Motivation

• HTML markup can be used to represent
  – **Semantics**: h1 means that an element is a top-level heading
  – **Presentation**: h1 elements look a certain way
• It’s advisable to separate semantics from presentation because:
  – It’s easier to present documents on multiple platforms (browser, cell phone, spoken, …)
  – It’s easier to generate documents with consistent look
  – Semantic and presentation changes can be made independently of one another (**division of labor**)
  – **User control** of presentation is facilitated
Style Sheet Languages

• Cascading Style Sheets (CSS)
  – Applies to (X)HTML as well as XML documents in general
  – Focus of this chapter

• Extensible Stylesheet Language (XSL)
  – Often used to transform one XML document to another form, but can also add style
  – XSL Transformations covered in later chapter
CSS Introduction

• A styled HTML document

produced by the style sheet style1.css:

```css
body { background-color: lime }
p { font-size: x-large; background-color: yellow }
```
CSS Introduction

```html
<!DOCTYPE html>
  PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>CSSHelloWorld.html</title>
    <link rel="stylesheet" type="text/css" href="style1.css" title="Style 1" />
    <link rel="alternate stylesheet" type="text/css" href="style2.css" title="Style 2" />
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

The link element associates style sheet with doc.
CSS Introduction

<!DOCTYPE html>
  PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>CSSHelloWorld.html</title>
    <link rel="stylesheet" type="text/css" href="style1.css" title="Style 1" />
    <link rel="alternate stylesheet" type="text/css" href="style2.css" title="Style 2" />
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>

The type attribute specifies the style language used.
CSS Introduction

<!DOCTYPE html>
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>CSSHelloWorld.html</title>
<link rel="stylesheet" type="text/css" href="style1.css" title="Style 1" />
<link rel="alternate stylesheet" type="text/css" href="style2.css" title="Style 2" />
</head>
<body>
<p>Hello World!</p>
</body>
</html>

href attribute provides style sheet URL
CSS Introduction

```html
<!DOCTYPE html>
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>CSSHelloWorld.html</title>
    <link rel="stylesheet" type="text/css" href="style1.css" title="Style 1" />
    <link rel="alternate stylesheet" type="text/css" href="style2.css" title="Style 2" />
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

title attribute provides style sheet name
CSS Introduction
CSS Introduction

```html
<!DOCTYPE html>
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>
    CSSHelloWorld.html
  </title>
  <link rel="stylesheet" type="text/css" href="style1.css"
  title="Style 1" />
  <link rel="alternate stylesheet" type="text/css" href="style2.css"
  title="Style 2" />
</head>
<body>
  <p>Hello World!
  </p>
</body>
</html>
```

Alternative, user selectable style sheets can be specified
CSS Introduction

• A styled HTML document

produced by the style sheet `style2.css`:

```html
    p { font-size: smaller; letter-spacing: 1em }
```
CSS Introduction

- Single document can be displayed on multiple media platforms by tailoring style sheets:

  ```html
  <link rel="stylesheet" type="text/css" href="style1.css" media="screen, tv, projection" />
  <link rel="stylesheet" type="text/css" href="style2.css" media="handheld, print" />
  ```

This document will be **printed** differently than it is **displayed**.
CSS Syntax

• Parts of a style rule (or statement)
CSS Syntax: Selector Strings

• Single element type:

\[
p \ { \text{font-size:smaller; letter-spacing:1em } }\]

• Multiple element types:

\[
h1,h2,h3,h4,h5,h6 \ { \text{background-color:purple } }\]

• All element types:

\[
* \ { \text{font-weight:bold } }\]

• Specific elements by id:

\[
#p1, #p3 \ { \text{background-color:aqua } }\]
CSS Syntax: Selector Strings

• Single element type:

```css
p { font-size: smaller; letter-spacing: 1em }
```

• Multiple element types:

```css
h1, h2, h3, h4, h5, h6 { background-color: purple }
```

