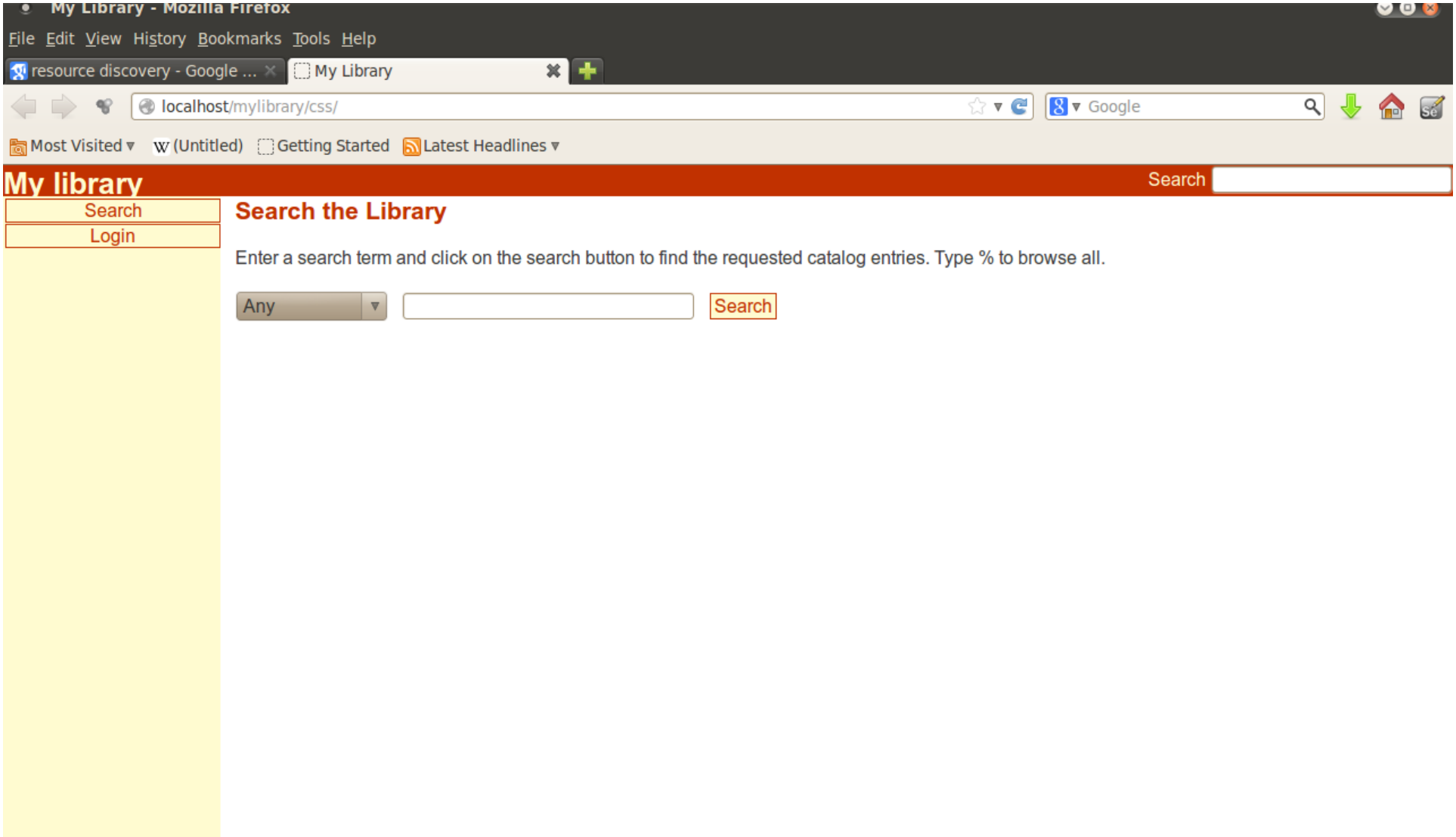
CSS

Example Web Page Document and Presentation



Document and Presentation HTML way

*index.html (/var/www/mylibrary/html) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

*index.html

```
<html>
 <head>
  <title>My Library</title>
 </head>
 <body>

     <table border=0 width=100% cellpadding=3 cellspacing=0 bgcolor="#C13100">
      <tr>
       <td>
        <font size=6 color="#FFFBD0">
          <b>My library</b>
       </td>
       <td></td>
       <td align=right>
         <form action=actions/search.php method=GET><font size=2 color="#FFFBD0">
          Search <input type=text name=q>
          <input type=hidden name=criteria value=Any>
         </form>
       </td>
      </tr>
     </table>
     <table width="100%" height="95%"border="0" cellspacing="0" cellpadding="0" bgcolor="#FFFBD0" >
      <tr>
       <td valign=top width=140>
```

