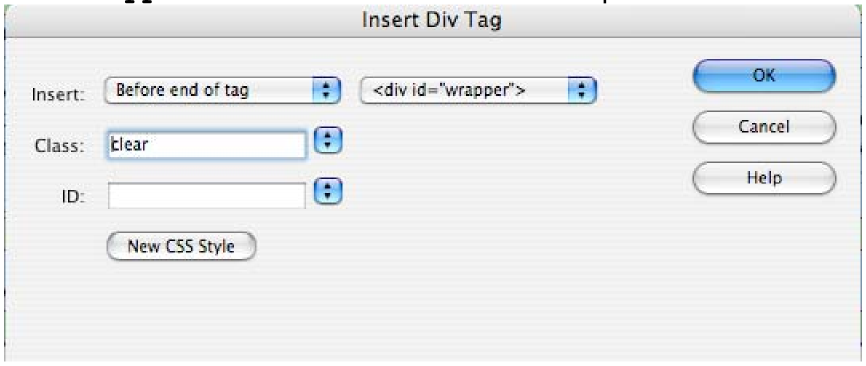
Now we want to put our all of our text and the image into a division and add a background color. Select all the body content and from the Insert Panel, select the Insert Div Tag icon and choose wrap around selection and choose the ID "wrapper". Or in the code view, add `<div id="wrapper">` opening tag and closing `</div>` tag to contain our all the content, inside the <body> element. The `#wrapper` rule with a background color has already been defined in the embedded head css. Notice that the background color is not encompassing the full image. Part of the image is hanging out. Just as we saw the image extend beyond the individual block paragraphs and heading, the block element of a division also does not contain the float.

Now go to the code view and delete the all paragraph text and heading text, leaving only the image and observe what happens to the `div` with our background color. Preview it in Firefox and you will see that there is nothing left visible of the div except the border. Because the floated image is taken out of the flow of the document, the wrapper div has no actual content and collapses. (NOTE: IE6 and below ignores this css standard behavior if the container has a width OR height, so be sure you are viewing with Firefox or another compliant browser.)

Undo (CMD or CTRL Z) to restore the content.

You will often want to have a background image or color, or border to surround all the content in a `<div>` that has floated elements that extend beyond the content. To do this we will use the clear property again; this time to trick the div into containing the floats. But we don't want to add `clear` to change the position of any of the wrapping text already here, so we have to add something to force the "clear." For this demonstration, we will add an empty <div>, just inside the <div id="wrapper">, using custom class of `.clearfloat` on the extra div, to get the needed clear property. Create a new CSS Rule for a custom class named `.clearfloat`. In the CSS Rule definition, choose **Box** category and set**, clear:both;** and **height:0;** in the **Text category** set **font-size: 1px; line-height: 0px.** The extra properties and values for height, font and line-height are used to make this clearing trick work in all browsers. Wow, our first hack! Since a div has no semantic meaning and we set the height is zero, this div will not have any impact on your content or spacing of elements. This will let us use this class anytime we want to clear floats, whether they are right or left. (Although sometimes you will need to be very specific if you don't want to break a layout that has other floats.)

Place your cursor after the heading level one text and from the Toolbar Menu select Insert>Layout Object>Div tag. In the dialog box, choose **Before end of tag**, select **`<div id="wrapper">`** and select **Class: clear** and press OK.


Insert Div Tag

Insert: Before end of tag — <div id="wrapper">

Class: clear

ID:

Then immediately press delete to get rid of the extra placeholder text that DW put in this new div. Now see that the container div wrapper now completely contains the floated element and the background color and border extends around the content as desired.

There are other ways to accomplish the desired result and may be used if better suited to your overall layout, but they may have undesired effects in some browsers:

Remove the extra div tag in the code view.