• All element types:

```css
* { font-weight: bold }
```

• Specific elements by id:

```css
#p1, #p3 { background-color: aqua }
```
CSS Syntax: Selector Strings

• Single element type:

```css
p { font-size: smaller; letter-spacing: 1em }
```

• Multiple element types:

```css
h1, h2, h3, h4, h5, h6 { background-color: purple }
```

• All element types:

```css
* { font-weight: bold }

universal selector
```

• Specific elements by id:

```css
#p1, #p3 { background-color: aqua }
```
CSS Syntax: Selector Strings

• Single element type:
  
  p     { font-size: smaller; letter-spacing: 1em }

• Multiple element types:
  
  h1, h2, h3, h4, h5, h6  { background-color: purple }

• All element types:
  
  *  { font-weight: bold }

• Specific elements by id:
  
  #p1, #p3  { background-color: aqua }
CSS Syntax: Selector Strings

• Elements belonging to a style class:

\[
#p4, .takeNote \{ \text{font-style:italic} \}
\]

class selector

– Referencing a style class in HTML:

\[
<\text{span class="takeNote special cool"}>
\]

• Elements of a certain type and class:

\[
\text{span.special} \{ \text{font-size:x-large} \}
\]
CSS Syntax: Selector Strings

• Elements belonging to a **style class**:

```html
#p4, .takeNote { font-style: italic }
```

– Referencing a style class in HTML:

```html
<span class="takeNote special cool">this span belongs to three style classes</span>
```

• Elements of a certain type and class:

```html
span.special { font-size:x-large }
```
CSS Syntax: Selector Strings

• Elements belonging to a style class:
  
  \#p4, .takeNote { font-style:italic }

  – Referencing a style class in HTML:

  `<span class="takeNote special cool">`

• Elements of a certain type and class:

  `span.special { font-size:x-large }`

  this rule applies only to span’s belonging to class special.
CSS Syntax:
Selector Strings

• Source anchor elements:

```css
a:link { color:black }
a:visited { color:yellow }
a:hover { color:green }
a:active { color:red }
```

• Element types that are descendents:

```css
ul ol li { letter-spacing:1em }
```
CSS Syntax: Selector Strings

• Source anchor elements:

  a:link { color:black }
a:visited { color:yellow }
a:hover { color:green }
a:active { color:red }

• Element types that are **descendants**:

  ul ol li { letter-spacing:1em }
  rule applies to li element that is
CSS Syntax: Selector Strings

• Source anchor elements:

```css
a:link { color:black }
a:visited { color:yellow }
a:hover { color:green }
a:active { color:red }
```

• Element types that are **descendants**:

```css
ul ol li { letter-spacing:1em }
```

rule applies to `li` element that is part of the content of an `ol` element
CSS Syntax: Selector Strings

• Source anchor elements:

```css
a:link { color:black }
a:visited { color:yellow }
a:hover { color:green }
a:active { color:red }
```

• Element types that are **descendants**:

```css
ul ol li { letter-spacing:1em }
```

rule applies to a li element that is part of the content of an ol element that is part of the content of a ul element
CSS Syntax

• Style rules covered thus far follow ruleset syntax

• At-rule is a second type of rule
  
  @import url("general-rules.css");

  – Reads style rules from specified URL
  – Must appear at beginning of style sheet
Style Sheets and HTML

- Style sheets referenced by `link` HTML element are called **external** style sheets.
- Style sheets can be **embedded** directly in HTML document using `style` element.

```html
<head>
  <title>InternalStyleSheet.html</title>
  <style type="text/css">
    h1, h2 { background-color:aqua }
  </style>
</head>
```

- Most HTML elements have `style` attribute (value is list of style declarations).
Style Sheets and HTML

• Rules of thumb:
  – Use external style sheets to define site-wide style
  – Prefer style sheets (either external or embedded) to style attributes
  – XML special characters
    • Must use references in embedded style sheets and style attribute
    • Must not use references in external style sheets
CSS Rule Cascade

• What if more than one style declaration applies to a property of an element?

```css
* { font-weight: bold }
#p3 { font-weight: normal }
```