Document and Presentation CSS way

*index.html (/var/www/mylibrary/css) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

*index.html

```
<html>
 <head>
  <title>My Library</title>
  <link rel="stylesheet" type="text/css" href="reset.css">
  <link rel="stylesheet" type="text/css" href="stylesheet.css">
 </head>
 <body>
     <div class="header">
         <div class="title">My library</div>
         <div class="search">
          <form action=actions/search.php method=GET>
            Search <input type=text name=q>
            <input type=hidden name=criteria value=Any>
           </form>
         </div>
     </div>
     <nav class="leftNavigation">
          <a class="button" href="index.html">Search</a>
          <a class="button" href="login.html">Login</a>

     </nav>
     <section class="mainContent">
          <h1>Search the Library</h1>
          <br>
```

CSS Stylesheet

stylesheet.css (/var/www/mylibrary/css) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

*index.html stylesheet.css

```
body{

     font-family:arial;
     font-size:12pt;
}

.header{

     position:static;
     top:0; left:0;
     z-index: 10;
     width:100%;
     height:20pt;
     background-color: #C13100;
     color: #FFFBD0;

}




.header .title{
     font-weight:bold;
     font-size:20pt;
}
```

CSS rule

Syntax

```
selector(s) {
	property-name:property-value;
	property-name:property-value;
	...
}
```

Example

```
body{
	font-family:arial;
	font-size:12pt;
}
```

Selectors

- Selector represents a structure in HTML / XML document
- Ranges from simple to complex contextual representations
- Selectors are used to search matching elements in the document and apply the styling rules / properties

Examples

Selector	Type	HTML	Description
body { … }	Type / Element	<body> … … </body>	Matches the body element of HTML document
.header { … }	Class	… … … …	Matches all elements having a class='header'
#searchButton{...}	Id		Matches an element having an id='searchButton'

Advanced Selectors

- Leads to more specific selection
- Relies on combinations of selectors
- Different combinators lead to different meanings

Simple combination Examples

div.header: All **div** elements having **class='header'.** Ignores other **div** elements

div#header: Only the **div** element having id**='header'.** Ignores other **div** elements or any other element having id='header'

Grouping Examples

.header , .footer , .menu: selectors separated by comma represents a union or group. Applies the stated properties to all selectors in the group

Advanced Selectors

Nesting Examples

Descendants

h1 em: Separated by space. It represents an **em** element being the descendant of an **h1** element. It is a correct and valid, but partial, description of the following fragment

```
<h1>This <span class="myclass">headline
is <em>very</em> important</span></h1>
```

Children

body>p: Separated by >. It represents a **p** element that is a child of **body**

div ol>li p: Represents a **p** element that is a descendant of **li** that is a child of **ol** that is further a descendant of **div**

Advanced Selectors

Attribute Selectors

select elements based upon the attribute

```
a[target] {
    background-color: yellow;
}
```

or based upon the attribute values

```
a[target="_blank"] {
    background-color: yellow;
}
```

or other combinations such as contains, begins or ends with a specific value

Pseudo-classes and Pseudo-elements

- Pseudo-classes
  - represent a special state of an element
  - useful for styling on special actions (e.g. focus, hover, visited, etc)
  - may be used for selecting specific elements (e.g. first-child, last-child, nth-child(n) etc)

```
a:hover {
    color: #FF00FF;
}
```

```
p:first-child {
    background-color: yellow;
}
```

- Pseudo-elements
  - represent a special portion of an element
  - useful for styling specified parts of an element (e.g. first letter, line, etc)
  - may be used for inserting content before or after an element

```
p::first-line {
    color: #ff0000;
    font-variant: small-caps;
}
```

```
h1::before {
    content: url(smiley.gif);

}
```

Where to specify CSS Rules

External CSS sheet

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

Internal CSS

```
<html>
  <head>
    <style type=”text/css”> … </style>
  </head>
</html>
```

Inline CSS

```
<div style=”font-weight:bold; text-decoration:underline; ...” > … </div>
```

Origin

- **Author:** The author specifies style sheets for a source document according to the conventions of the document language. For instance, in HTML, style sheets may be included in the document or linked externally.
- **User**: The user may be able to specify style information for a particular document. For example, the user may specify a file that contains a style sheet or the user agent may provide an interface that generates a user style sheet (or behaves as if it did).
- **User agent:** Conforming user agents must apply a default style sheet (or behave as if they did). A user agent's default style sheet should present the elements of the document language in ways that satisfy general presentation expectations for the document language

Inheritance and cascade

- Elements can inherit properties of parents
- The final result is a cascade / aggregate according to CSS cascading rules

