

Page Sections and the CSS Box Model

The HTML ID attribute

Allows you to give a unique ID to any element on a page

Each ID must be unique; can only be used once in the page

```
<p>Spatula City! Spatula City!</p> <p id="mission">Our mission is  
to provide the most spectacular spatulas and splurge on our specials  
until our customers <q>explode</q> with splendor!</p>
```

HTML

Spatula City! Spatula City!
Our mission is to provide the most spectacular spatulas
and splurge on our specials until our customers explode
with splendor!

output

In Page Navigation

A link target can include an ID at the end, precede by a #

Browser will load that page and scroll to element with given ID

```
<p>Visit <a href=
"http://www.textpad.com/download/index.html#downloads">
textpad.com</a> to get the TextPad editor.</p> <p><a
href="#mission">View our Mission Statement</a></p>
```

HTML

Visit [textpad.com](http://www.textpad.com/download/index.html) to get the TextPad editor.
[View our Mission Statement](#)

output

CSS ID Selector

Applies style only to the paragraph that has the ID of mission

Element can be specified explicitly `p#mission { ... }`

```
#mission {  
  font-style: italic;  
  font-family: "Garamond", "Century Gothic", serif;  
}
```

HTML

Spatula City! Spatula City!

Our mission is to provide the most spectacular spatulas and splurge on our specials until our customers explode with splendor!

output

CSS Class Selector

Classes are a way to group some elements and give a style to only that group

Unlike an **id**, a **class** can be reused as much as you like on the page

```
<p class="shout">Spatula City! Spatula City!</p>  
<p class="special">See our spectacular spatula specials!</p>  
<p class="special">Today only: satisfaction guaranteed.</p>
```

HTML

Spatula City! Spatula City!
See our spectacular spatula specials!
Today only: satisfaction guaranteed.

output

CSS Class Selector

Applies corresponding rule to any element with class `special` or a `p` with class `shout`

```
.special {  
  background-color: yellow;  
  font-weight: bold;  
}  
p.shout {  
  color: red;  
  font-family: cursive;  
}
```

CSS

Spatula City! Spatula City!

See our spectacular spatula specials!

Today only: satisfaction guaranteed.

output

Multiple Classes

```
<h2 class="shout">Spatula City!  Spatula City!</h2>  
<p class="special">See our spectacular spatula specials!</p>  
<p class="special shout">Satisfaction guaranteed.</p>  
<p class="shout">We'll beat any advertised price!</p>
```

HTML

Spatula City! Spatula City!

See our spectacular spatula specials!

Satisfaction guaranteed.

We'll beat any advertised price!

output

Grouped Selectors

To group selectors, separate each selector with a comma.

```
h1, h2, p {  
  text-align: center;  
  color: red;  
}
```

Apply styles to h1, h2 and p together

Pseudo Classes

A pseudo-class is used to define a special state of an element.

```
/* unvisited link */
a:link {
  color: #FF0000;
}
/* visited link */
a:visited {
  color: #00FF00;
}
/* mouse over link */
a:hover {
  color: #FF00FF;
}
/* selected link */
a:active {
  color: #0000FF;
}
```

Class	Description
:active	an activated or selected element
:focus	an element that has the keyboard focus
:hover	an element that has the mouse over it
:link	a link that has not been visited
:visited	a link that has already been visited
:first-letter	the first letter of text inside an element
:first-line	the first line of text inside an element
:first-child	an element that is the first one to appear inside another

Pseudo Elements

Selector	Example	Example description
<u>::after</u>	p::after	Insert content after every <p> element
<u>::before</u>	p::before	Insert content before every <p> element
<u>::first-letter</u>	p::first-letter	Selects the first letter of every <p> element
<u>::first-line</u>	p::first-line	Selects the first line of every <p> element
<u>::marker</u>	::marker	Selects the markers of list items
<u>::selection</u>	p::selection	Selects the portion of an element that is selected by a user