• The CSS **rule cascade** determines which style rule’s declaration applies
CSS Rule Cascade

1. Select style sheets and insert rules for HTML attributes

2. Prioritize declarations by origin and weight

3. Break ties based on specificity (style attribute or most specific selector)

4. Break ties based on position within style sheet (last occurring wins)
CSS Rule Cascade

Select appropriate style sheets based on user selection and media type.
CSS Rule Cascade

Treat HTML attributes such as width and height of img as if defined by style rule instead.
CSS Rule Cascade

Five origin/weight levels:
1. user/important
2. author/important
3. author/normal
4. user/normal
5. user agent/normal
CSS Rule Cascade

• User can define a style sheet
  – Explicitly (easy in IE)
  – Implicitly (preferences)
• User/important highest priority in CSS2 to accommodate users with special needs
  – Rules made important by adding “!important”: 
    ```
p { text-indent:3em; font-size:larger !important }
```
CSS Rule Cascade

Specificity:
1. style attribute
2. rule with selector:
   1. 
   2. 
   3. 
   4. Break ties based on position within style sheet (last occurring wins)
CSS Rule Cascade

Conceptually, create one long style sheet. Later style rules have higher priority than earlier rules.
CSS Inheritance

• What if no style declaration applies to a property of an element?
• Generally, the property value is inherited from the nearest ancestor element that has a value for the property
• If no ancestor has a value (or the property does not inherit) then CSS defines an initial value that is used
CSS Inheritance

```html
<body>
  <ul>
    <li>
      List item outside and <span>inside</span> a span.
      <p>
        Embedded paragraph outside and <span>inside</span> a span.
      </p>
    </li>
  </ul>
</body>
```
CSS Inheritance

• Most properties inherit **computed value**
  – Exception discussed later: line-height
• A little thought can usually tell you whether a property inherits or not
  – Example: height does not inherit
CSS Font Properties

- A **font** is a mapping from code points to **glyphs**

  ![Image of font mapping](image)

  - **Glyph**: The visual representation of a character.
  - **Character cell (content area)**: The area where the glyph is displayed.
CSS Font Properties

• A **font family** is a collection of related fonts (typically differ in size, weight, etc.)

```html
<p style="font-family: 'Jenkins v2.0'">
</p>
```

• font-family property can accept a list of families, including **generic** font families

```html
font-family: "Edwardian Script ITC", "French Script MT", cursive, generic
```
CSS Font Properties

![CSS Font Properties](image)

generic fonts are system-specific
CSS Font Properties

• Many properties, such as font-size, have a value that is a **CSS length**
• All CSS length values except 0 need units

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>in</td>
<td>inches</td>
</tr>
<tr>
<td>cm</td>
<td>centimeters</td>
</tr>
<tr>
<td>mm</td>
<td>millimeters</td>
</tr>
<tr>
<td>pt</td>
<td>points: 1/72-inch</td>
</tr>
<tr>
<td>pc</td>
<td>picas: 12 points</td>
</tr>
<tr>
<td>px</td>
<td>pixel: typically 1/96-inch (see text).</td>
</tr>
<tr>
<td>em</td>
<td>1em is roughly the height of a capital letter in the reference font (see text).</td>
</tr>
<tr>
<td>ex</td>
<td>1ex is roughly the height of the lowercase ’x’ character in the reference font (see text).</td>
</tr>
</tbody>
</table>
CSS Font Properties

Computed value of font-size property

em height

baseline height

M x

dashed rectangle

baseline

cell bottom

ex height

cell height
CSS Font Properties

- **Reference font** defines em and ex units
  - Normally, reference font is the font of the element being styled
  - Exception: Using em/ex to specify value for font-size