HTML

```
<h1>The headline <em>is</em> important!</h1>
```

CSS

```
h1 { font-weight : bold }
em { text-decoration : underline }
```

Result

```
h1: bold
em : bold + underline
```

Inheritance and cascade

- Conflicts need to be resolved in cascade

HTML

```
<h1>The headline <em>is</em> important!</h1>
```

CSS

```
h1 { font-weight : bold }
em { font-weight : normal ; text-decoration : underline }
```

Result

```
h1: bold
em : normal + underline
```

Calculating Selector's Specificity

- Specificity is determined by concatenation of four values: a,b,c,d
  - **a:** 1 if the declaration is inline, 0 otherwise
  - **b:** number of **ID** attributes in selector
  - **c:** number of **class** (other attributes) in selector
  - **d:** number of element names

- Concatenate values of a, b, c and d to arrive at a numeric value
- Higher number means more specificity

Examples

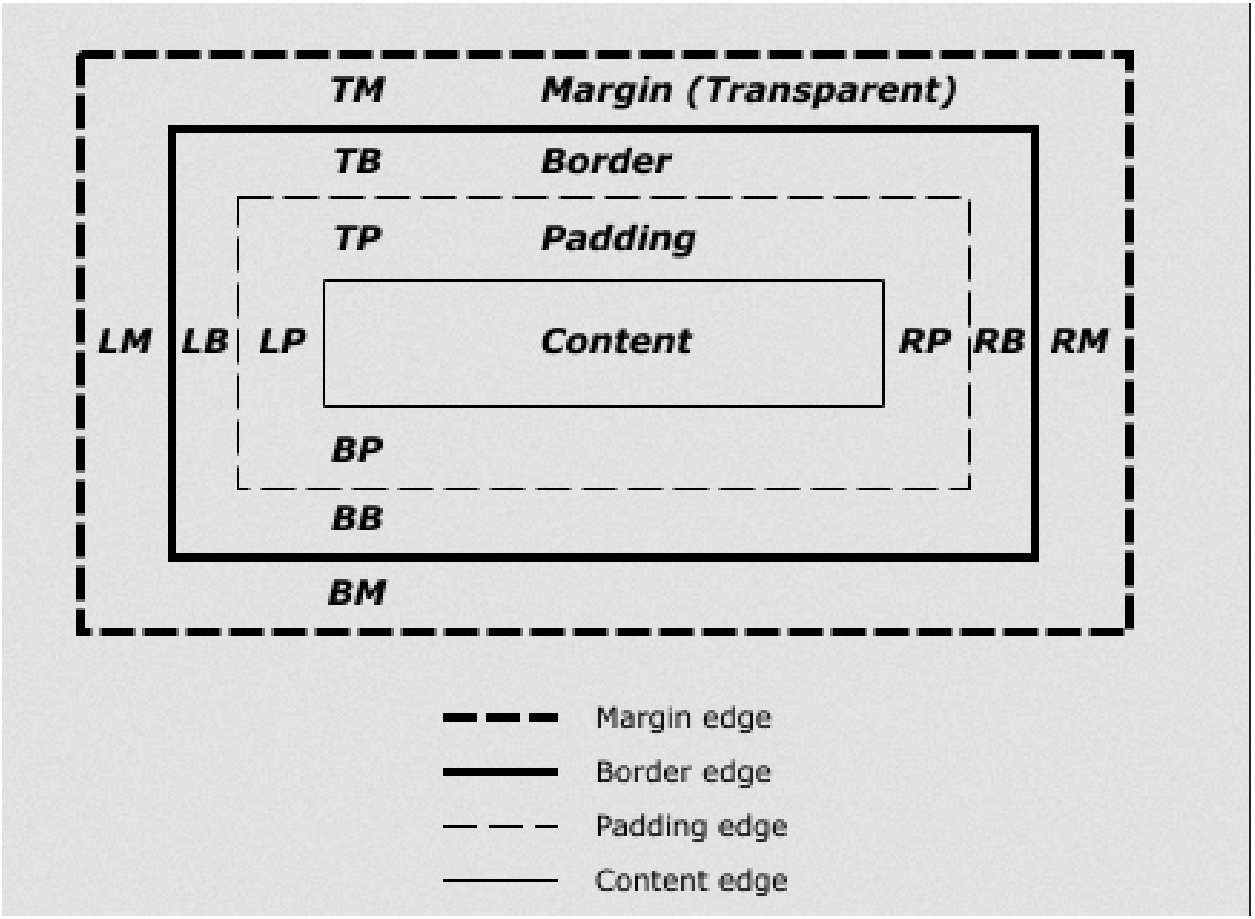
Selector	a	b	c	d	Specificity
*	0	0	0	0	0000 = 0
li	0	0	0	1	0001 = 1
ul li	0	0	0	2	0002 = 2
ul li.red	0	0	1	2	0012 = 12
#item	0	1	0	0	0100 = 100
li#item	0	1	0	1	0101 = 101
style=”...”	1	0	0	0	1000 = 1000

CSS Cascading Order / Conflict Resolution

1. Find all declarations that apply to the element
2. Sort according to **importance** (normal or important) **and origin** (author, user, or user agent)
    1. User Agent declarations
    2. User Normal declarations
    3. Author Normal declarations
    4. Author important declarations
    5. User important declarations
3. Sort rules with the same importance and origin by **specificity** of selector: more specific selectors will override more general ones.
4. Finally, sort by **order** specified: if two declarations have the same weight, origin and specificity, the latter specified wins.