CSS Attribute Selectors

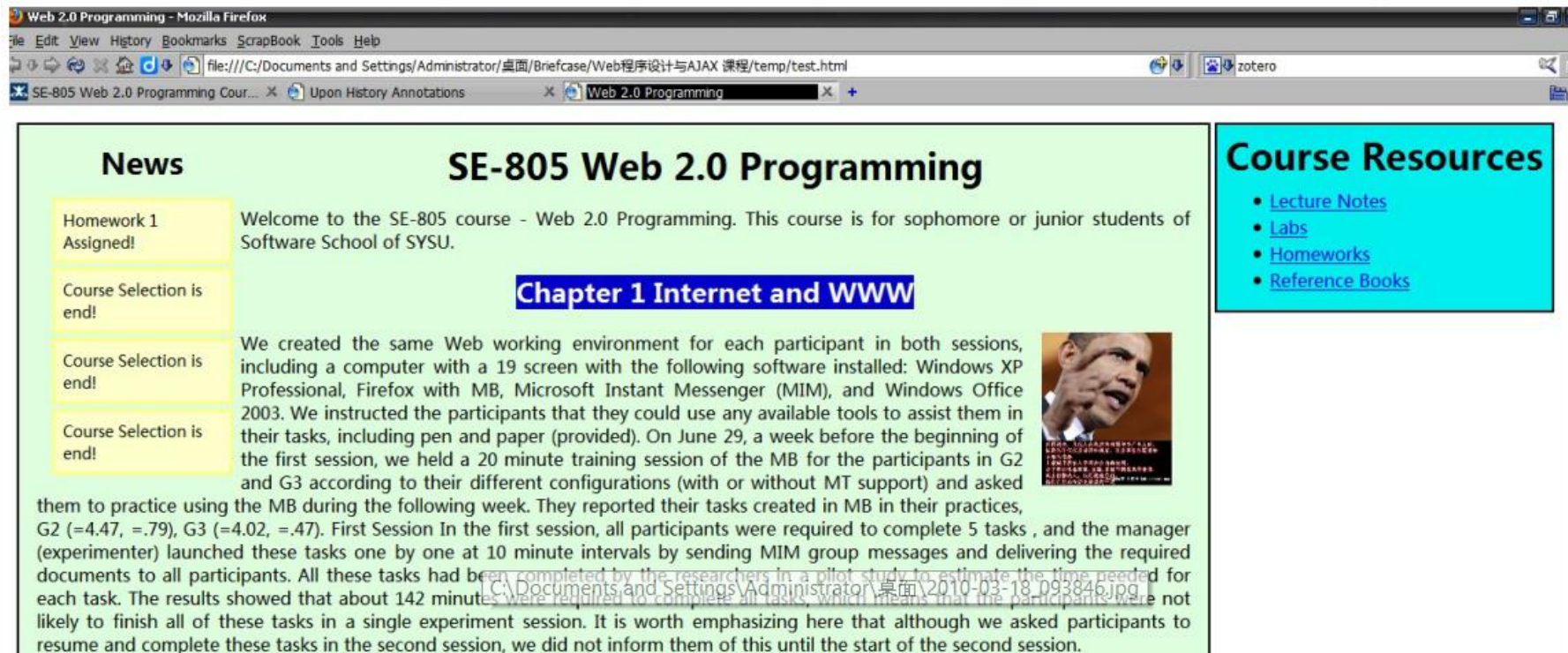
```
a[target="_blank"] {  
    background-color: yellow;  
}  
  
[class^="top"] {  
    background: yellow;  
}
```

```
input[type="text"] {  
    width: 150px;  
    display: block;  
    margin-bottom: 10px;  
    background-color: yellow;  
}
```

```
input[type="button"] {  
    width: 120px;  
    margin-left: 35px;  
    display: block;  
}
```

Page Segmentation

style individual elements, groups of elements, and sections of text of the page differently
create complex page layouts



Sections of a Page: <div>

A tag used to indicate a logical section or area of a page

Has no appearance by default, but you can apply styles to it

```
<div class="shout">  
  <h2>Spatula City!  Spatula City!</h2>  
  <p class="special">See our spectacular spatula specials!</p>  
  <p>We'll beat any advertised price!</p>  
</div>
```

HTML

Spatula City! Spatula City!

See our spectacular spatula specials!

