Browsers have default margins and paddings built into them. To get rid of these defaults.

body {

margin:0;

padding:0;

}

To get rid of the default margins and paddings between the divs, do following:

p {

margin:0;

padding:0;

}

Working with **CSS reset** will help you ensure that your web app looks exactly the same in each and every browser.

Padding is pretty similar to margins but it works on the inside of an object rather than the outside.

margin:100px will put a margin of 100px all the way around the box.

The width of a box is width: + padding-left: + padding-right:

The height of a box is height: + padding-top: + padding-bottom:

With margin *position* is not used.

Cool website <http://html-color-codes.info/>

Color zilla Chrome addin

Inline CSS

<p style=”color:green; font-size=50%”> </p>

Internal CSS

<head>

<style type=”text/css”>

p: {

color:green;

font-size:200%;

}

h1:{

color:pink;

}

</style>

</head>

Classes and IDs

Classes are applied throughout the page, wherever the class is mentioned.

IDs are applied only *once* per page.

<head>

<style type=”text/css”>

.red{

color:red;

}

.large {

font-size:200%

}

#green{

}

.underline{

text-decoration:underline;

}

#first-section{

color:blue;

background-color:pink;

width:50%

width:100px

float:right;

}

#second-section{

background-color:yellow;

float:left;

}

</style>

</head>

<body>

<div id=”first-section”>

<p class =”red large”> The quick brown fox jumps over the lazy dog.</p>

<p class=”large”>Wow, I love internal CSS</p>

</div>

<div id=”second-section”>

<p id=”green”>This is some more text. <span class=”underline”>And this text is underlined.</span></p>

<div>

</body>

**Floating**

.clear-float{

clear:both;

}

<div class=”clear-float”></div>

Floating is very flexible. But when you want to be precise about the location of the elements, then use Positioning.

**Positioning**

**position:relative;**

z-index:1;

z-index:2;

In case of overlapping elements, the element with the highest z-index will show over the top.

opacity: (range is between 0 and 1). Opacity of 0 will make the element hidden. Opacity of 1 will make it fully opaque.

opacity:0.5

**position:absolute;**

Absolute positioning will make the position of *all* the elements absolute. So, if you don’t provide the values for left, right, top, and bottom, *all* the elements will move up to the top-left corner of the page.

Absolute positioning takes the float out of the page! So, in my opinion, the page is not responsive any more.

**position:fixed;**

Fixed position can be used if you have a menu at the top of the page, and you want to show the menu even if the user scrolls the page vertically.

Difference between fixed and absolute positioning.

When a page is scrolled,

Element with position:fixed; will stay on the same place on the page, meaning it will always appear on the page.

Element with position:absolute; will move. So if we scroll further down, eventually the element will not show on the page.

**Margins**

Difference between using Relative positioning and margins.

margin:100px; will put a margin of 100px all around the div. (margin of 100 for top, bottom, left, and right).

margin can be used along with float.

margin: 100px 50px; will set the top and bottom margins to 100px and left and right margins to 50px.

margin: 100px 50px 20px 10px; (top=100, right=50, bottom=20, left=10)

margin-left:20px;

margin-right:

margin-top:

margin-bottom:

|  |  |
| --- | --- |
| **Use margins when you want** | **Use positioning when you want** |
| elements to remain in the flow of the page. | elements to take out of the flow of the page |
| To spread everything out relative to it. | Move them around on top of the other elements (and that sort of the things). |

**Padding**

|  |  |
| --- | --- |
| Margins are outside an element (such as a div) | Padding is inside of an element. |

width:300px;

padding: 5px 10px 12px 15px;

The total width of the element (div) would be 300 + 10 + 15 = 325 px.

**Borders** border: 1px black solid [width] [color] [style]

border-radius:20px; will make the edges round for a square div.

border-radius:50% will create a circle out of a square div.

**Fonts:**

Search for CSS text styling

Search for web safe fonts. (Fonts supported by all browsers).

Search for google fonts.

Search for pseudo classes links

font-family: “Times New Roman”, Times, serif

font-weight: bold;

font-style: italic;

text-decoration: underline;

If you want all of the web links not to be underlined, you do

a {

text-decoration: none;

}

**Pseudo classes**

a:hover{

text-decoration: underline;

color: green;

}

a:visited{

color: green;

}

Search for css styling text

<http://www.w3schools.com/css/css_text.asp>

|  |  |  |
| --- | --- | --- |
| **CSS tag** | **Not allowed.** | **Use instead** |
| <span> | margin:  margin-top:  … | position:relative;  top:14px;  padding-right:50px; |
| <div> |  | For <div>s, we can use margins and floats. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

If at one place you are setting a border, and for the same class/id you are setting border:none, then putting !important will solve the issue.

**HTML5 and CSS**

<http://Caniuse.com> will tell you what browsers support a particular HTML5 feature.

<http://html5pattern.com/> for a lot of Regex expressions that you can use for text pattern attribute.

<p>Code outside form: <input form="myForm" type="text"></p>

When data within the form is submitted, then the data above will also be submitted with the form.

<p><input type="submit" value="submit" formaction="processForm2.php"></p>

Code above will submit the form data data processForm2.php, even though form action is set to submit to processForm.php. So you can have multiple submit buttons, each submitting the same form to a different location/file.

**formethod** is used to specify which method the submit button will submit with. (Get, POST, DELETE, UPDATE) etc.

<p>Multiple file upload: <input type="file" multiple></p>

**multiple** will allow to add multiple files at a time.