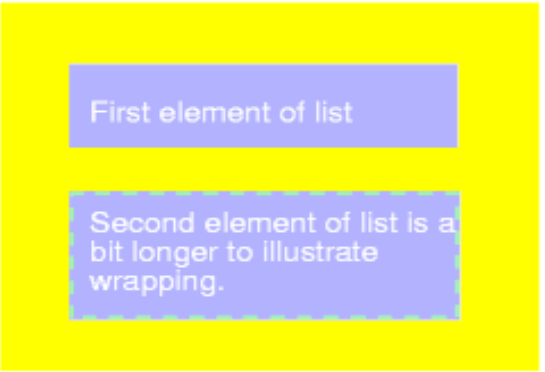
Source: CSS Specification (http://www.w3.org/TR/CSS21/cascade.html#cascade)

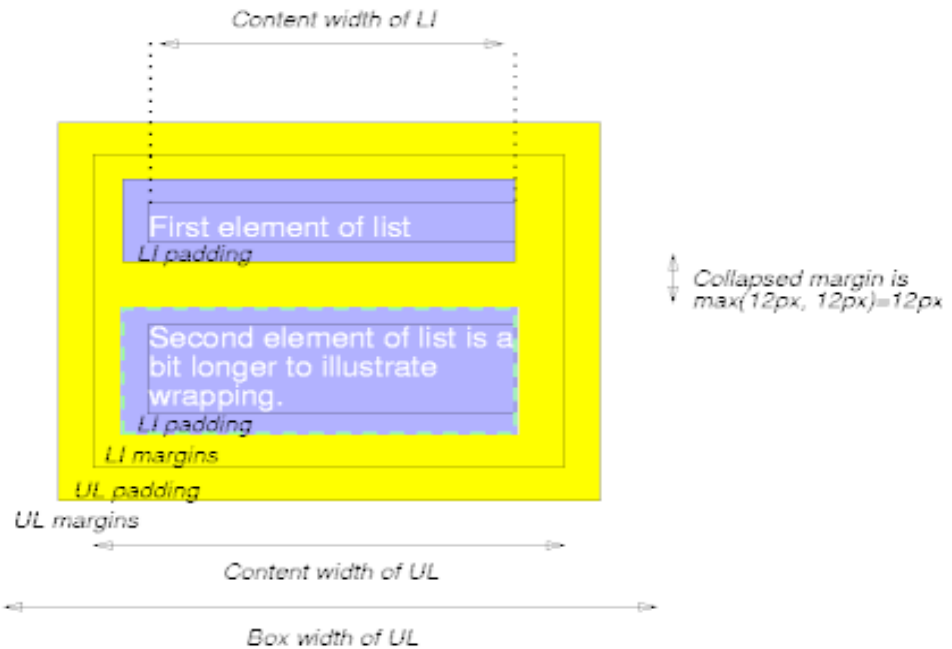
CSS Box Model



- Every element has 4 layers that surround it
- Affects spacing around elements

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
  <HEAD>
    <TITLE>Examples of margins, padding, and borders</TITI
    <STYLE type="text/css">
      UL {
        background: yellow;
        margin: 12px 12px 12px 12px;
        padding: 3px 3px 3px 3px;
                                    /* No borders set */
      }
      LI {
        color: white;               /* text color is whit
        background: blue;           /* Content, padding v
        margin: 12px 12px 12px 12px;
        padding: 12px 0px 12px 12px; /* Note 0px padding
        list-style: none            /* no glyphs before a
                                    /* No borders set */
      }
      LI.withborder {
        border-style: dashed;
        border-width: medium;       /* sets border width
        border-color: lime;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <UL>
      <LI>First element of list
      <LI class="withborder">Second element of list is
           a bit longer to illustrate wrapping.
    </UL>
  </BODY>
</HTML>
```





display

Block level	Inline level
display: block display: list-item display: table	display: inline display: inline-block display: inline-table

- Primarily two types of display: block and inline
- Block level elements are formatted visually as blocks / boxes (e.g. Paragraphs) – vertically stacked
- Blocks may further contain other blocks or inline elements
- Inline content is distributed in form of lines and does not introduce a new block content – horizontally aligned
- display:none doesn't display element at all

Positioning Schemes

- Normal Flow
  - Sequence of elements is considered, following the (block / inline) display rules
- Absolute Positioning