We'll beat any advertised price!

output

Inline Section:

Has no onscreen appearance, but you can apply a style or ID to it, which will be applied to the text inside the span

So, when should we use <div>, , and when <p>, <h1>, etc

```
<h2>Spatula City!  Spatula City!</h2>
<p>See our <span class="special">spectacular</span> spatula specials!</p>
<p>We'll beat <span class="shout">any advertised price</span>!</p> HTML
```

Spatula City! Spatula City!
See our **spectacular** spatula specials!
We'll beat **any advertised price!**

output

CSS Context Selectors or Combinators

Applies the given properties to selector2 only if it is inside a selector1 on the page

```
selector1 selector2 {  
  properties  
}
```

CSS

Applies the given properties to selector2 only if it is directly inside a selector1 on the page (selector2 tag is immediately inside selector1 with no tags in between)

```
selector1 > selector2 {  
  properties  
}
```

CSS

li strong

```
<p>Shop at <strong>Hardwick's Hardware</strong>...</p>
<ul>
  <li>The <strong>best</strong> prices in town!</li>
  <li>Act while supplies last!</li>
</ul>
```

HTML

```
li strong { text-decoration: underline; }
```

CSS

Shop at **Hardwick's Hardware...**

- The **best** prices in town!
- Act while supplies last!

output

div > p

```
<div>
```

```
<p>Paragraph 1 in the div.</p>
```

```
<p>Paragraph 2 in the div.</p>
```

```
<section>
```

```
<!-- not Child but Descendant -->
```

```
<p>Paragraph 3 in the div (inside a section element).</p>
```

```
</section>
```

```
<p>Paragraph 4 in the div.</p>
```

```
</div>
```

```
<p>Paragraph 5. Not in a div.</p>
```

```
<p>Paragraph 6. Not in a div.</p>
```

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div (inside a section element).

Paragraph 4 in the div.

Paragraph 5. Not in a div.

Paragraph 6. Not in a div.

More Complex Example

```
<div id="ad">
  <p>Shop at <strong>Hardwick's Hardware</strong>...</p>
  <ul>
    <li class="important">The <strong>best</strong>
    prices in town!</li>
    <li>Act <strong>while supplies last!</strong></li>
  </ul>
</div>
```

HTML

```
#ad li.important strong { text-decoration: underline; }
```

CSS

Shop at **Hardwick's Hardware...**

- The **best** prices in town!
- Act **while supplies last!**

output

CSS Cascade

The browser's style sheet is the weakest.

The user's style sheet takes precedence over the browser's style sheet.

The author's style sheet is the strongest and takes precedence over the user's and the browser's style sheets.

The (X)HTML style attribute is more important than styles defined in any style sheet.

Within a style sheet, when conflict occurs, the most specific rule wins.

Specificity (Priority)

Inline style > # > . > p

```
<html>
<head>
  <style>
    #demo {color: blue;}
    .test {color: green;}
    p {color: red;}
  </style>
</head>
<body>
```

```
<p id="demo" class="test" style="color: pink;">Hello World!</p>
```

```
</body>
</html>
```

Specificity Hierarchy

Inline styles - Example: `<h1 style="color: pink;">`

IDs - Example: `#navbar`

Classes, pseudo-classes, attribute selectors - Example: `.test`, `:hover`, `[href]`

Elements and pseudo-elements - Example: `h1`, `::before`

Selector	Specificity Value	Calculation
p	1	1
p.test	11	1 + 10
p#demo	101	1 + 100
<p style="color: pink;">	1000	1000
#demo	100	100
.test	10	10
p.test1.test2	21	1 + 10 + 10
#navbar p#demo	201	100 + 1 + 100
*	0	0 (the universal selector is ignored)

!important

```
#myid {  
  background-color: blue;  
}
```

```
.myclass {  
  background-color: gray;  
}
```

```
p {  
  background-color: red !important;  
}
```

it will override ALL previous styling rules for that specific property on that element!