```html
<div id="d1" style="font-size:12pt">  
  <div id="d2" style="font-size:2em">  
  
  parent element’s font is reference font

```
CSS Font Properties

• Other ways to specify value for font-size:
  – **Percentage** (of parent font-size)
    
    ```
    font-size: 85%
    ```
  – **Absolute size** keyword: xx-small, x-small, small, medium (initial value), large, x-large, xx-large
    
    • User agent specific; should differ by ~ 20%
  – **Relative size** keyword: smaller, larger
    
    • Relative to parent element’s font
### CSS Font Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>font-style</td>
<td>normal (initial value), italic (more cursive than normal), or oblique (more slanted than normal).</td>
</tr>
<tr>
<td>font-weight</td>
<td>bold or normal (initial value) are standard values, although other values can be used with font families having multiple gradations of boldness (see CSS2 [W3C-CSS-2.0] for details).</td>
</tr>
<tr>
<td>font-variant</td>
<td>small-caps, which displays lowercase characters using uppercase glyphs (small uppercase glyphs if possible), or normal (initial value).</td>
</tr>
</tbody>
</table>
CSS Font Properties

• Text is rendered using line boxes

\[
\text{This is sure fun!}
\]

• Height of line box given by \texttt{line-height}
  – Initial value: \texttt{normal} (\textit{i.e.}, cell height; relationship with \texttt{em} height is font-specific)
  – Other values (following are equivalent):
    \begin{align*}
    \text{line-height}:1.5\text{em} \\
    \text{line-height}:150\% \\
    \text{line-height}:1.5
    \end{align*}
CSS Font Properties

- **font shortcut property:**

```css
{ font: italic bold 12pt "Helvetica",sans-serif }
```

```css
{ font-style: italic;
  font-variant: normal;
  font-weight: bold;
  font-size: 12pt;
  line-height: normal;
  font-family: "Helvetica",sans-serif }
```

Initial values used if no value specified in font property list
## CSS Text Formatting

<table>
<thead>
<tr>
<th>Property</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>text-decoration</td>
<td>none (initial value), underline, overline, line-through, or space-separated list of values other than none.</td>
</tr>
<tr>
<td>letter-spacing</td>
<td>normal (initial value) or a length representing additional space to be included between adjacent letters in words. Negative value indicates space to be removed.</td>
</tr>
<tr>
<td>word-spacing</td>
<td>normal (initial value) or a length representing additional space to be included between adjacent words. Negative value indicates space to be removed.</td>
</tr>
<tr>
<td>text-transform</td>
<td>none (initial value), capitalize (capitalizes first letter of each word), uppercase (converts all text to uppercase), lowercase (converts all text to lowercase).</td>
</tr>
<tr>
<td>text-indent</td>
<td>length (initial value 0) or percentage of box width, possibly negative. Specify for block elements and table cells to indent text within first line box.</td>
</tr>
<tr>
<td>text-align</td>
<td>left (initial value for left-to-right contexts), right, center, or justified. Specify for block elements and table cells.</td>
</tr>
<tr>
<td>white-space</td>
<td>normal (initial value), pre. Use to indicate whether or not white space should be retained.</td>
</tr>
</tbody>
</table>
CSS Text Color

• Font color specified by color property

• Two primary ways of specifying colors:
  – Color name: black, gray, silver, white, red, lime, blue, yellow, aqua, fuchsia, maroon, green, navy, olive, teal, purple, full list at http://www.w3.org/TR/SVG11/types.html#Color
  – red/green/blue (RGB) values
CSS Text Color
CSS Text Color

<table>
<thead>
<tr>
<th>Format</th>
<th>Example</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional, integer arguments</td>
<td>rgb(255,170,0)</td>
<td>Use arguments as RGB values.</td>
</tr>
<tr>
<td>Functional, percentage arguments</td>
<td>rgb(100%,66.7%,0%)</td>
<td>Multiply arguments by 255 and round to obtain RGB values (at most one decimal place allowed in arguments).</td>
</tr>
<tr>
<td>Hexadecimal</td>
<td>#ffaa00</td>
<td>The first pair of hexadecimal digits represents the red intensity, second and third represent green and blue, respectively.</td>
</tr>
<tr>
<td>Abbreviated hexadecimal</td>
<td>#fa0</td>
<td>Duplicate the first hexadecimal digit to obtain red intensity, duplicate second and third to obtain green and blue, respectively.</td>
</tr>
</tbody>
</table>
CSS Box Model

- Every rendered element occupies a box:
CSS Box Model
CSS Box Model