  - Explicitly stated position with respect to parent
  - Disregards normal flow (sequence of elements)
- Float
  - Makes a box shift left or right on the line
  - Content flows down the right-side of a left floated box and left-side of a right floated box

Positioning schemes

- position : static
  - Determine position according to normal Flow
- position : relative
  - First, determine position according to normal flow
  - Then, offset the position relative to normal flow

Normal flow positioning

- position : absolute
  - Determine position relative to the parent / container
  - Parent should be positioned (non-static)
  - Search for a parent recursively till document root <html>
- position : fixed
  - Relative to screen

Absolute positioning

Relationship between display, position & float

- **if** 'display' has the value 'none', **then**
  - 'position' and 'float' do not apply. In this case, the element generates no box.
- **else if** 'position' has the value 'absolute' or 'fixed', **then**
  - the box is absolutely positioned, the computed value of 'float' is 'none', and display is set as specified. The position of the box will be determined by the 'top', 'right', 'bottom' and 'left' properties and the box's containing block.
- **else if** 'float' has a value other than 'none', **then**
  - the box is floated and 'display' is set as specified.
- **else if** the element is the root element, **then**
  - 'display' is set as block
- **else**
  - the remaining 'display' property values apply as specified.

Static Positioning Example

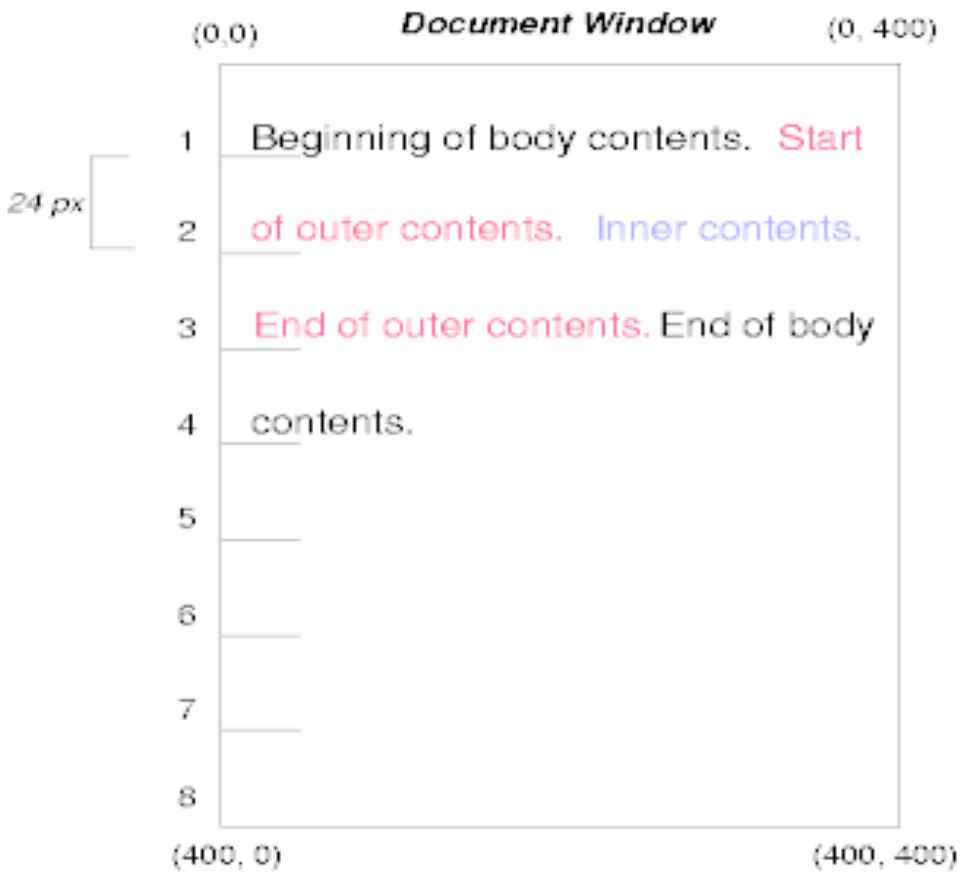
```
<HTML>
  <HEAD>
    <TITLE>Comparison of positioning
                schemes</TITLE>
  </HEAD>
  <BODY>
    <P>Beginning of body contents.
      <SPAN id="outer"> Start of outer contents.
      <SPAN id="inner"> Inner contents.</SPAN>
      End of outer contents.</SPAN>
      End of body contents.
    </P>
  </BODY>
</HTML>
```

