

# WEB TECHNOLOGIES A COMPUTER SCIENCE PERSPECTIVE

JEFFREY C. JACKSON

## Chapter 3 Style Sheets: CSS

# Motivation

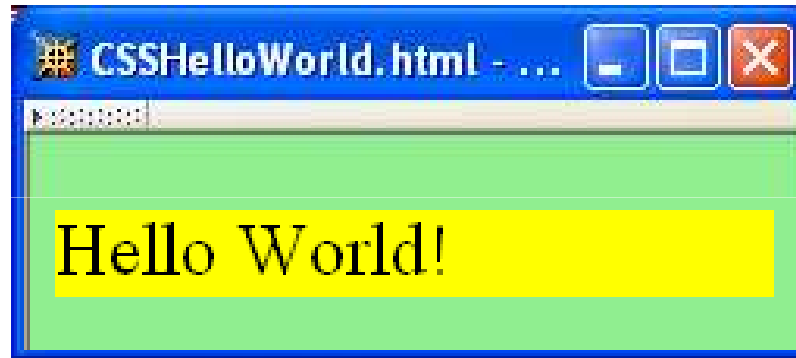
- HTML markup can be used to represent
  - **Semantics**: h1 **means** that an element is a top-level heading
  - **Presentation**: h1 elements **look** a certain way
- It's advisable to separate semantics from presentation because:
  - It's easier to present documents on **multiple platforms** (browser, 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`:

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

# 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>
```

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>
```

type attribute specifies 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

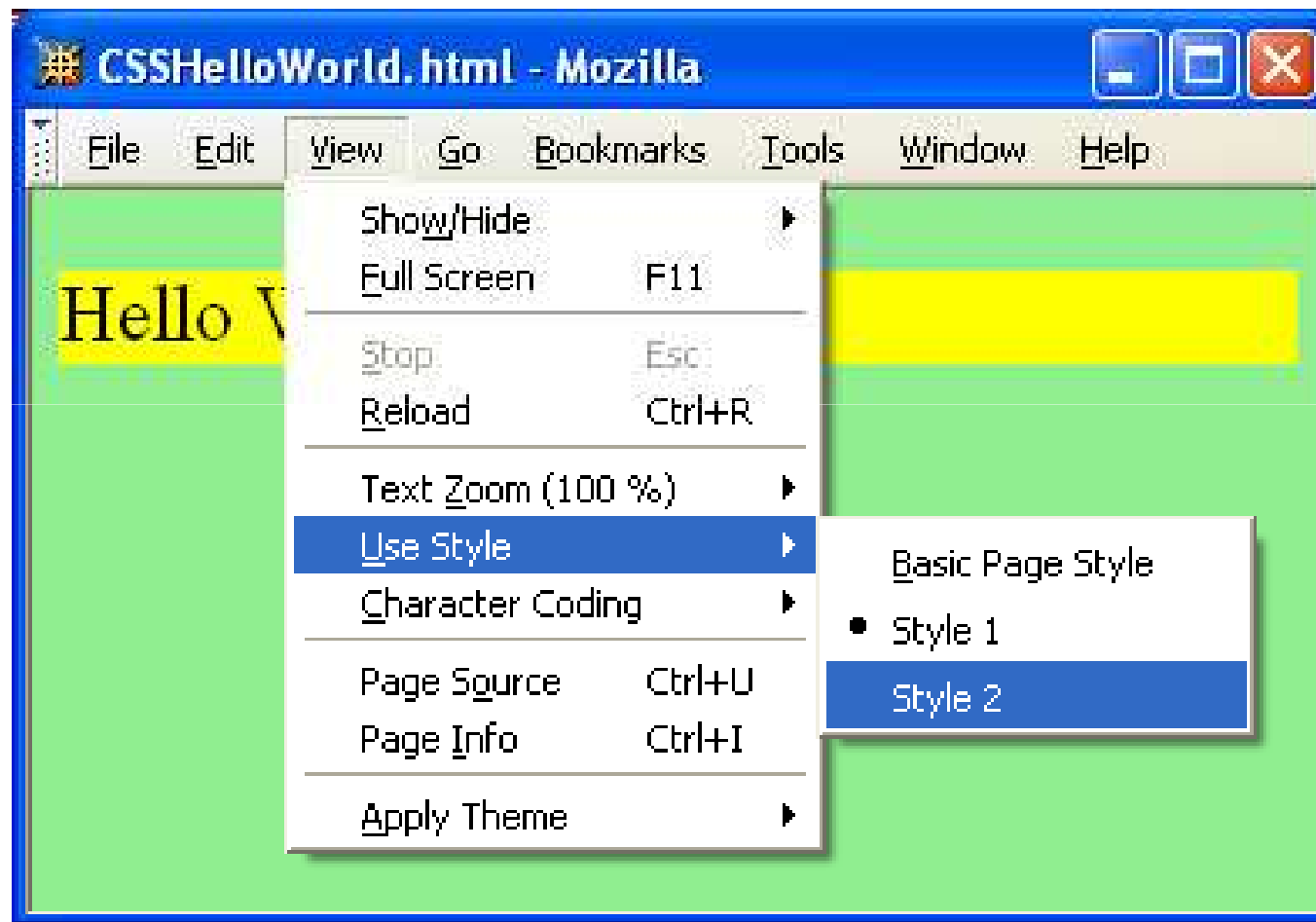
# CSS Introduction

```
<!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



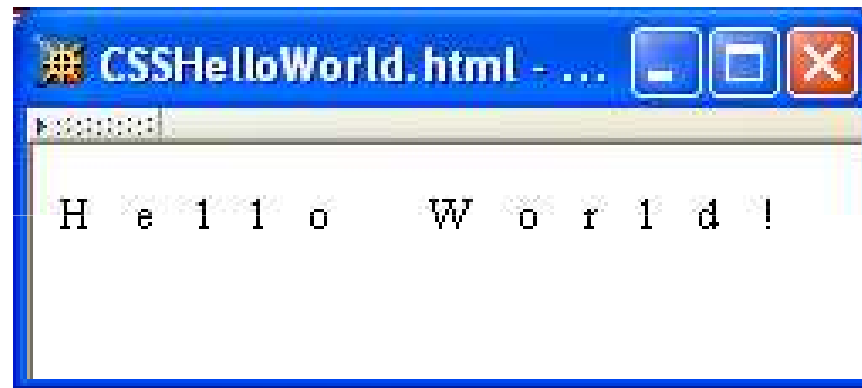
# CSS Introduction

```
<!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`:

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

# CSS Introduction

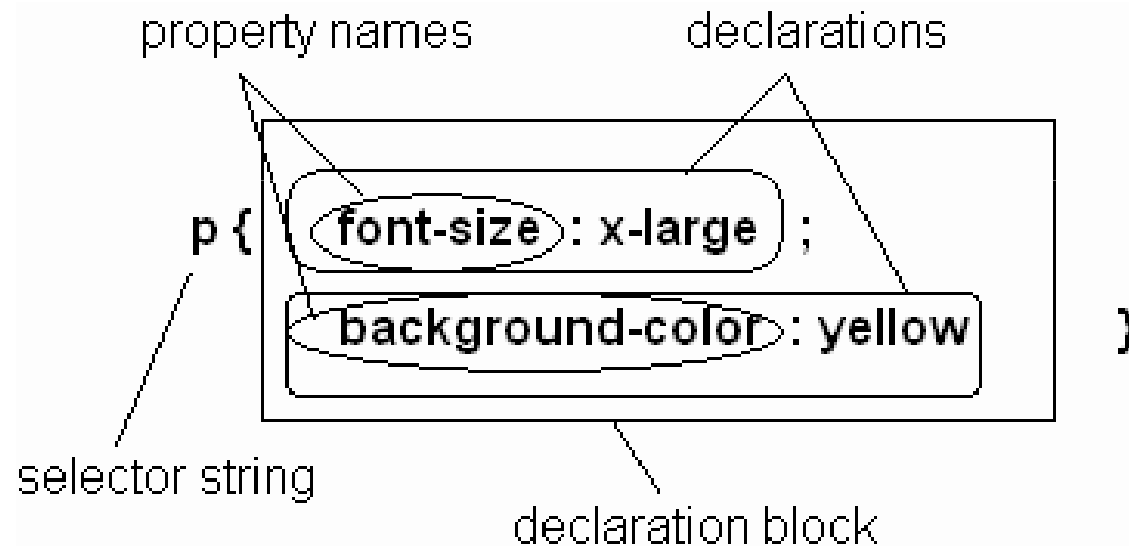
- Single document can be displayed on multiple media platforms by tailoring style sheets:

```
<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 { 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

- Single element type:

 { font-size:smaller; letter-spacing:1em }  
type selector

- Multiple element types:

 ,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


- 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 }  
universal selector
```

- Specific elements by id:

```
#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:

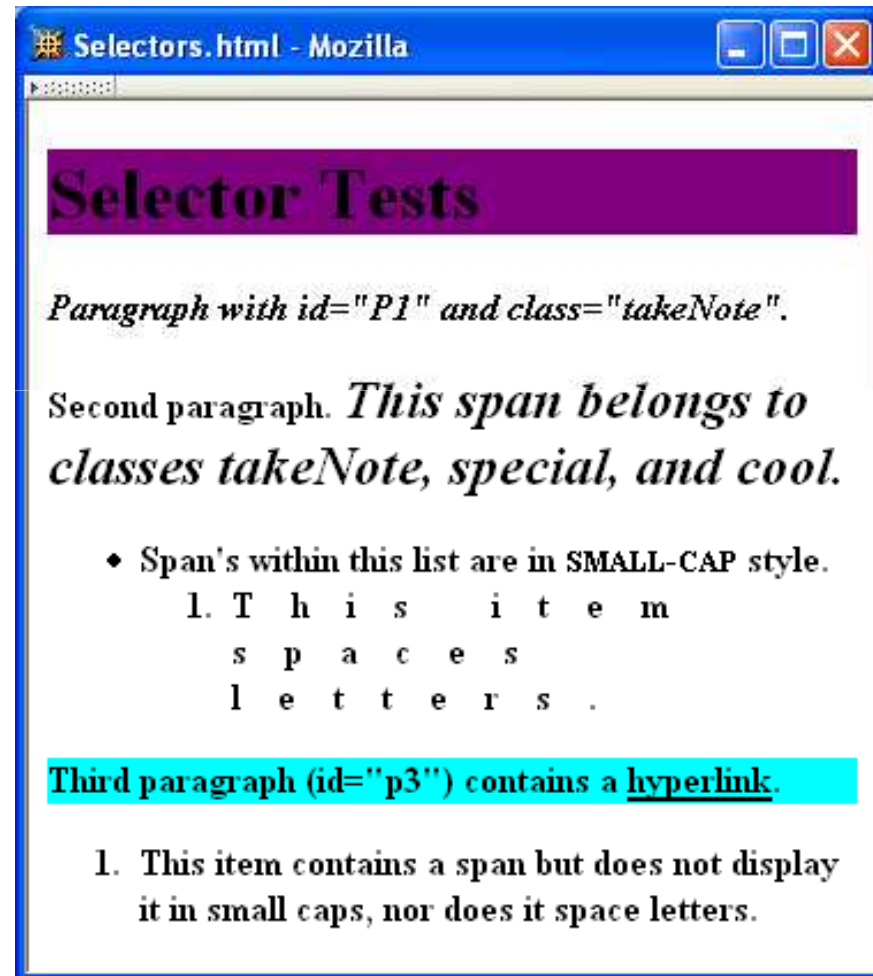
```
#p1, #p3 { background-color:aqua }
```

ID selector

# CSS Syntax: Selector Strings

```
<p id="p3">
```

Third paragraph (id="p3")



# CSS Syntax: Selector Strings

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

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

class selector

- Referencing a style class in HTML:

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

- Elements of a certain type and class:

```
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">
```

this span belongs to three style classes

- Elements of a certain type and class:

```
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:

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

pseudo-classes

- Element types that are descendants:

```
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:

```
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  
part of the content of an `ol` element

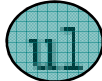


# CSS Syntax: Selector Strings

- 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  
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

URL relative to style sheet URL

```
@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

```
<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

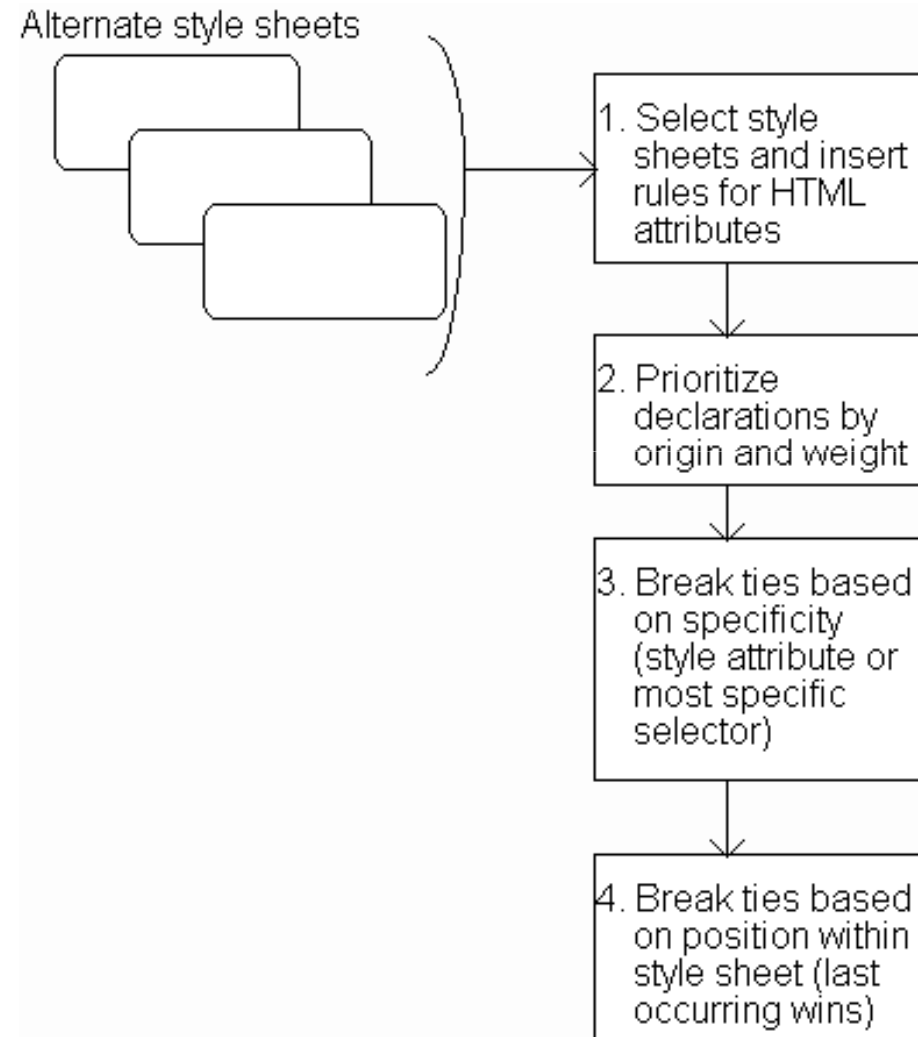
# CSS Rule Cascade

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

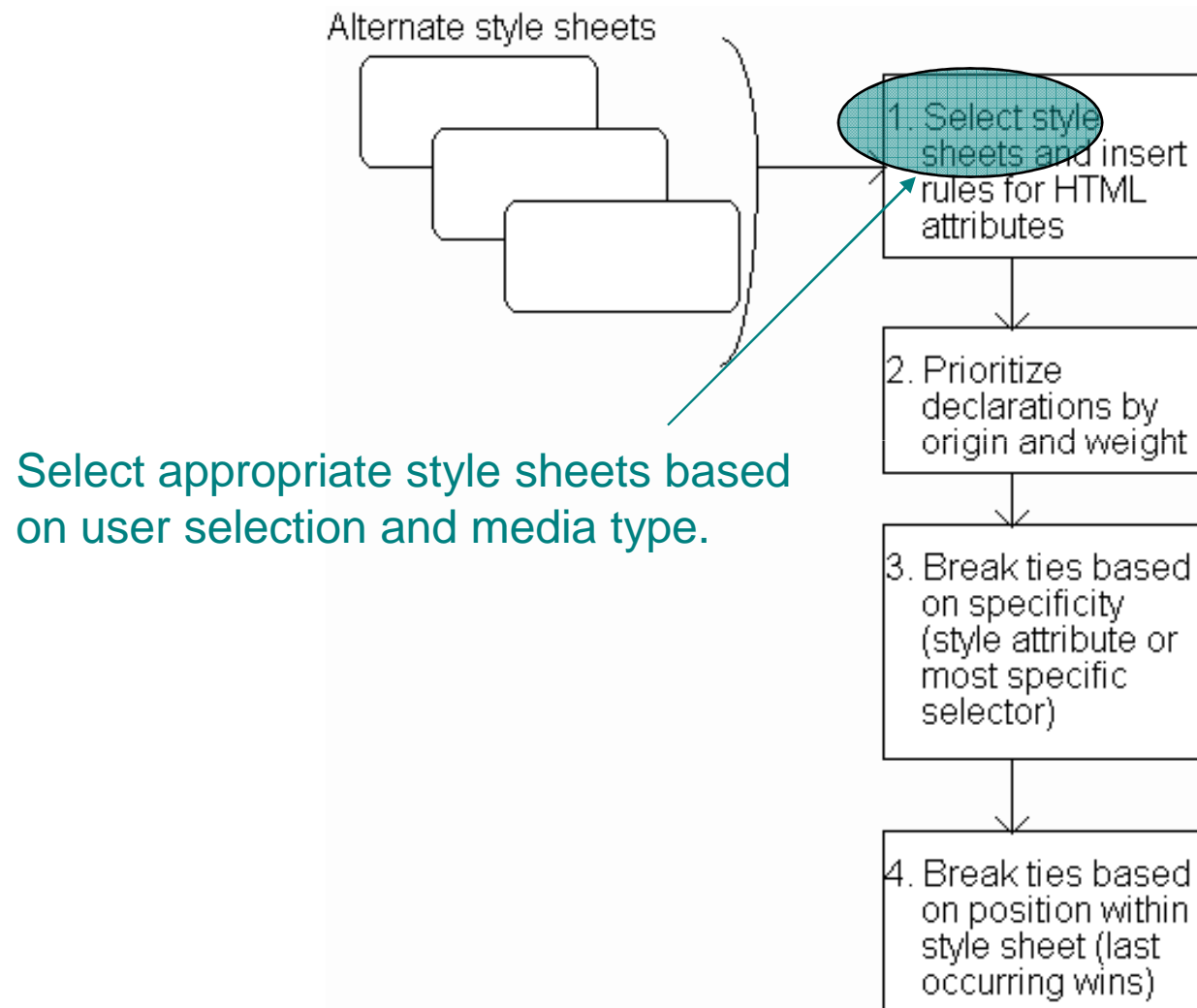
```
* { font-weight:bold }  
#p3 { font-weight:normal }
```