```css
span {
    margin-left: 1cm;
    border-left-width: 10px;
    border-left-color: silver;
    border-left-style: solid;
    padding-left: 0.5cm;
    border-right-width: 5px;
    border-right-color: silver;
    border-right-style: solid
}
```

The first span and second span.
CSS Box Model

```css
span {
  margin-left: 1cm;
  border-left-width: 10px;
  border-left-color: silver;
  border-left-style: solid;
  padding-left: 0.5cm;
  border-right-width: 5px;
  border-right-color: silver;
  border-right-style: solid;
}
```

The first span and second span.
CSS Box Model

```css
span {
  margin-left: 1cm;
  border-left-width: 10px;
  border-left-color: silver;
  border-left-style: solid;
  padding-left: 0.5cm;
  border-right-width: 5px;
  border-right-color: silver;
  border-right-style: solid
}
```

The first span and second span.
## CSS Box Model

**TABLE 3.9:** Basic CSS style properties associated with the box model.

<table>
<thead>
<tr>
<th>Property</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>padding-{top,right,bottom,left}</code></td>
<td>CSS length (Sec. 3.6.2).</td>
</tr>
<tr>
<td><code>padding</code></td>
<td>One to four length values (see text).</td>
</tr>
</tbody>
</table>

**TABLE 3.10:** Meaning of values for certain shorthand properties that take one to four values.

<table>
<thead>
<tr>
<th>Number of values</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>Assign this value to all four associated properties (<code>top</code>, <code>right</code>, <code>bottom</code>, and <code>left</code>).</td>
</tr>
<tr>
<td>Two</td>
<td>Assign first value to associated <code>top</code> and <code>bottom</code> properties, second value to associated <code>right</code> and <code>left</code> properties.</td>
</tr>
<tr>
<td>Three</td>
<td>Assign first value to associated <code>top</code> property, second value to <code>right</code> and <code>left</code>, and third value to <code>bottom</code>.</td>
</tr>
<tr>
<td>Four</td>
<td>Assign first value to associated <code>top</code> property, second to <code>right</code>, third to <code>bottom</code>, and fourth to <code>left</code>.</td>
</tr>
</tbody>
</table>
CSS Box Model

TABLE 3.9: Basic CSS style properties associated with the box model.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>border-top-width</code></td>
<td>Thin, medium (initial value), thick, or a length.</td>
</tr>
<tr>
<td><code>border-width</code></td>
<td>One to four <code>border-*width</code> values.</td>
</tr>
<tr>
<td><code>border-top-width</code></td>
<td>Color value. Initial value is value of element's <code>color</code> property.</td>
</tr>
<tr>
<td><code>border-color</code></td>
<td><code>transparent</code> or one to four <code>border-*color</code> values.</td>
</tr>
</tbody>
</table>
## CSS Box Model

### TABLE 3.9: Basic CSS style properties associated with the box model.

<table>
<thead>
<tr>
<th><code>border-{top,right,bottom,left}-style</code></th>
<th>none (initial value), hidden, dotted, dashed, solid, double, groove, ridge, inset, outset.</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>border-style</code></td>
<td>One to four <code>border-*</code>-style values.</td>
</tr>
</tbody>
</table>

---

I want my paragraphs to be noticed. Do you think that this will do it? Styles here starting at top and going clockwise are solid, dashed, dotted, and double.

Here's my second paragraph. What do you think of this one? Styles here starting at top and going clockwise are groove, inset, outset, ridge.

---

I want my paragraphs to be noticed. Do you think that this will do it? Styles here starting at top and going clockwise are solid, dashed, dotted, and double.

Here's my second paragraph. What do you think of this one? Styles here starting at top and going clockwise are groove, inset, outset, ridge.
TABLE 3.9: Basic CSS style properties associated with the box model.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>border-{top, right, bottom, left}</td>
<td>One to three values (in any order) for border-*-width, border-*-color, and border-*-style. Initial values are used for any unspecified values.</td>
</tr>
<tr>
<td>border</td>
<td>One to three values; equivalent to specifying given values for each of border-top, border-right, border-bottom, and border-left.</td>
</tr>
<tr>
<td>margin-{top, right, bottom, left}</td>
<td>auto (see text) or length.</td>
</tr>
<tr>
<td>margin</td>
<td>One to four margin-* values.</td>
</tr>
</tbody>
</table>
CSS Box Model

- If multiple declarations apply to a property, the last declaration overrides earlier specifications