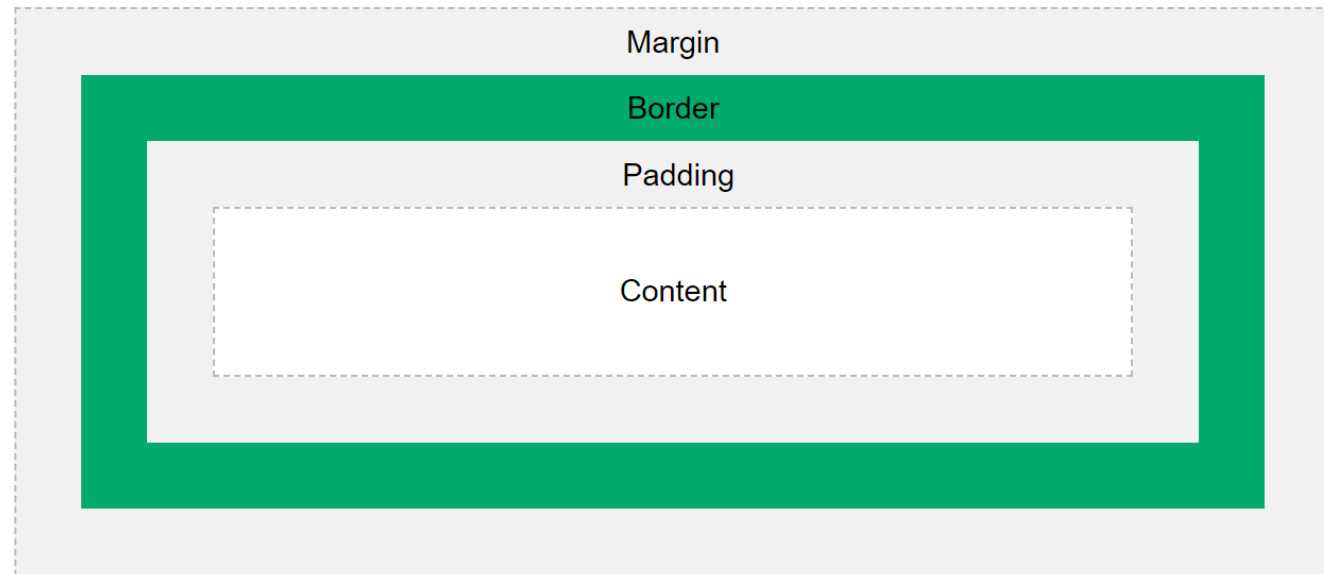
The CSS Box Model

For layout purpose, every element is composed of:

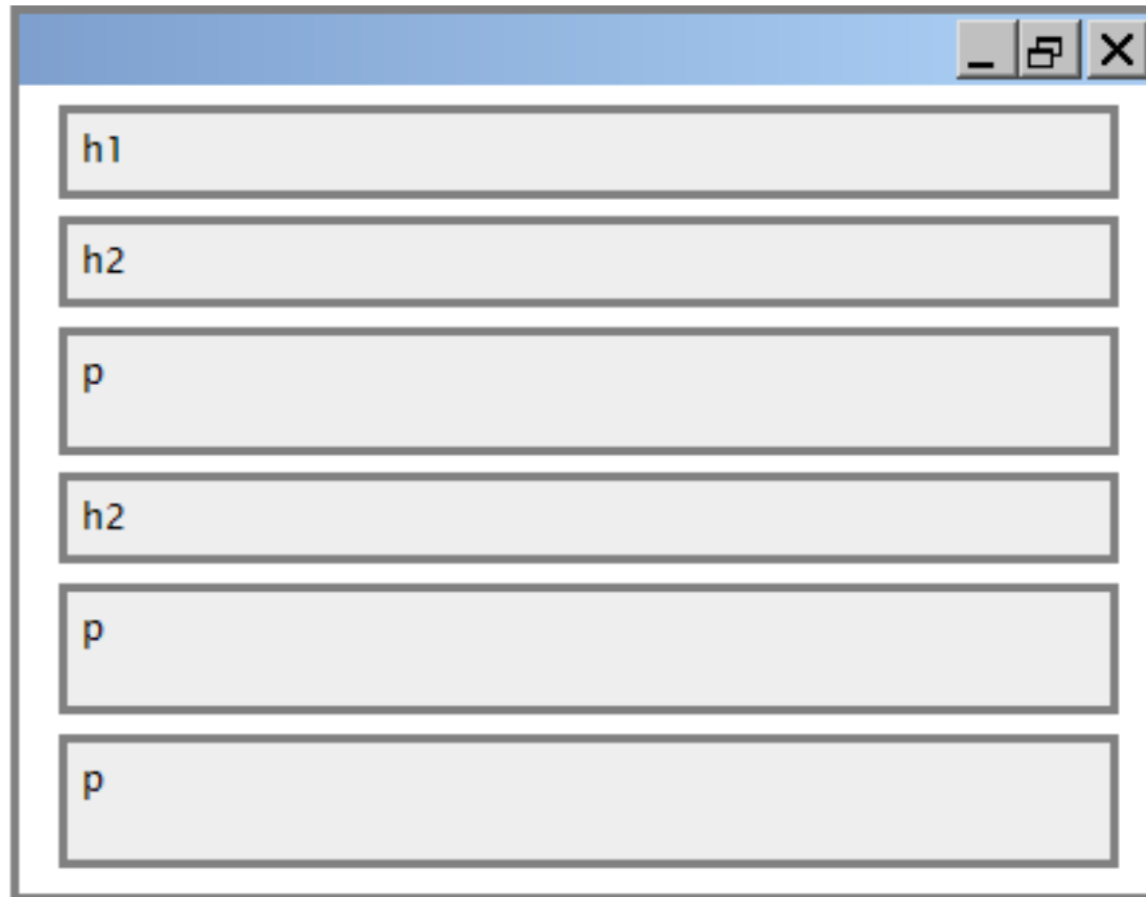
- The actual element's content
- A border around the element
- A padding between the content and the border (inside)
- A margin between the border and other