- The CSS **rule cascade** determines which style rule's declaration applies

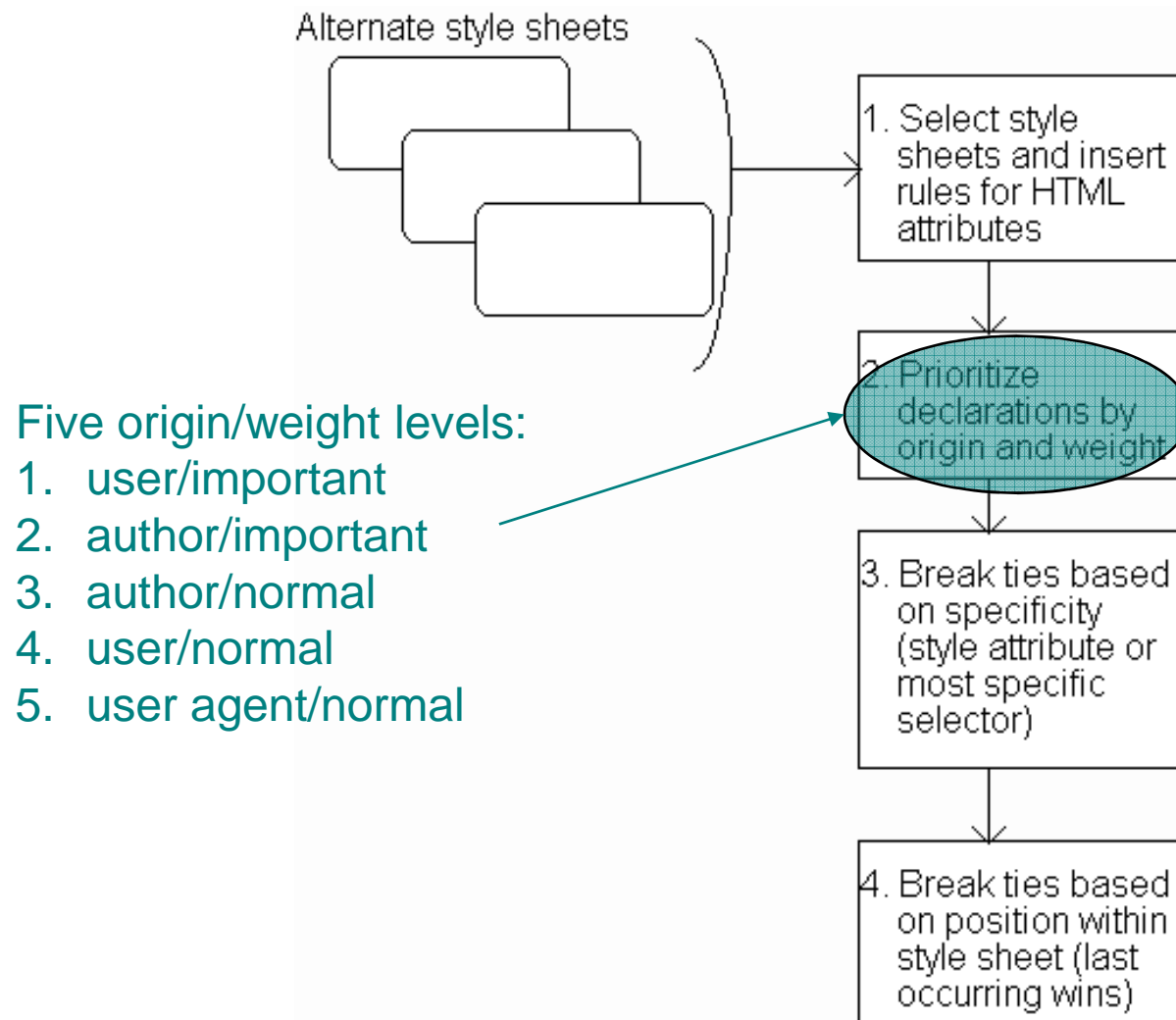
# CSS Rule Cascade



# CSS Rule Cascade



# CSS Rule Cascade



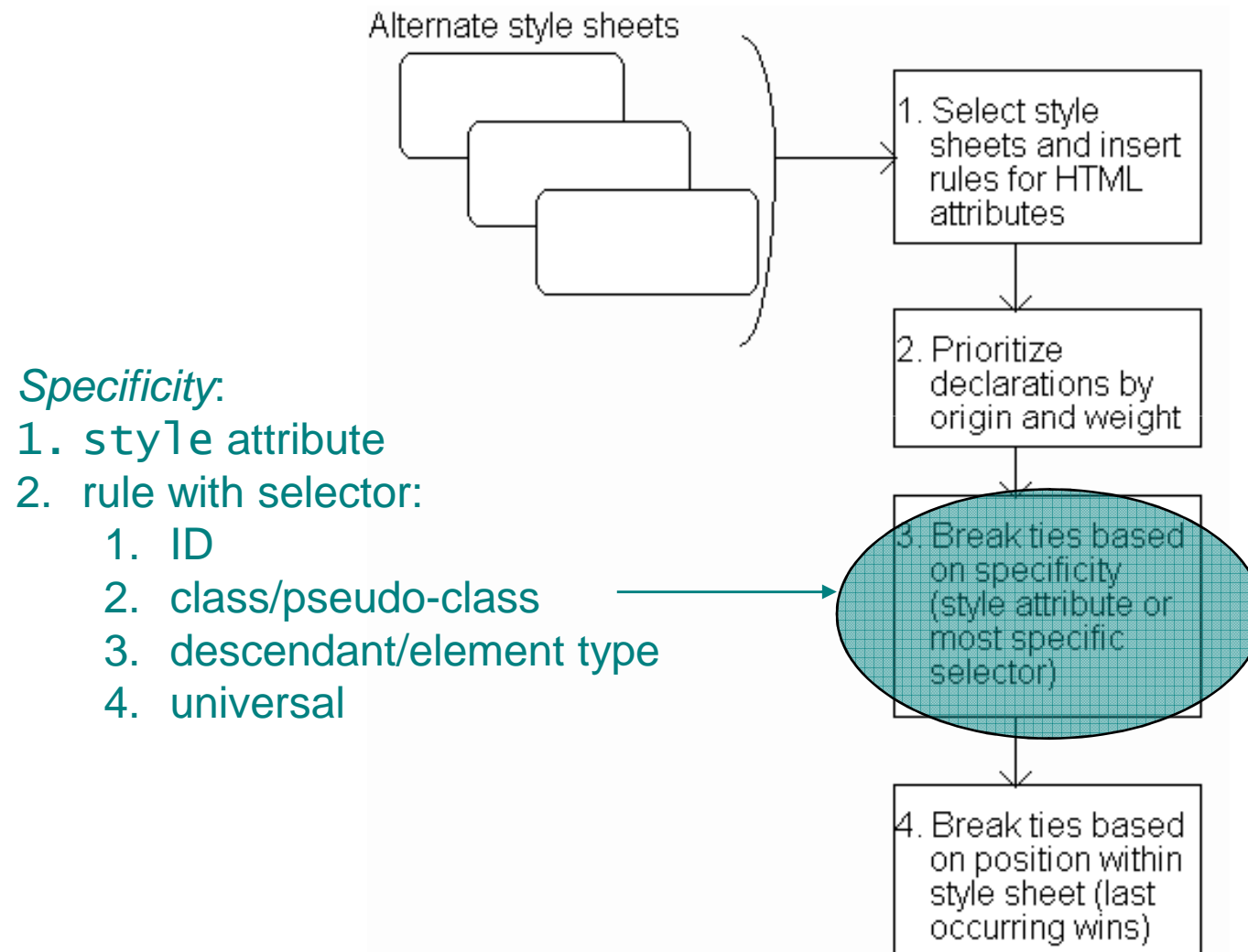


# CSS Rule Cascade

- 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”:

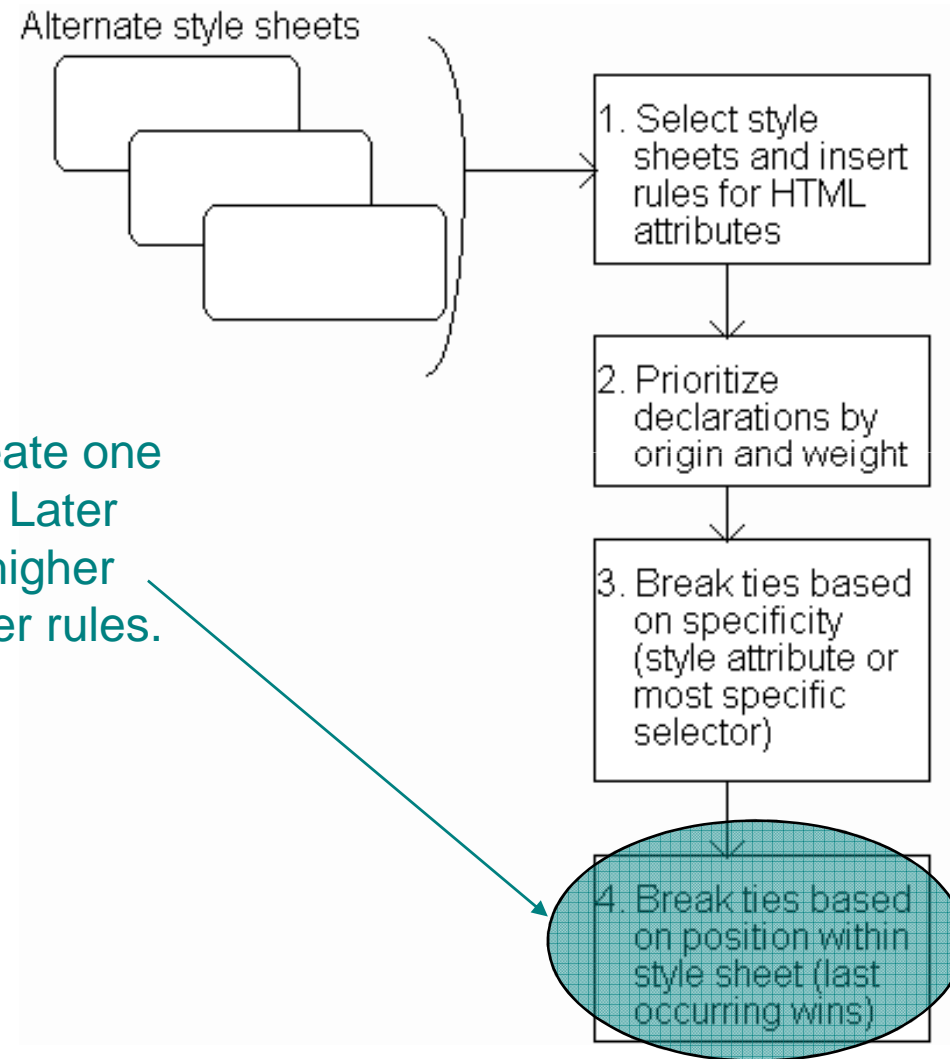
```
p { text-indent:3em; font-size:larger !important }
```

# CSS Rule Cascade



# CSS Rule Cascade

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



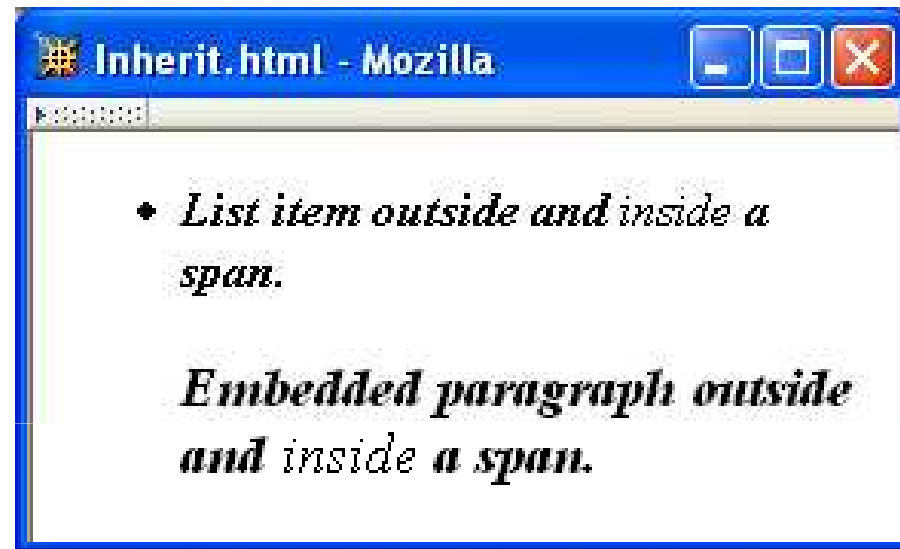
# CSS Inheritance

- 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

```
body { font-weight:bold }  
li { font-style:italic }  
p { font-size:larger }  
span { font-weight:normal }
```

```
<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

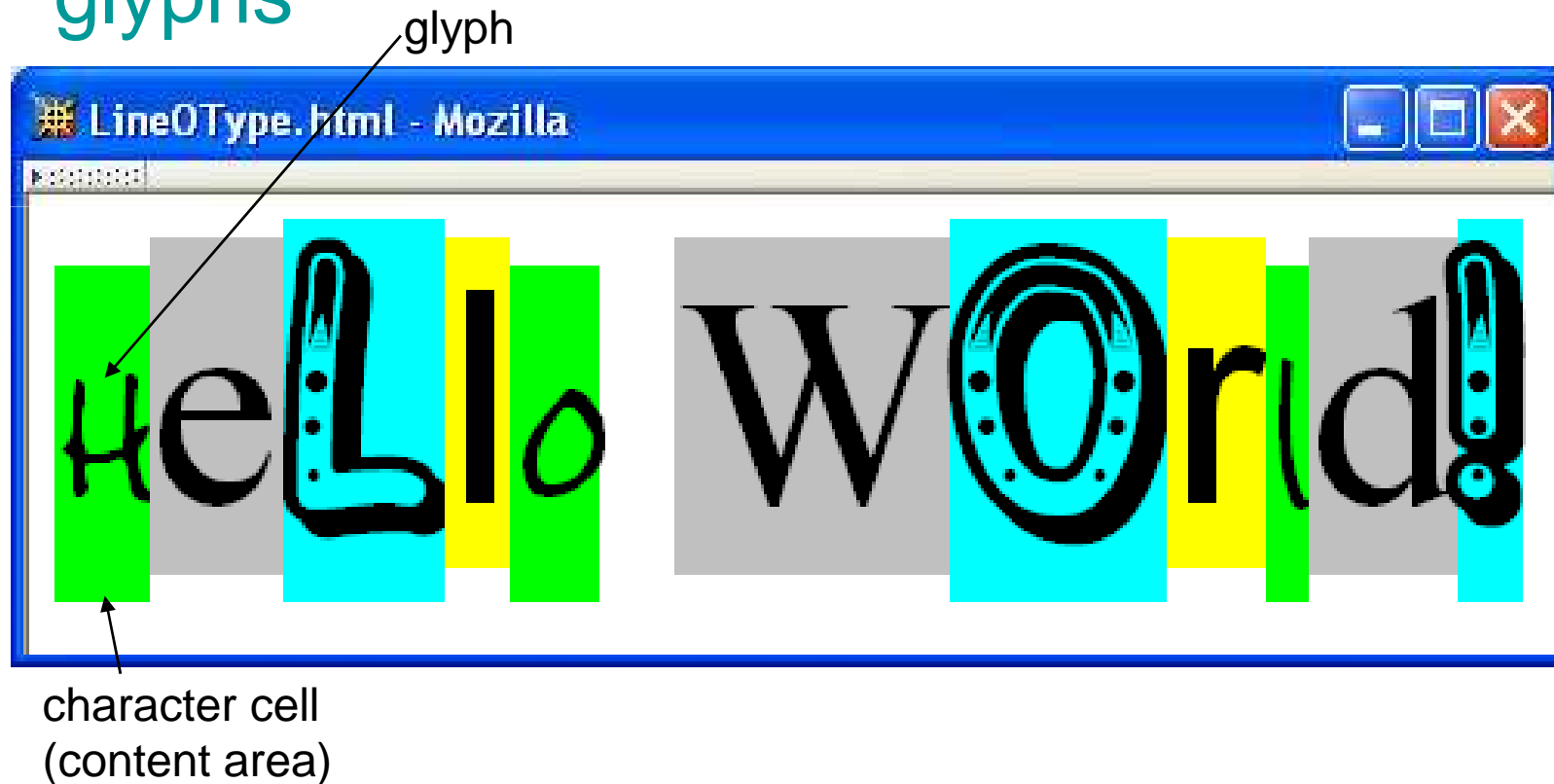
- Property values:
  - Specified: value contained in declaration
    - Absolute: value can be determined without reference to context (e.g., 2cm)
    - Relative: value depends on context (e.g., 1larger)
  - Computed: absolute representation of relative value (e.g., 1larger might be 1.2 x parent font size)
  - Actual: value actually used by browser (e.g., computed value might be rounded)

# 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**





# CSS Font Properties

- A **font** is a mapping from code points to **glyphs**  
glyphs do not necessary stay inside cells!



# CSS Font Properties

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

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

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