```css
{ border: 15px solid; border-left: 30px inset red; color: blue }
```

Left border is 30px wide, inset style, and red
Backgrounds

- **background-color**
  - Specifies background color for content, padding, and border areas
  - Margin area is always transparent
  - Not inherited; initial value transparent

- **background-image**
  - Specifies (using `url()` function) image that will be tiled over an element
Backgrounds

This is some text written over top of an image which is a tiled background for the body.
Normal Flow Layout

- In **normal flow processing**, each displayed element has a corresponding box
  - html element box is called **initial containing block** and corresponds to entire document
  - Boxes of child elements are contained in boxes of parent
  - Sibling **block elements** are laid out one on top of the other
  - Sibling **inline elements** are one after the other
Normal Flow Layout

Paragraph 1: blah blah blah more blah a span blah blah blah blah blah blah blah blah blah

Paragraph 2: blah blah blah more blah a span blah blah blah blah blah blah blah blah blah

Paragraph 3: Image 1

Image 2

Canvas (body)
Initial Containing Block (html)
Browser Client Area
Normal Flow Layout

```html
html, body { border:solid red thin }
html { border-width:thick }
body { padding:15px }
div { margin:0px; padding:15px; border:solid black 2px }
.shade { background-color:aqua }
.topMargin { margin-top:10px }

<body>
  <div id="d1">
    <div id="d2">
      <div id="d3" class="shade"></div>
    </div>
    <div id="d4" class="shade topMargin"></div>
  </div>
</body>
```
Normal Flow Layout

Top edges of block boxes are in document order
Normal Flow Layout

• What is a “block element”?
  – Element with value block specified for its display property
  – User agent style sheet (not CSS) specifies default values; typical block elements include html, body, p, pre, div, form, ol, ul, dl, hr, h1 through h6
  – Most other elements except li and table-related have inline specified for display
Normal Flow Layout

- When blocks stack, adjacent margins are **collapsed** to the size of the larger margin.
Normal Flow Layout

• Initial value of *width* property is *auto*, which for block boxes means to make the content area as wide as possible within margin/padding constraints:

![Diagram showing normal flow layout](image_url)

Width of block boxes increases as browser client area is widened
Normal Flow Layout

- Can also specify CSS length or percentage (of parent’s content width) for width property

```css
#d3 { width:50% }
```

By default, width of right margin is adjusted to accommodate a change to width
Normal Flow Layout

• Can also specify CSS length or percentage (of parent’s content width) for width property

```css
#d4 { width: 50%; margin-left: auto; margin-right: auto }
```

Centering can be achieved by setting both margins to auto
Normal Flow Layout

- Boxes corresponding to character cells and inline elements are laid out side by side in line boxes that are stacked one on top of the other.
Normal Flow Layout

• Specify value for `vertical-align` to position an inline element within line box:
Beyond Normal Flow

• CSS allows for boxes to be positioned outside the normal flow:
  – **Absolute** positioning

![Image showing CSS absolute positioning example]

- span’s removed from normal flow and positioned relative to another box
Assignment “Style sheets”

• See the assignment description
• Use book and W3C CSS 2.1 Recommendation as background

http://www.w3.org/TR/CSS21/