content (outside)

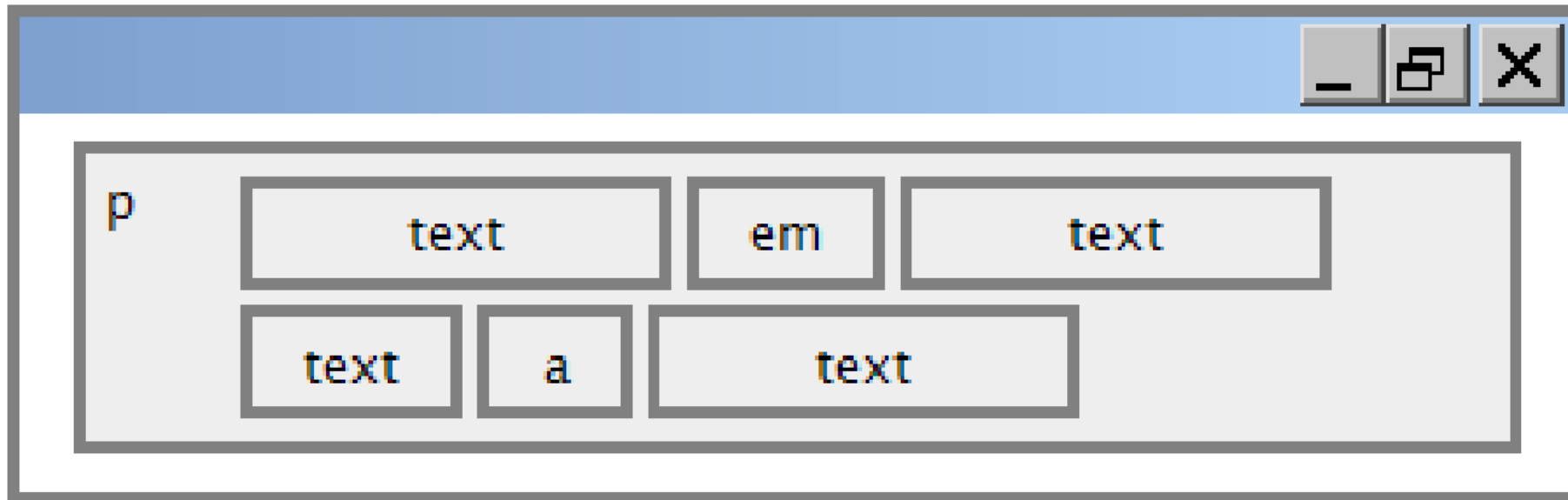
- $\text{width} = \text{content width} + \text{L/R padding} + \text{L/R margin}$
- $\text{height} = \text{content height} + \text{T/B padding} + \text{T/B margin}$