```
font-family:"Edwardian Script ITC", "French Script MT", cursive
```

first choice font

# CSS Font Properties

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

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

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

```
font-family:"Edwardian Script ITC","French Script MT",cursive
```


second choice font

# CSS Font Properties

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

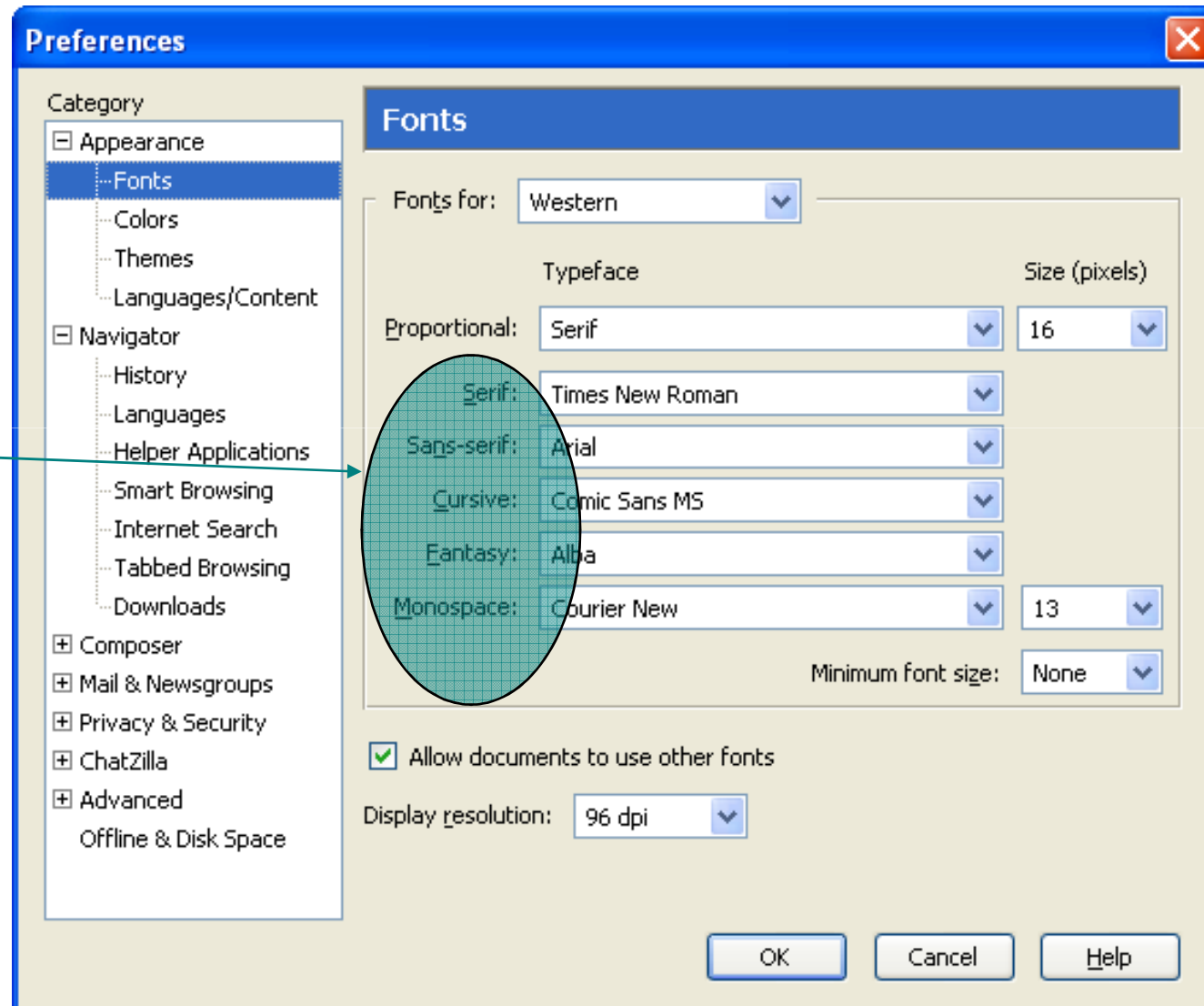
```
<p style="font-family:'Jenkins v2.0'">
```

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

```
font-family:"Edwardian Script ITC","French Script MT",  
generic
```

# CSS Font Properties

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 3.4: CSS length unit identifiers.

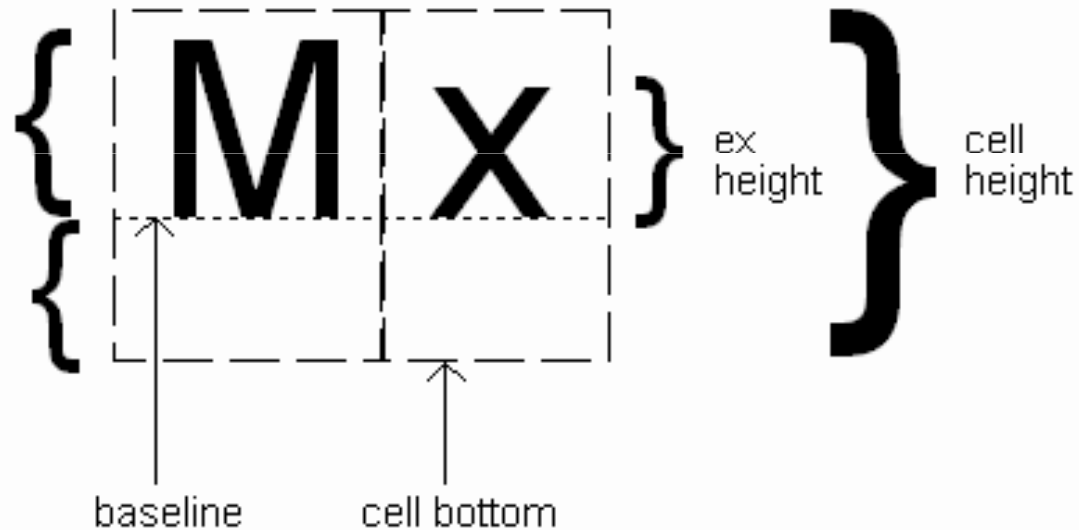
Identifier	Meaning
in	inches
cm	centimeters
mm	millimeters
pt	points: 1/72-inch
pc	picas: 12 points
px	pixel: typically 1/96-inch (see text).
em	1em is roughly the height of a capital letter in the reference font (see text).
ex	1ex is roughly the height of the lowercase 'x' character in the reference font (see text).

# CSS Font Properties

Computed value  
of font-size  
property



baseline  
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

```
<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 3.5: Additional font style properties.

Property	Possible values
<code>font-style</code>	<code>normal</code> (initial value), <code>italic</code> (more cursive than normal), or <code>oblique</code> (more slanted than normal).
<code>font-weight</code>	<code>bold</code> or <code>normal</code> (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).
<code>font-variant</code>	<code>small-caps</code> , which displays lowercase characters using uppercase glyphs (small uppercase glyphs if possible), or <code>normal</code> (initial value)

# CSS Font Properties

- Text is rendered using line boxes



- Height of line box given by **line-height**
  - Initial value: normal (*i.e.*, cell height; relationship with em height is font-specific)
  - Other values (following are equivalent):

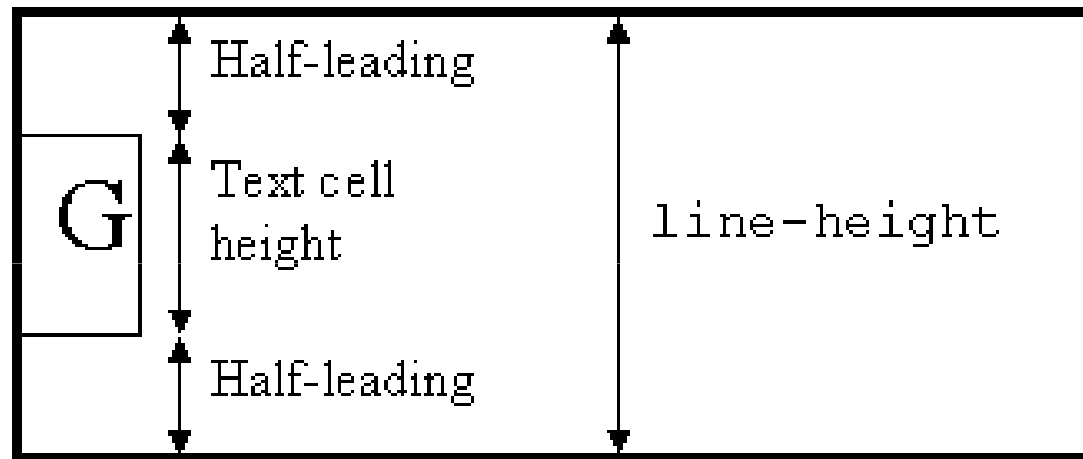
`line-height: 1.5em`

`line-height: 150%`

`line-height: 1.5`

# CSS Font Properties

- When `line-height` is greater than cell height:



- Inheritance of `line-height`:
  - Specified value if normal or unit-less number
  - Computed value otherwise

# CSS Font Properties

- font shortcut property:

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



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

# CSS Font Properties

- font shortcut property:

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



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

# CSS Font Properties

- font shortcut property:

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



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

# CSS Font Properties

- font shortcut property:

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



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



# CSS Font Properties

- font shortcut property:

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



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

# CSS Font Properties

- font shortcut property:

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



```
{ font-style: italic;  
  font-variant: normal;  
  font-weight: bold; Initial values used if no value specified in font  
  font-size: 12pt; property list  
  line-height: normal;  
  font-family: "Helvetica",sans-serif }
```

# CSS Font Properties

- font shortcut property:

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



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

specifying line-height

```
{ font: bold oblique small-caps 12pt/2 "Times New Roman",serif }
```

any order

size and family required,  
order-dependent

# CSS Text Formatting

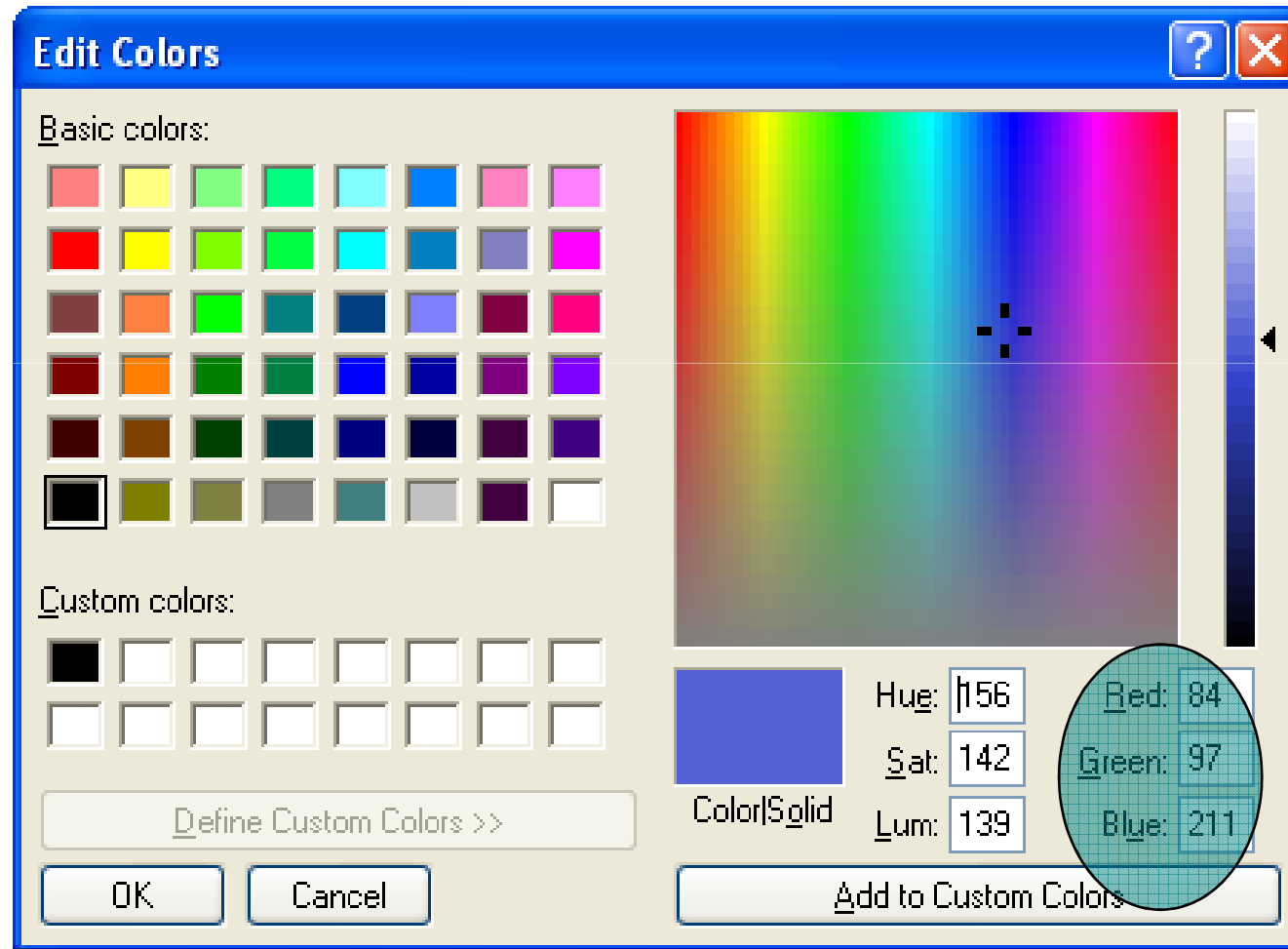
TABLE 3.6: Primary CSS text properties.

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

# 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#ColorKeywords>
  - red/green/blue (RGB) values

# CSS Text Color



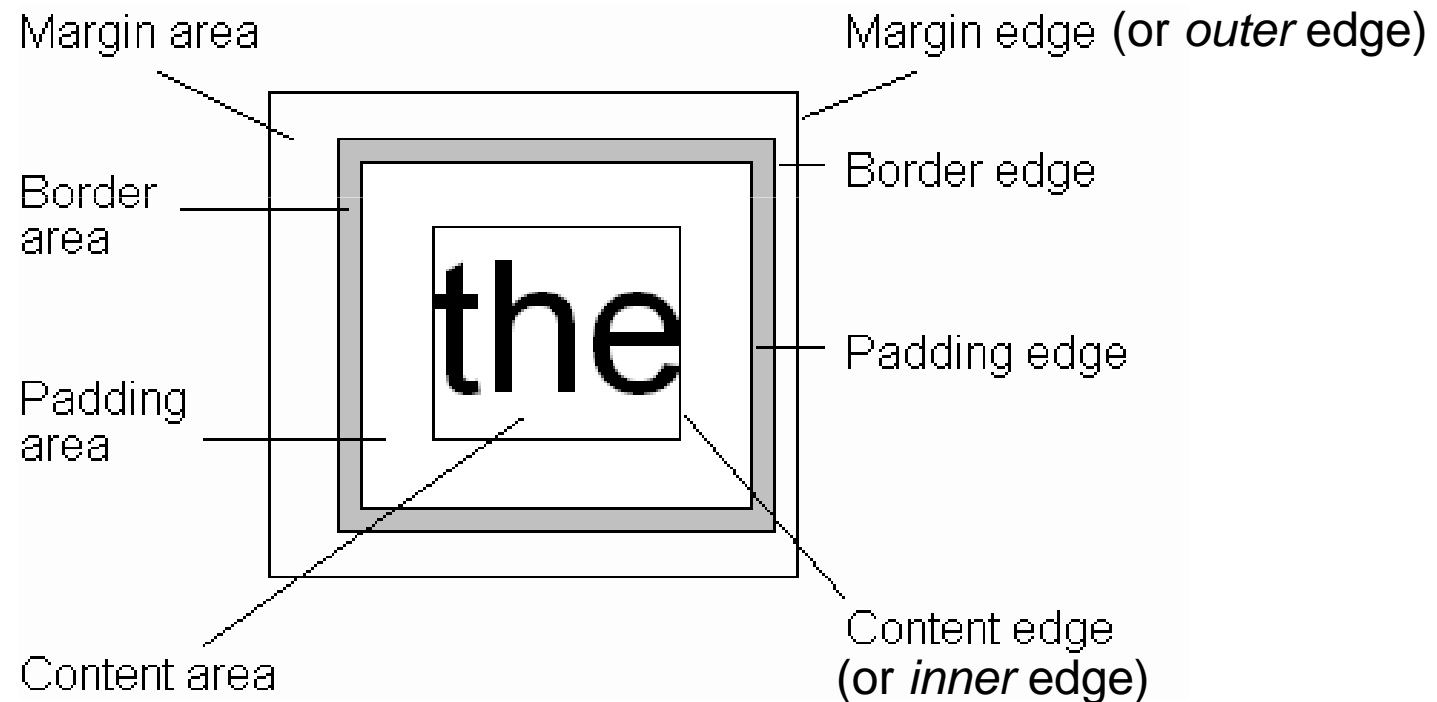
# CSS Text Color

TABLE 3.7: Alternative formats for specifying numeric color values.

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

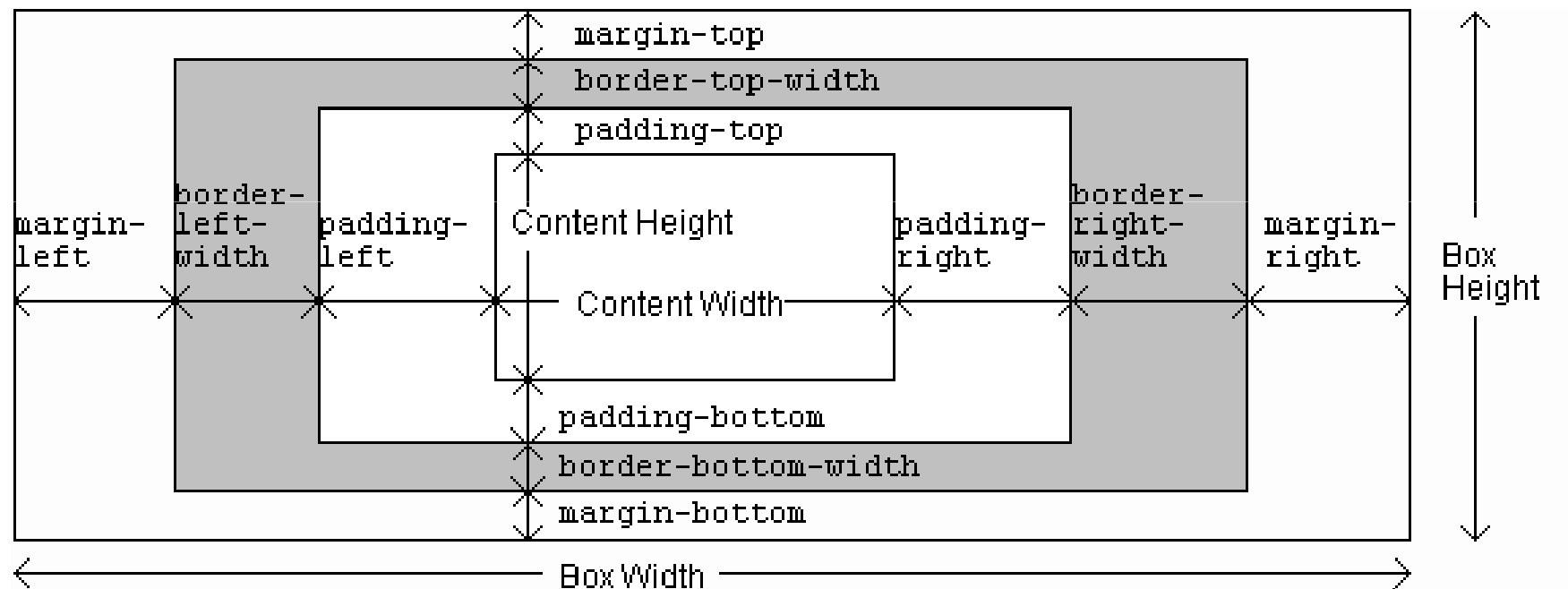
# CSS Box Model

- Every rendered element occupies a box:



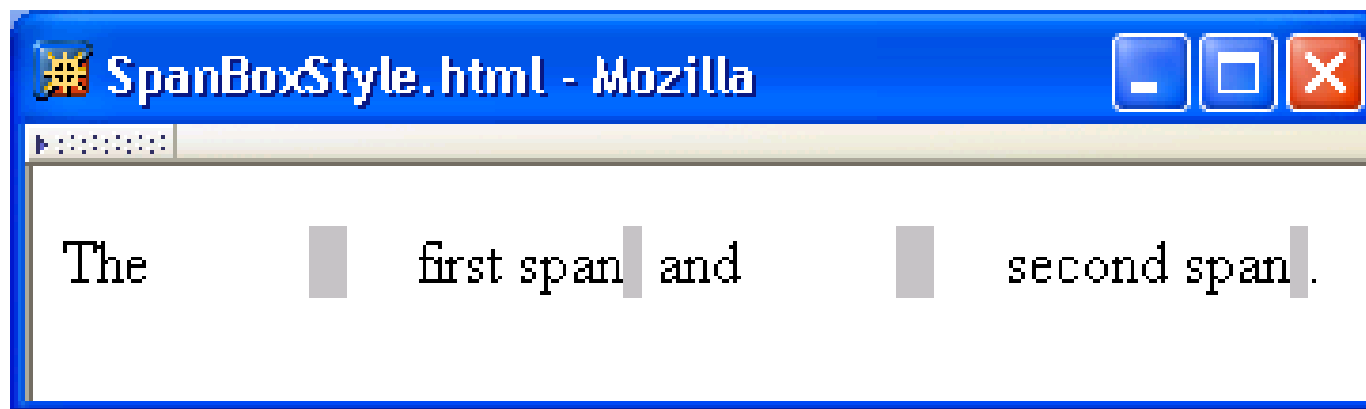


# CSS Box Model



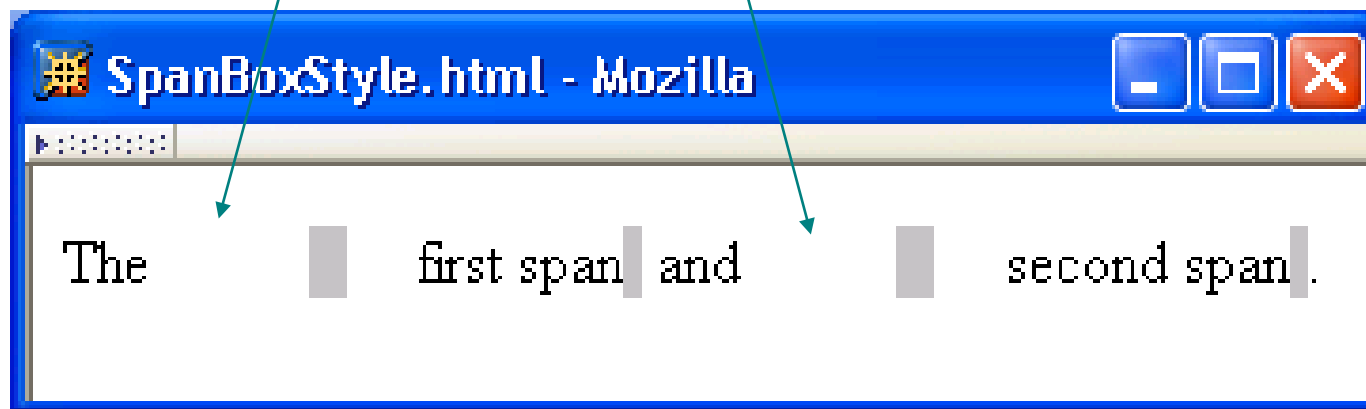
# CSS Box Model

```
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 }
```



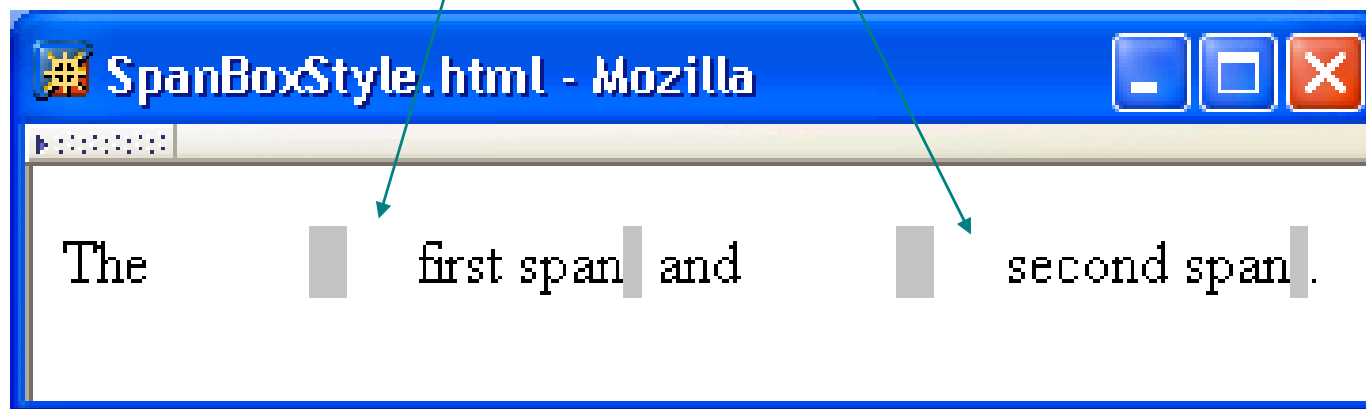
# CSS Box Model

```
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 }
```



# CSS Box Model

```
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 }
```



# CSS Box Model

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

Property	Values
<code>padding={top,right,bottom,left}</code>	CSS length (Sec. 3.6.2).
<code>padding</code>	One to four length values (see text).

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

Number of values	Meaning
One	Assign this value to all four associated properties ( <code>top</code> , <code>right</code> , <code>bottom</code> , and <code>left</code> ).
Two	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.
Three	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> .
Four	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> .

# CSS Box Model

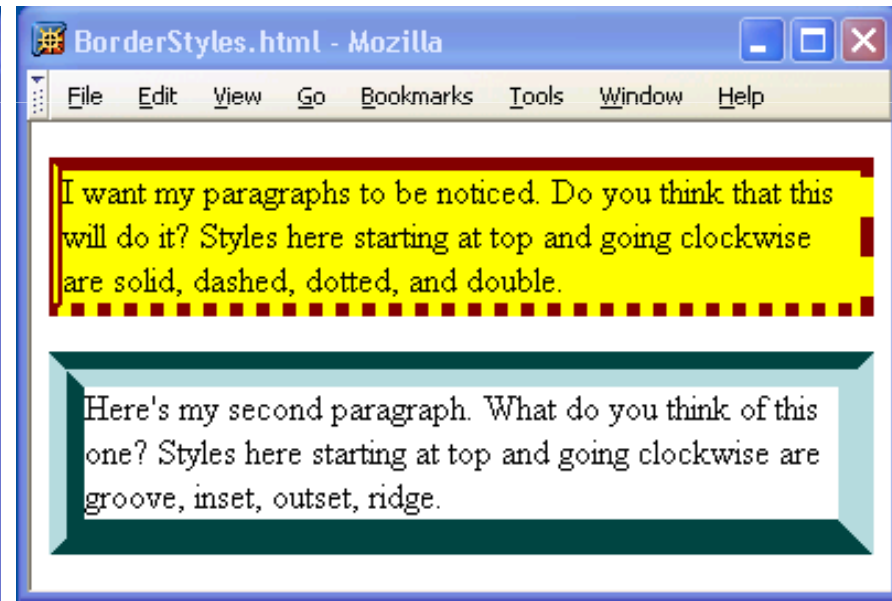
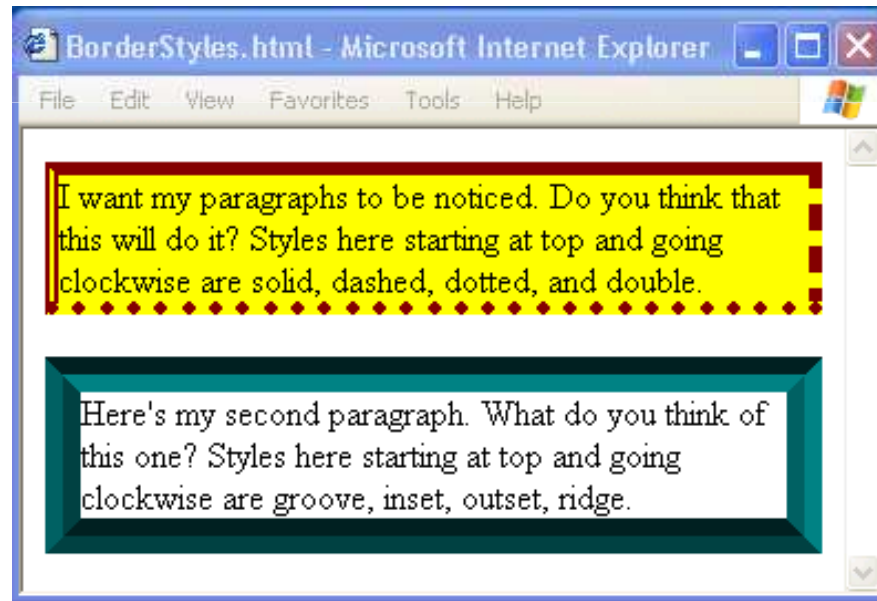
TABLE 3.9: Basic CSS style properties associated with the box model.

<code>border-{top,right,bottom,left}-width</code>	<code>thin</code> , <code>medium</code> (initial value), <code>thick</code> , or a length.
<code>border-width</code>	One to four <code>border-*-width</code> values.
<code>border-{top,right,bottom,left}-color</code>	Color value. Initial value is value of element's <code>color</code> property.
<code>border-color</code>	<code>transparent</code> or one to four <code>border-*-color</code> values.

# CSS Box Model

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

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



# CSS Box Model

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

<code>border-{top,right,bottom,left}</code>	One to three values (in any order) for <code>border-*-width</code> , <code>border-*-color</code> , and <code>border-*-style</code> . Initial values are used for any unspecified values.
<code>border</code>	One to three values; equivalent to specifying given values for each of <code>border-top</code> , <code>border-right</code> , <code>border-bottom</code> , and <code>border-left</code> .
<code>margin-{top,right,bottom,left}</code>	<code>auto</code> (see text) or length.
<code>margin</code>	One to four <code>margin-*</code> values.