- o Add a float to the container. Create the following rule or add these properties to your container id rule (replace *wrapper* with your container name. Choose float:right or float:left depending on your layout, and set a width that is appropriate):

```
 #wrapper {float:left;
          width: 100%;}
```

  Check it in the browser. Then Undo (CMD or CTRL Z)
- o Add overflow:auto to the container. Create the following rule or add these properties to your container id rule (replace *wrapper* with your container name and set a width that is appropriate):

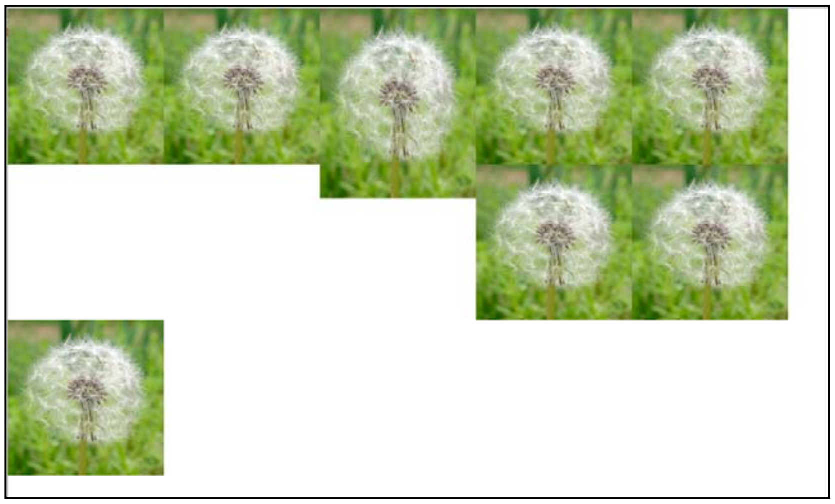
```
#wrapper {overflow:auto;
          width: 100%;}
```

For detailed explanations and solutions go to:
http://www.complexspiral.com/publications/containing-floats/
http://www.ejeliot.com/blog/59
http://css.maxdesign.com.au/floatutorial/

With your cursor at the end of the heading level one, insert two more short paragraphs of Loreum Ipsum text and you will see that once the block elements are past the bottom of the floated element, we see the normal behavior of starting the new block at the left and the clearing div would no longer be necessary as long as the text or other content extended beyond the floated element.

Practice Floats

Open `float2.html` There are nine images inside individual divs. Since the divs are block items, they naturally stack up starting a new row for each div and we can observe how floated block items work. Now add the css style `.floatright` to the first image, save and preview in the browser. Now change the style for the first image to .floatleft. Do this again for the second and third image. Go ahead and change the img element to have the property float:left, so that all the images will automatically float to the left. Preview in the browser and note that if you change the window size, the images will drop down to the next row when there is not enough room to the left. Return to Dreamweaver and in design view, make the third image taller by simply selecting the image and dragging the resize handles down to increase the size by about 20 pixels. Save and preview in the browser again. Notice that the images following the now taller 3rd one do not go all the way to the window left edge, but hang up on the bottom of the taller image and then only after the images "clear" this taller image in the next row, do they go all the way to edge again.