```
body { display: block; font-size:12px;
        line-height: 200%; width: 400px; height: 400px
}

p    { display: block }

span { display: inline }
```

```
#outer { color: red }
#inner { color: blue }
```



Relative Positioning Example

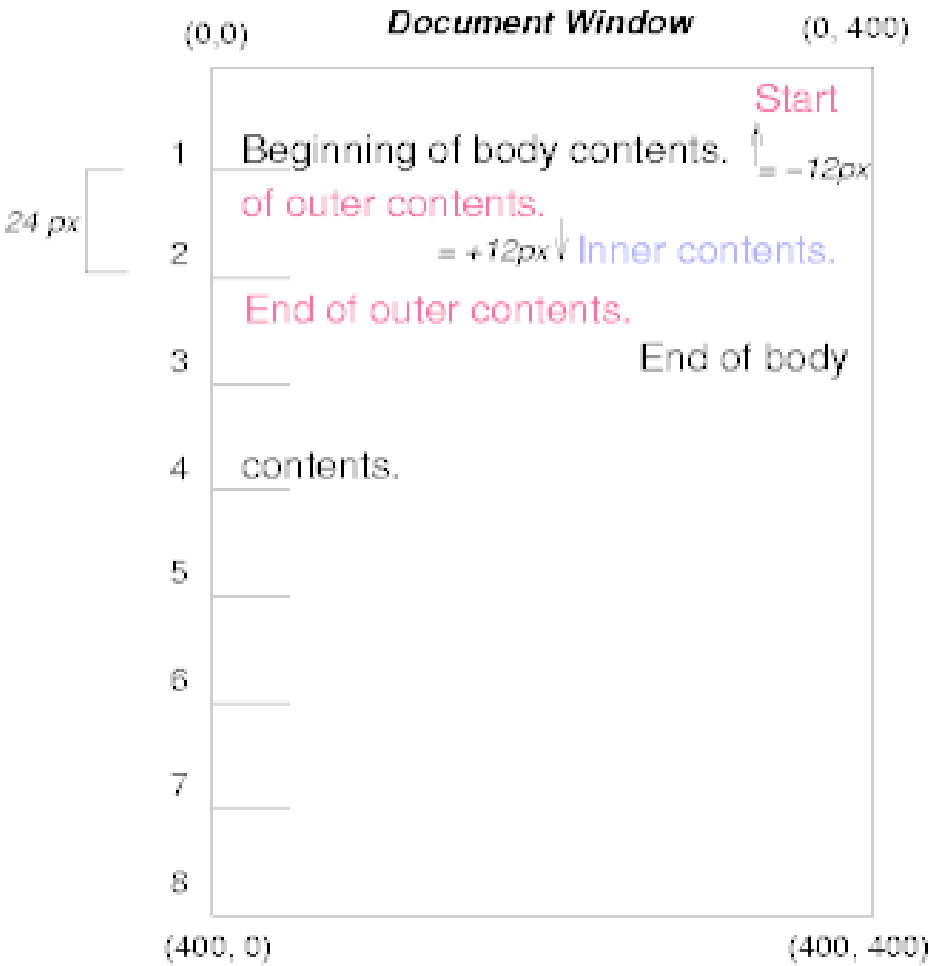
```
<HTML>
  <HEAD>
    <TITLE>Comparison of positioning
                schemes</TITLE>
  </HEAD>
  <BODY>
    <P>Beginning of body contents.
      <SPAN id="outer"> Start of outer contents.
      <SPAN id="inner"> Inner contents.</SPAN>
      End of outer contents.</SPAN>
      End of body contents.
    </P>
  </BODY>
</HTML>
```

```
body { display: block; font-size:12px;
        line-height: 200%; width: 400px; height: 400px
}