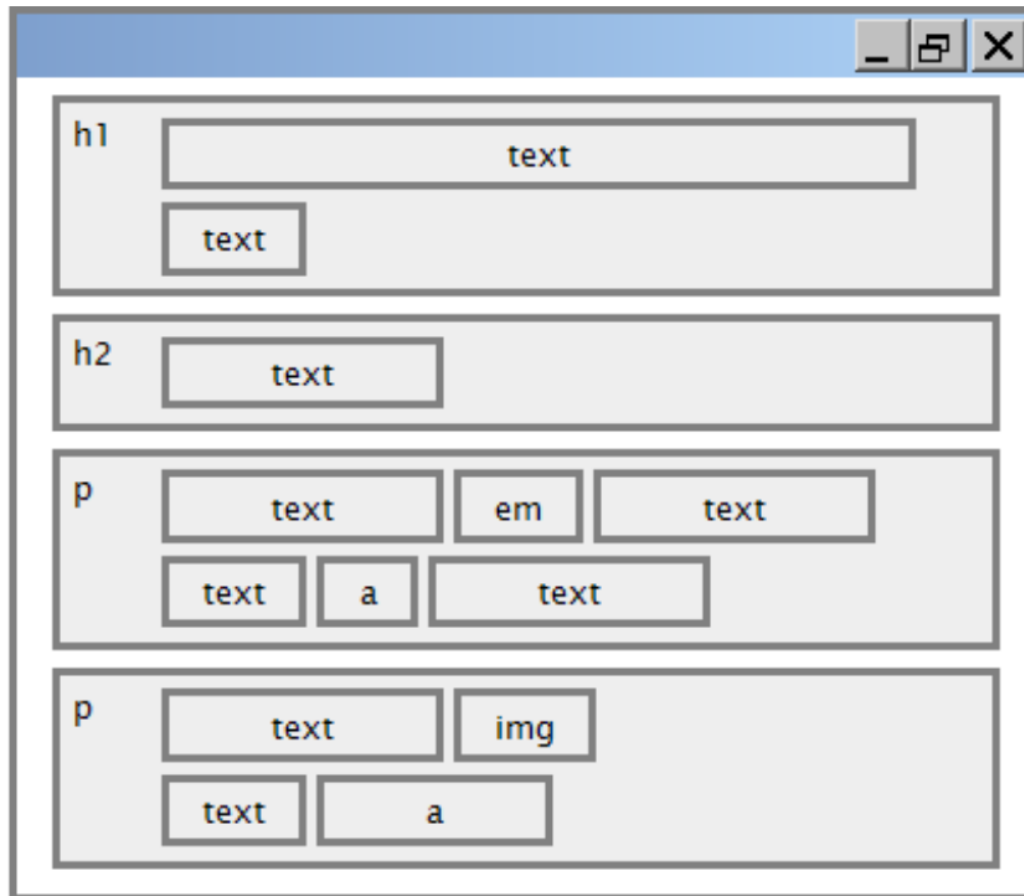
Document Flow (Block Elements)



Document Flow (Inline Elements)



Document Flow (Large Example)



Border

```
p {  
  border: 5px solid red;  
}
```

border-width

border-style (required)

border-color

```
p {  
  border-top-style: dotted;  
  border-right-style: solid;  
  border-bottom-style: dotted;  
  border-left-style: solid;  
}
```

I have borders on all sides.

I have a red bottom border.

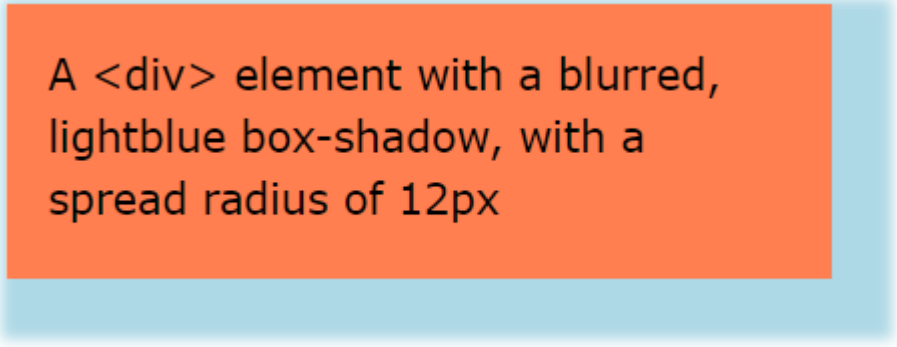
I have rounded borders.

I have a blue left border.

Box Shadow

```
div {  
  box-shadow: 10px 10px 5px 12px lightblue;  
}
```

Horizontal Vertical Blur Spread Color



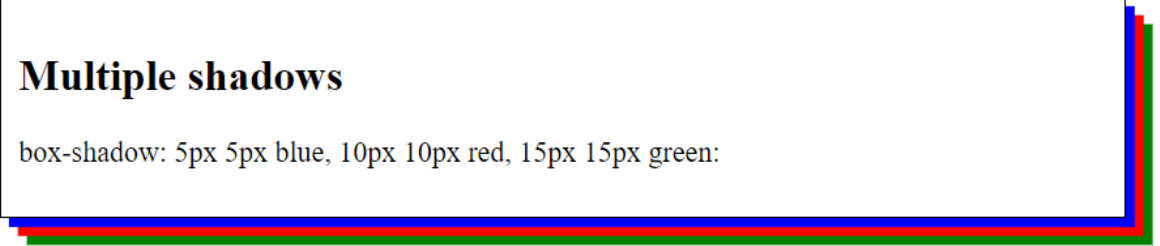
A <div> element with a blurred, lightblue box-shadow, with a spread radius of 12px