Creating Multi-Column Pages with Floats

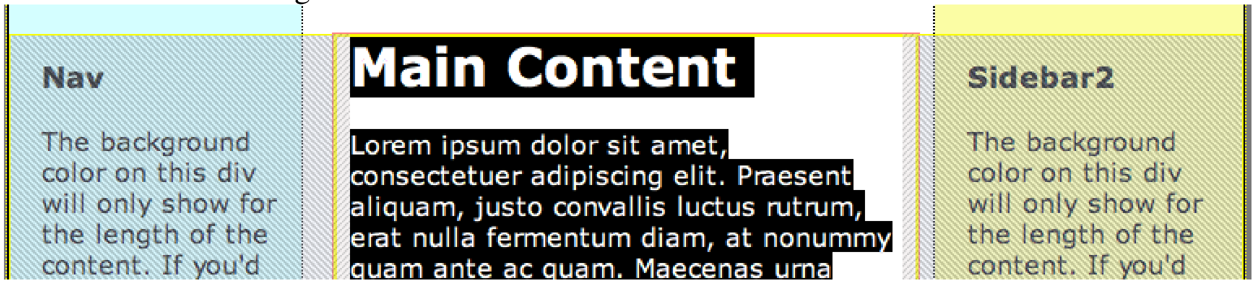
Open right-left-col.html. The document already has some semanticly structure html content. Notice that there are 4 divisions; container, nav, sidebar2 and maincontent. We want to layout the page with the nav on the left, sidebar2 on the right and maincontent in the middle. If we preview the page in Firefox, we will notice that the page is not centered. Add the normal centering rules, margins right and left to auto. Note that DW has written the shortcut for the margins as margin: 0 auto, which stands for top and bottom are zero and right and left are auto.

Internet Explorer 6 does not honor this property, so the easy way is to add a slight hack and tell the body to center text (text-align: center). Add this property and value to the body rule. But now all of our text is centered, probably not the look we want. Since we have a #container div that wraps all of our content, we can change it back to left aligned, selecting the #container rule and adding text-align:left to override the body property for all of our content. Isn't it exciting to make our second hack! Gotta' love those browsers.

Now we are going to make some columns. In the CSS Panel select the #nav selector rule and in the CSS Properties below, add a new property for float: left and since it is floated, it should have a width. Set it for 170px.

Now choose the CSS selector rule for #sidebar2 and set it to float: right and set the width for 170px.

That takes care of our two side columns, but notice that the central #maincontent is spilling out below the to side columns. This is because it is wrapping around the floated divs. In order to give the appearance that this content is in a separate column, regardless of how long the side columns or the maincontent are, we add margin left and right to keep the maincontent in a narrower column. Select the #maincontent rule and add the property margin: 0 200px to set the top and bottom to zero margin and the left and right to 200px margins (20 + 170 + 10). If you hover near the edge of the maincontent in the design view to get a pink border and then click, Dreamweaver will "visualize the margins.



The background color on this div will only show for the length of the content. If you'd

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam. Maecenas urna

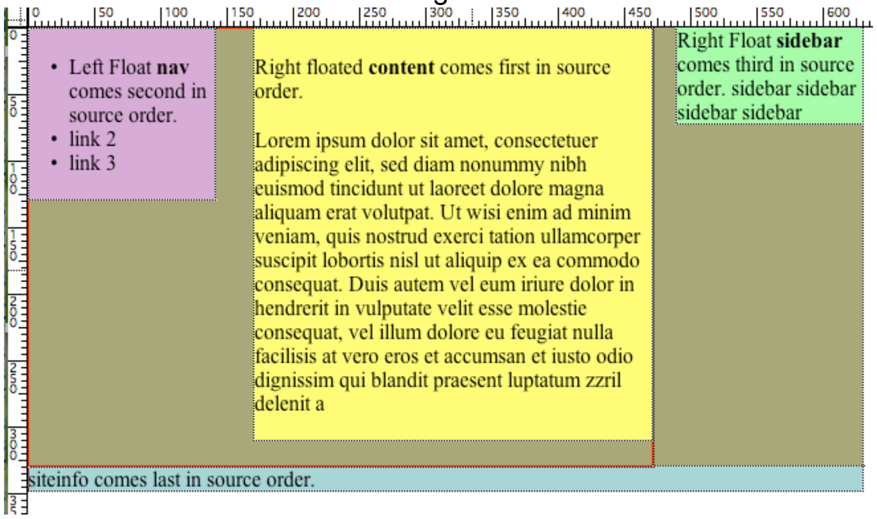
The background color on this div will only show for the length of the content. If you'd

If you wish to change the position of the content, you can simply go to the rule for #nav and #sidebar2 and change their float values. Notice that we didn't set a width on the center #maincontent. If we want this content to be liquid and have more room if the browser viewport is wider, we can change the #container from a fixed width to a percent. Select the #container and in the Box category, set the width to 80%.