p    { display: block }

span { display: inline }
```

```
#outer { position: relative; top: -12px; color: red }
#inner { position: relative; top: 12px; color: blue }
```



Floating Example

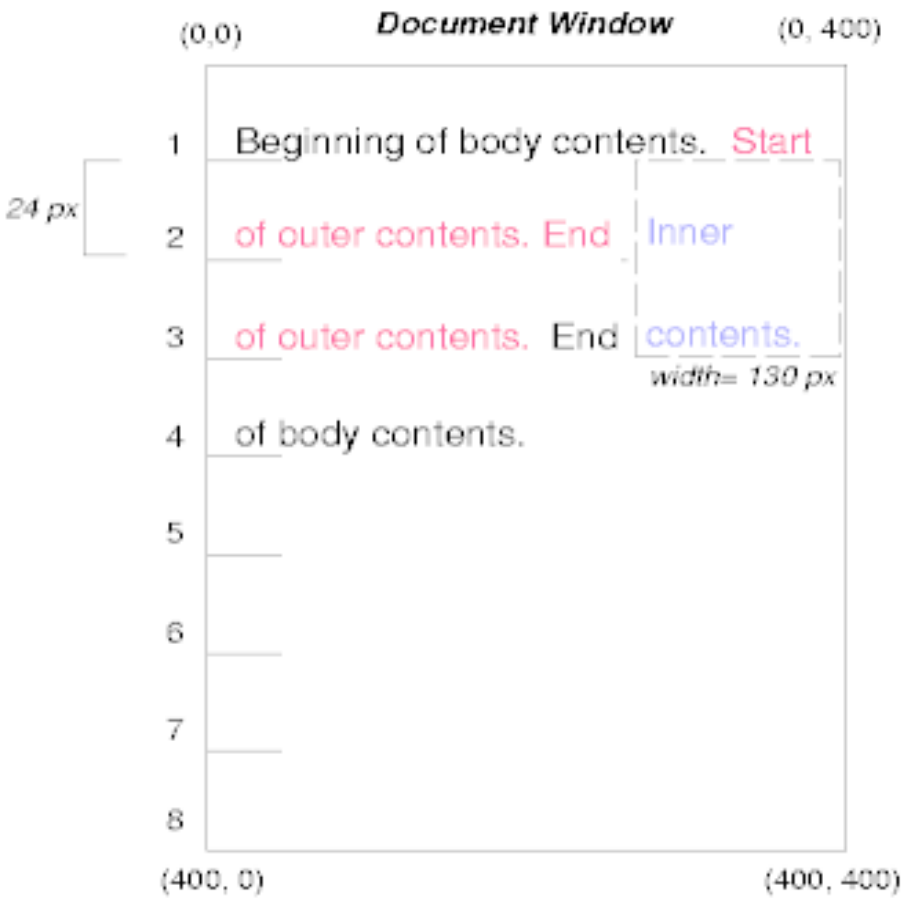
```
<HTML>
  <HEAD>
    <TITLE>Comparison of positioning
                schemes</TITLE>
  </HEAD>
  <BODY>
    <P>Beginning of body contents.
      <SPAN id="outer"> Start of outer contents.
      <SPAN id="inner"> Inner contents.</SPAN>
      End of outer contents.</SPAN>
      End of body contents.
    </P>
  </BODY>
</HTML>
```

```
body { display: block; font-size:12px;
        line-height: 200%; width: 400px; height: 400px
}

p    { display: block }

span { display: inline }
```

```
#outer { color: red }
#inner { float: right; width: 130px; color: blue }
```



Absolute Positioning Example

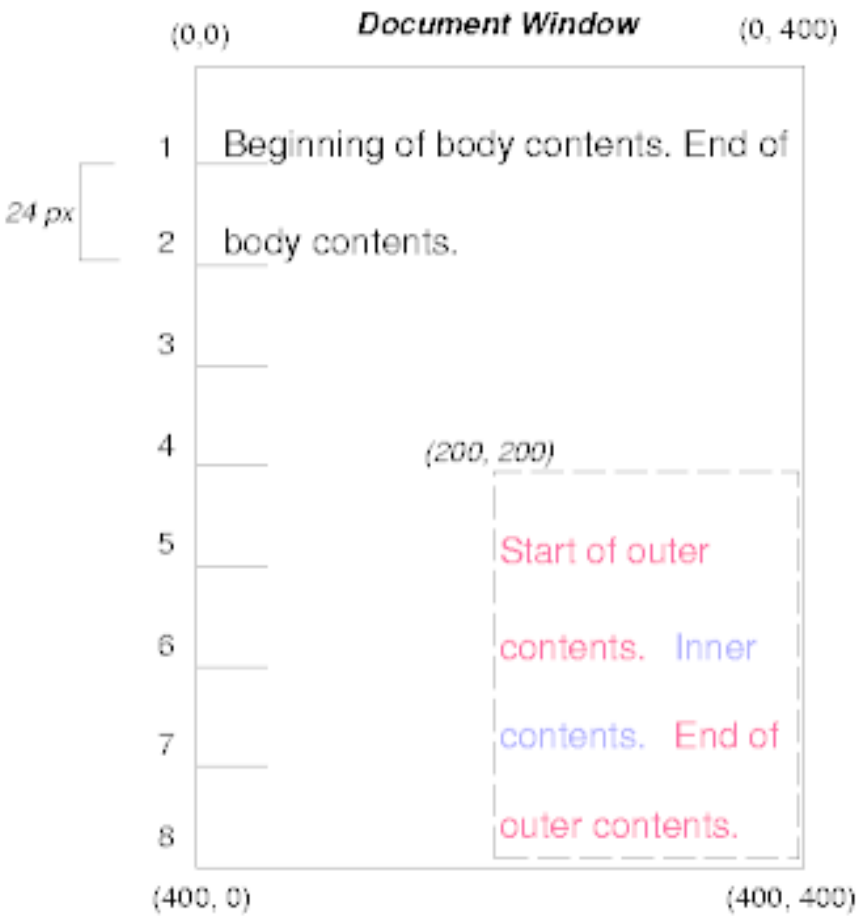
```
<HTML>
  <HEAD>
    <TITLE>Comparison of positioning
                schemes</TITLE>
  </HEAD>
  <BODY>
    <P>Beginning of body contents.
      <SPAN id="outer"> Start of outer contents.
      <SPAN id="inner"> Inner contents.</SPAN>
      End of outer contents.</SPAN>
      End of body contents.
    </P>
  </BODY>
</HTML>
```