Box Shadow

```
#example1 {  
  border: 1px solid;  
  padding: 10px;  
  box-shadow: 5px 5px blue, 10px 10px red, 15px 15px green;  
  margin: 20px;  
}
```

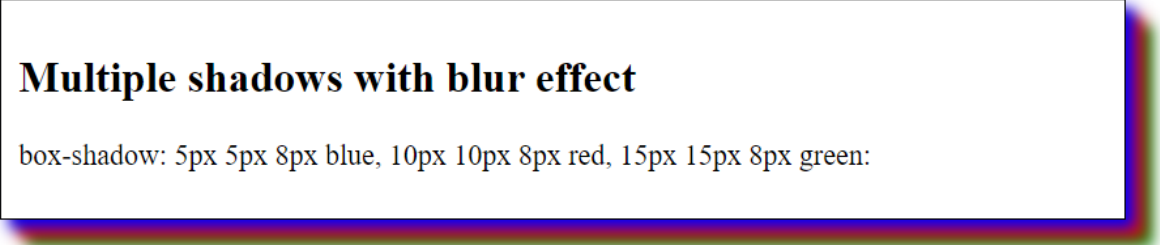
Multiple shadows

box-shadow: 5px 5px blue, 10px 10px red, 15px 15px green:



Multiple shadows with blur effect

box-shadow: 5px 5px 8px blue, 10px 10px 8px red, 15px 15px 8px green:



Polaroid Card with shadows

```
div.polaroid {  
  width: 250px;  
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);  
  text-align: center;  
}  
div.container {  
  padding: 10px;  
}
```



Hardanger, Norway

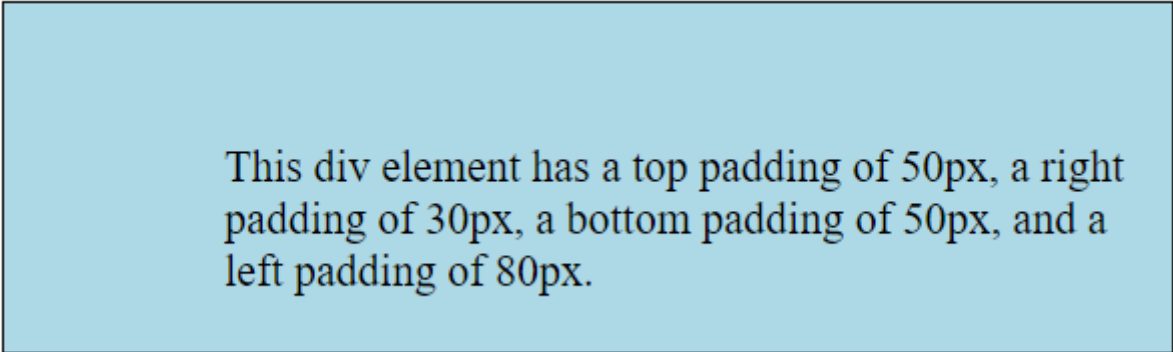
Padding

```
div {  
  border: 1px solid black;  
  background-color: lightblue;  
  padding: 15px;  
}
```

This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.

Padding

```
div {  
  border: 1px solid black;  
  background-color: lightblue;  
  padding-top: 50px;  
  padding-right: 10px;  
  padding-bottom: 20px;  
  padding-left: 80px;  
}
```



This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.

Margin

```
div {  
  border: 1px solid black;  
  margin: 20px;  
  background-color: lightblue;  
}
```

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

Margin

```
div {  
  border: 1px solid black;  
  margin-top: 10px;  
  margin-bottom: 20px;  
  margin-right: 5px;  
  margin-left: 80px;  
  background-color: lightblue;  
}
```

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

Short hand margins/paddings

If the margin property has four values:

`margin: 25px 50px 75px 100px;`

- top margin is 25px
- right margin is 50px
- bottom margin is 75px
- left margin is 100px

Dimensions

```
p { width: 350px; background-color: yellow; }  
h2 { width: 50%; background-color: aqua; }
```

CSS

This paragraph uses the first style
above.

An h2 heading

output