Source Order and Positioned Content

One advantage of positioning with css is that the order of the content in the html can be organized logically, with say, the main content first and then the sidebar; navigation next and site info usually placed in the footer last. Yet the designer can visually organize the information in a variety of ways on screen, so that the navigation appears first on the left or top, and content below or to the right. Open `source-order.html`. The four divisions are inside a container wrap for easy viewing, and as you can see in the design view, the order is content, nav, sidebar, and last siteinfo. We want our nav to appear in the left column, then our content in the middle and sidebar on the right. The site info should be below all 3 columns and last on the web page.

To accomplish this visual order, we do not change the html, except to add a wrapper around the content and nav divs so that we can shift them with floats. In design view select all the content in the first two divisions, starting with ***"Right floated content***" and ending with ***"Link 3".*** On the **Toolbar Menu** select **Insert>Layout Objects>Div tag** and choose Wrap around selection and select class **.floatwrapper** that has already been defined. You will see a red border defining the edges of this container. We are going to float the content right and the nav left to get our first two columns positioned.


- Left Float **nav** comes second in source order.
- link 2
- link 3

Right floated **content** comes first in source order.

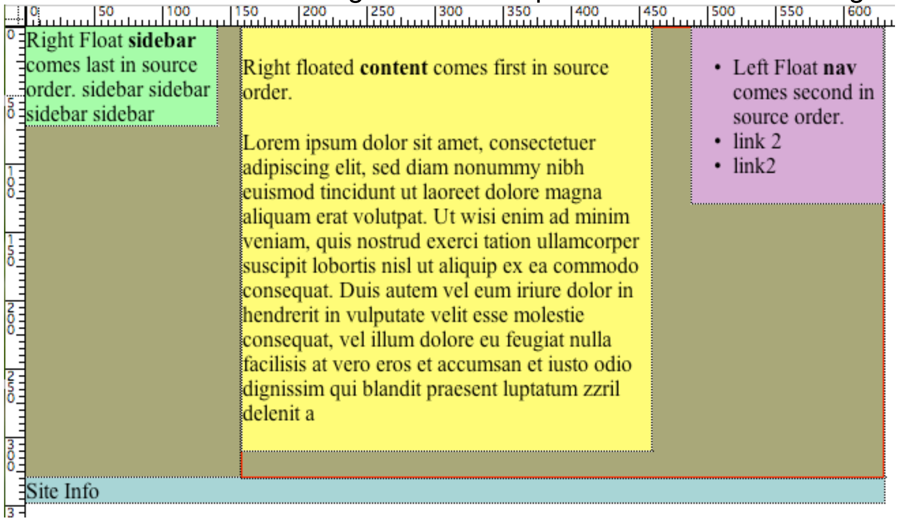
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit a

Right Float **sidebar** comes third in source order. sidebar sidebar sidebar sidebar

siteinfo comes last in source order.

Uncomment these two properties in the code in the head for the rules .containter and .nav. Now we want to float the sidebar to the right of the floatwrapper content to position it on the right side of our layout. Uncomment the float: right property for the .sidebar rule in the head. The .siteinfo rule already has the property clear:both to allow it to clear both the right and left floats in our previous columns.

Even more powerfully we can make a few changes to reposition the two outside columns. In the CSS panel, change the `.floatwrapper` to **float: right**. Change the `.nav` column div to **float: right**, change the `.content` column div to **float: left** and finally change the `.sidebar` to **float:left**. We have rearranged our visual positions without touching the html!



Right Float **sidebar** comes last in source order. sidebar sidebar sidebar sidebar

Right floated **content** comes first in source order.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit a

- Left Float **nav** comes second in source order.
- link 2
- link2

Site Info