```
body { display: block; font-size:12px;
        line-height: 200%; width: 400px; height: 400px
}

p    { display: block }

span { display: inline }
```

```
#outer {  position: absolute; top: 200px; left:
200px;  width: 200px; color: red; }
#inner { color: blue }
```



CSS3

- Still evolving
- W3C recommendations part of CSS3 standard
  - Selectors Level 3
  - Colors Level 3
  - Namespaces
  - Media Queries
- Proposals awaiting recommendation
  - Animations
  - Transforms
  - many others ...

Cross-browser compatibility

- Make use of standards
- Validate
  - W3's XHTML Validator
  - CSS Validator
- CSS Reset
  - YUI CSS reset
- Browser detection
- UI Frameworks
  - Bootstrap
  - Angular
  - React
  - Jquery
  - YUI

Browser Detection

- Includes css targeting a specific browser
- Client-side vs Server-side
- Feature detection
- Selector Hacks

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html lang="en">
	<head>
		<title>Test</title>
		<link href="all_browsers.css" rel="stylesheet" type="text/css">
		<!--[if IE]> <link href="ie_only.css" rel="stylesheet" type="text/css"> <![endif]-->
		<!--[if lt IE 7]> <link href="ie_6_and_below.css" rel="stylesheet" type="text/css"> <![endif]-->
		<!--[if !lt IE 7]><![IGNORE[--><![IGNORE[]]> <link href="recent.css" rel="stylesheet" type="text/css"> <!--<![endif]-->
		<!--[if !IE]>--> <link href="not_ie.css" rel="stylesheet" type="text/css"> <!--<![endif]-->
	</head>
	<body>
		<p>Test</p>
	</body>
</html>
```

```
#header {margin: 3em;}                    // for IE
html>body #header {margin: 1em;}          // for all other browsers
```

CSS Hacks

- Tricks for older browsers or older versions of modern browsers
- Not applicable or latest versions as they conform more to standards
- Examples
  - Box Model Hack for width and spacing issues
  - Wrappers for layout centering
  - Others

Feature Detection

- Javascript-based
  - Libraries like Modernizr help detect browser features
- CSS-based
  - Feature queries in CSS-3

```
header {
    display: block;
}

@supports (display:flexbox) {
    header {
        display: flexbox;
    }
}
```

CSS Reset

```
html{color:#000;background:#FFF;}

body,div,dl,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,h6,pre,code,form,fieldset,legend,input,textarea,p,
blockquote,th,td{margin:0;padding:0;}

table{border-collapse:collapse;border-spacing:0;}

fieldset,img{border:0;}

address,caption,cite,code,dfn,em,strong,th,var{font-style:normal;font-weight:normal;}

li{list-style:none;}caption,th{text-align:left;}

h1,h2,h3,h4,h5,h6{font-size:100%;font-weight:normal;}

q:before,q:after{content:'';}
abbr,acronym{border:0;font-variant:normal;}
sup{vertical-align:text-top;}sub{vertical-align:text-bottom;}
input,textarea,select{font-family:inherit;font-size:inherit;font-weight:inherit;}
input,textarea,select{*font-size:100%;}legend{color:#000;}
```

CSS Media Queries

- Used to limit the scope of stylesheet
- Allows presentation of content be tailored to a specific range of output devices without having to change the content itself
- CSS rules will apply only if the query evaluates to true
- Different styling rules for different devices
- One of the pillars of responsive web design

Media Query Structure

- Media query structure:
  - Media type: represents type of device i.e. all, print, speech and screen
  - Expression(s): query expressions used to test the presence or absence of features
  - Feature(s): individual features of the device e.g. width, height, orientation, etc

External Stylesheet example

```
<link rel="stylesheet" media="(max-width: 800px)" href="example.css" />
```

Internal Stylesheet example

```
<style>
@media (max-width: 600px and orientation: portrait) {
  .facet_sidebar {
    display: none;
  }
}
</style>
```