# CSS Box Model

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

```
{ 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 **til**ed over an element

# Backgrounds

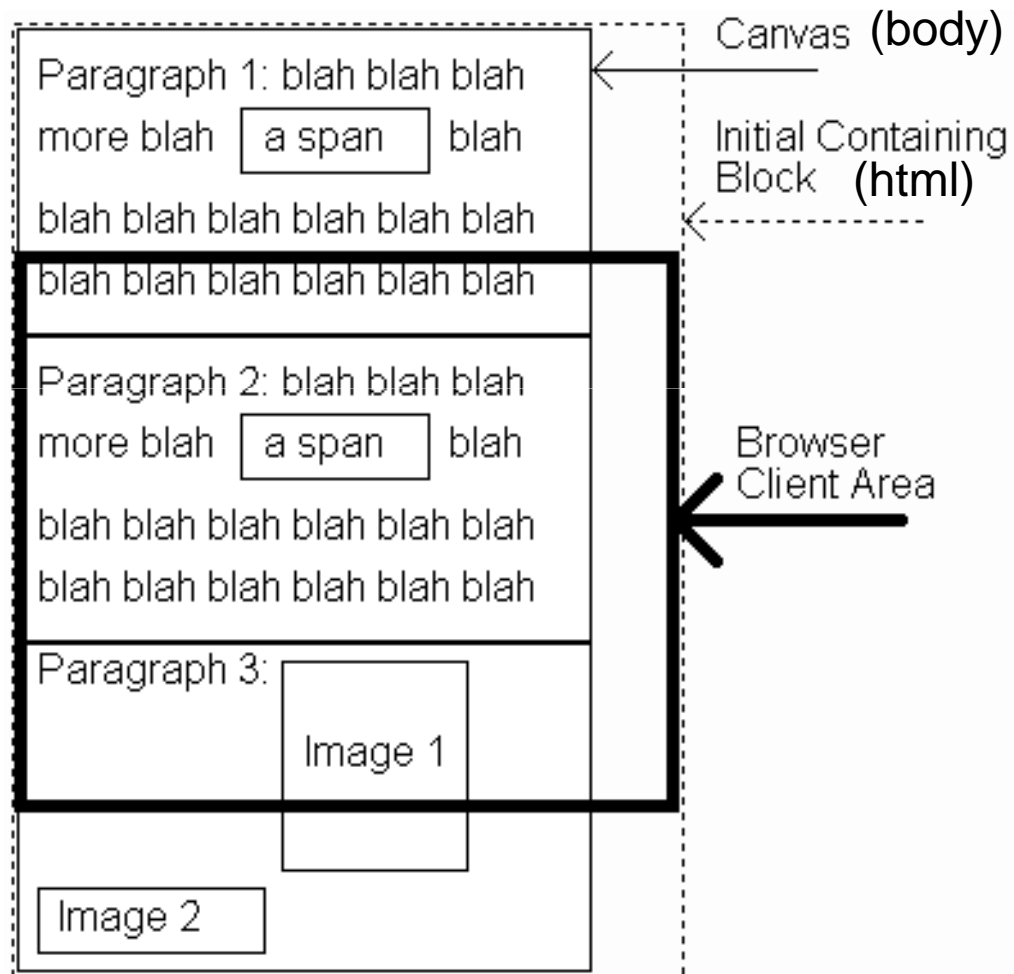
```
<body style="background-image:url('CucumberFlowerPot.png')">
```



# 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



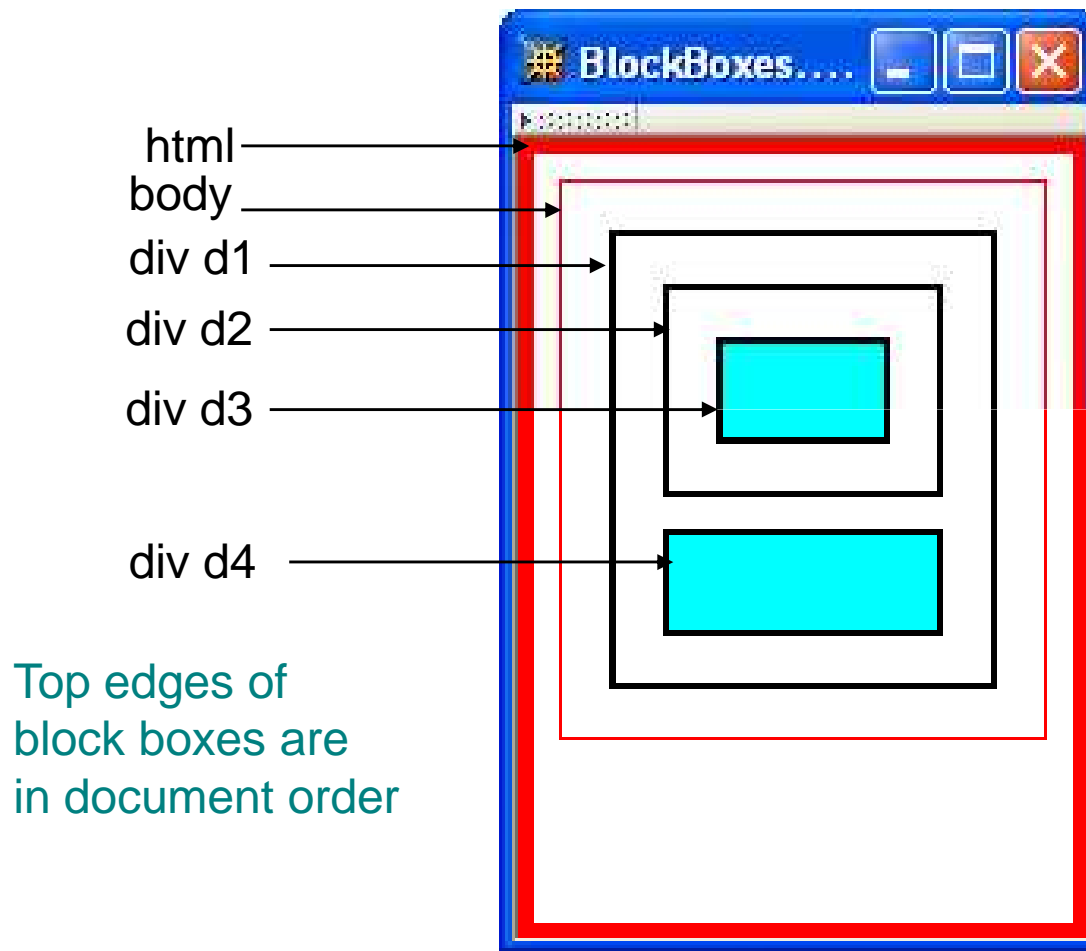
# Normal Flow Layout

```
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 }
```

Block  
elements  
only

```
<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



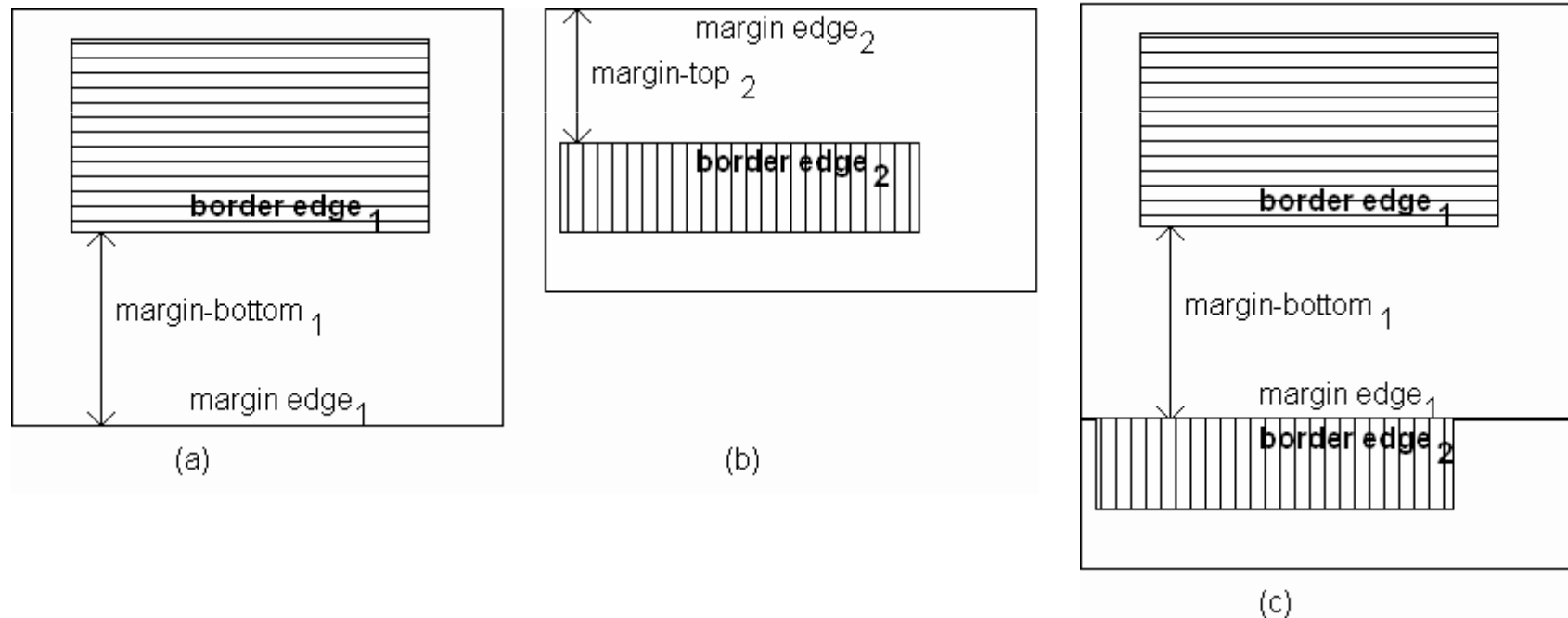
# Normal Flow Layout

- 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`



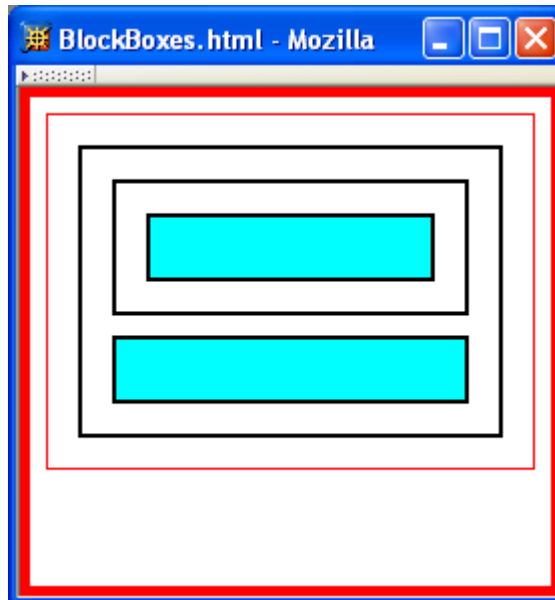
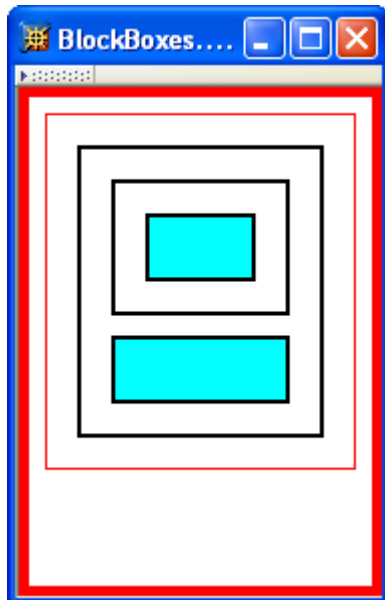
# Normal Flow Layout

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



# Normal Flow Layout

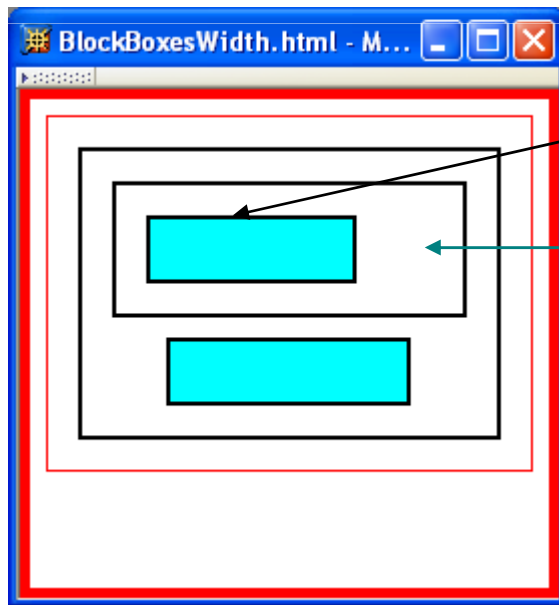
- Initial value of `width` property is `auto`, which for block boxes means to make the content area **as wide as possible within margin/padding constraints**:



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

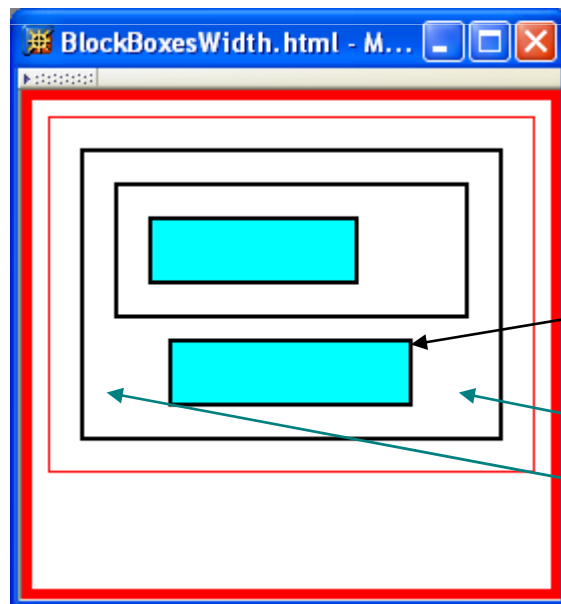


`#d3 { width:50% }`

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

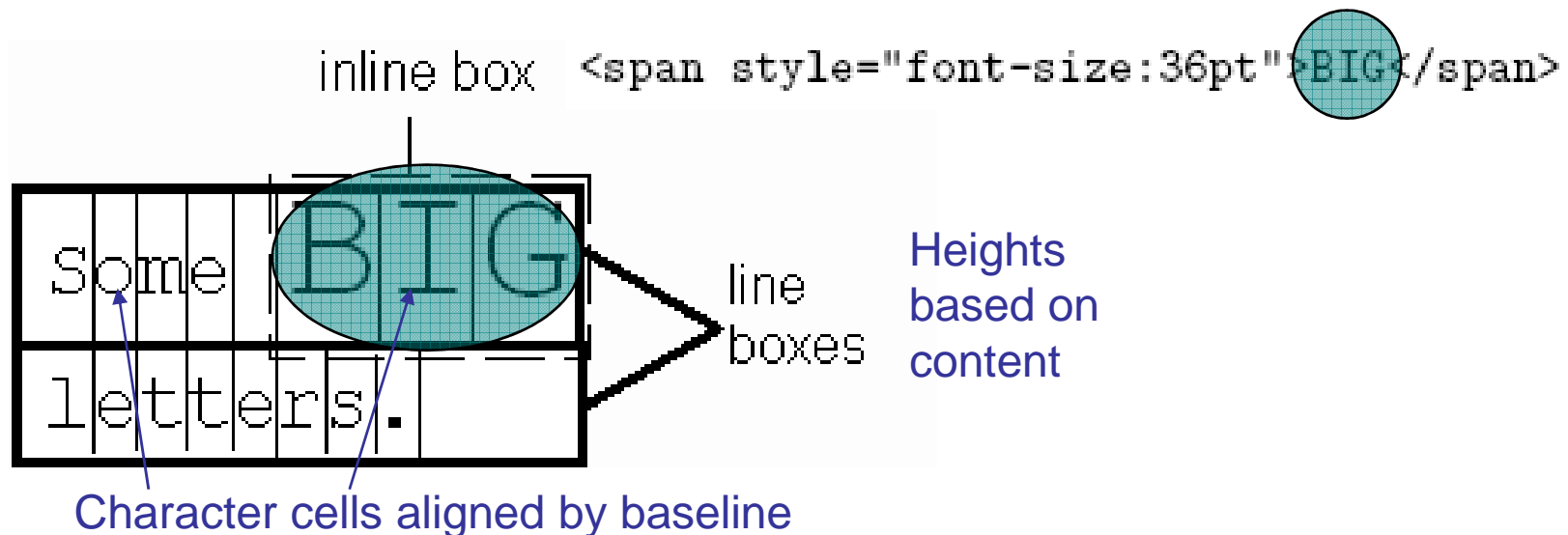


```
#d4 { width:50%; margin-left:auto; margin-right:auto }
```

Centering can be achieved by setting both margins to auto

# Normal Flow Layout

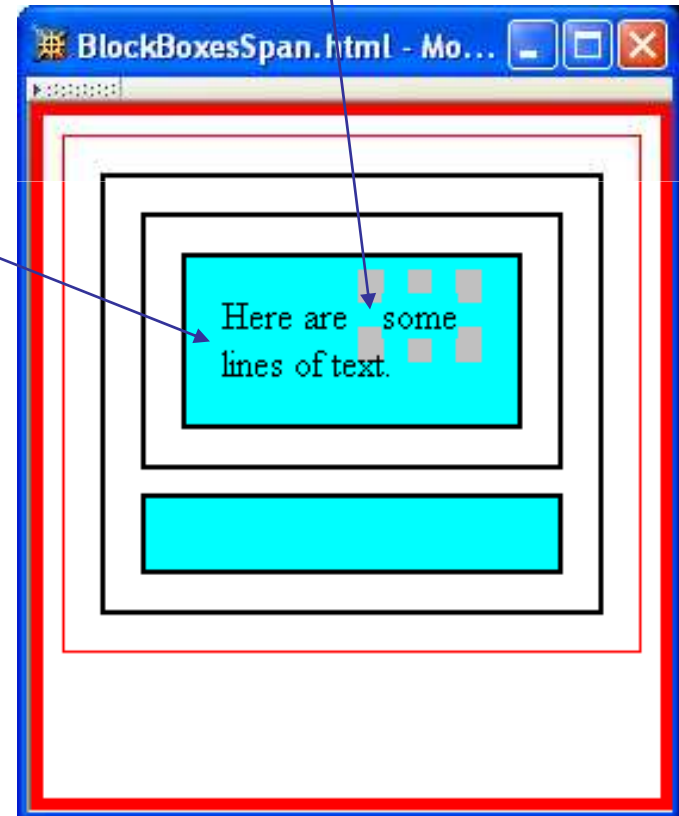
- Boxes corresponding to **character cells** and **inline elements** are laid out side by side in **line boxes** that are stacked one on top of the other



# Normal Flow Layout

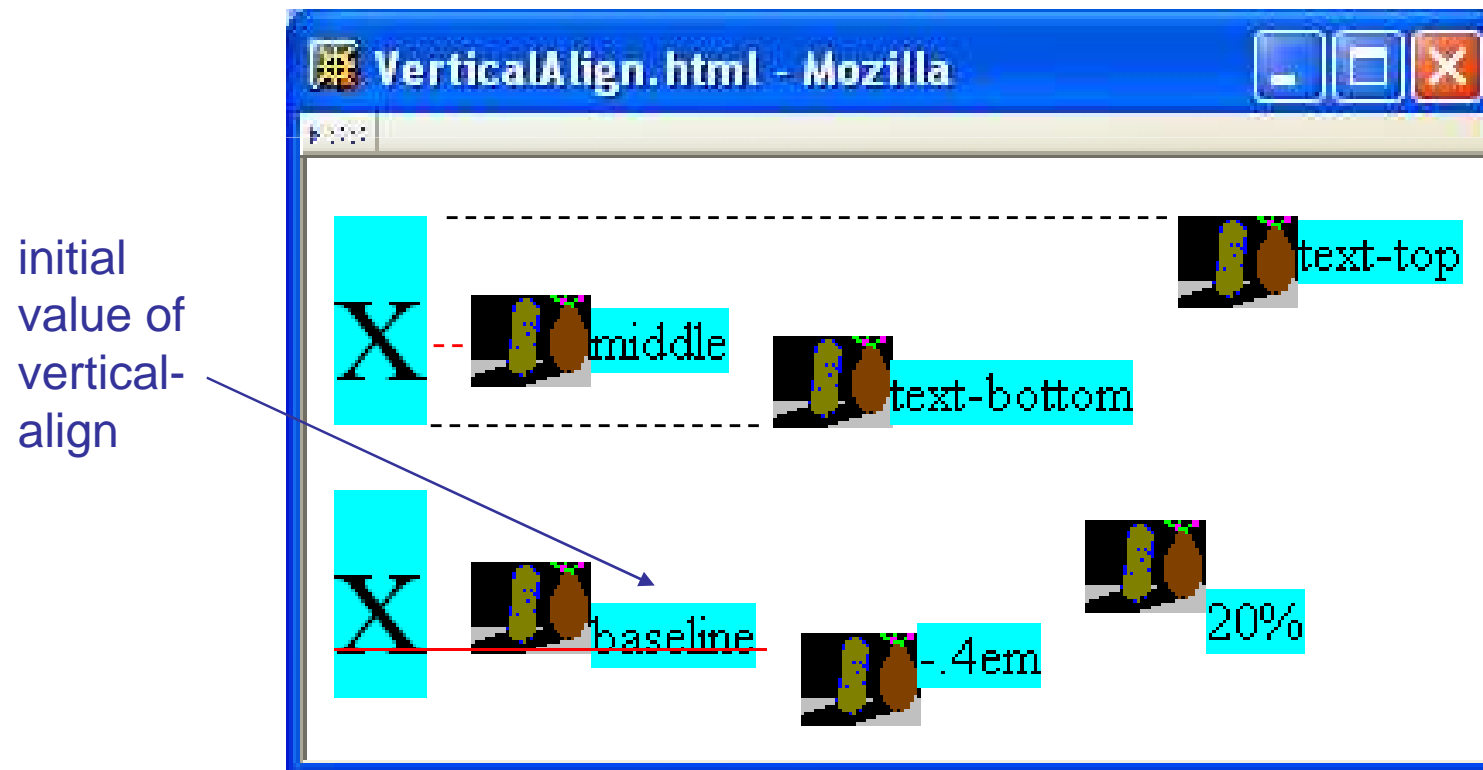
- Padding/borders/margins affect **width** but not **height** of inline boxes

```
<div id="d3" class="shade">  
  Here are  
  <span style="border:dotted silver 10px">some</span>  
  lines of text.  
</div>
```



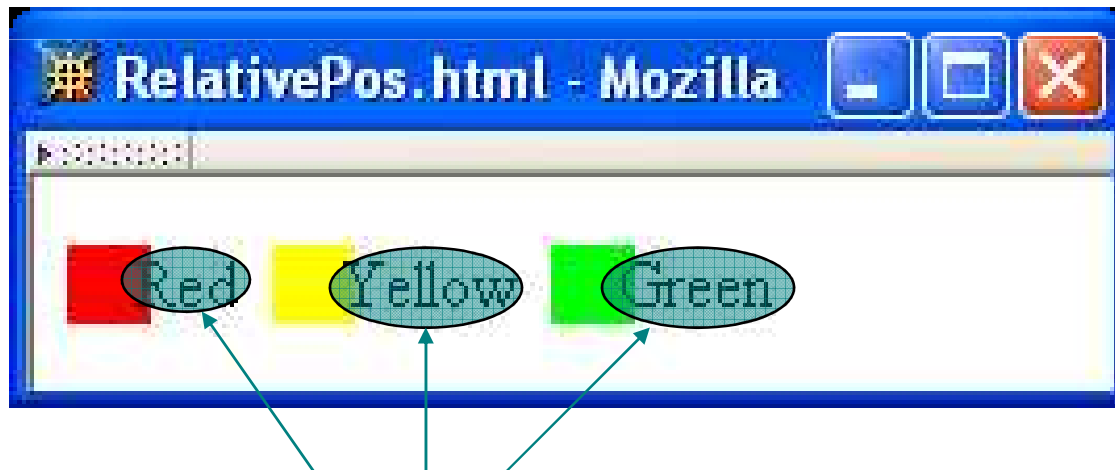
# Normal Flow Layout

- 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:
  - Relative positioning

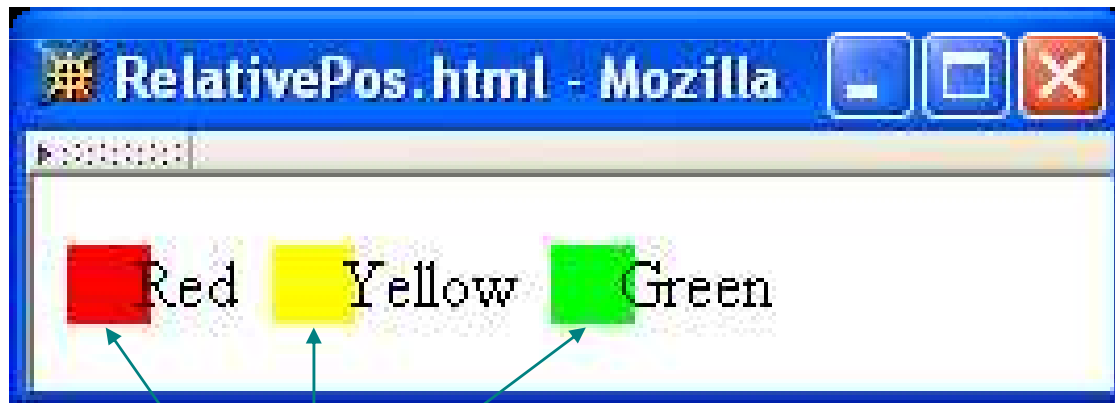


span's shifted backwards relative to normal flow



# Beyond Normal Flow

- CSS allows for boxes to be positioned outside the normal flow:
  - **Relative** positioning



other span's are not affected

# Beyond Normal Flow

- CSS allows for boxes to be positioned outside the normal flow:
  - Relative positioning

[illegible]

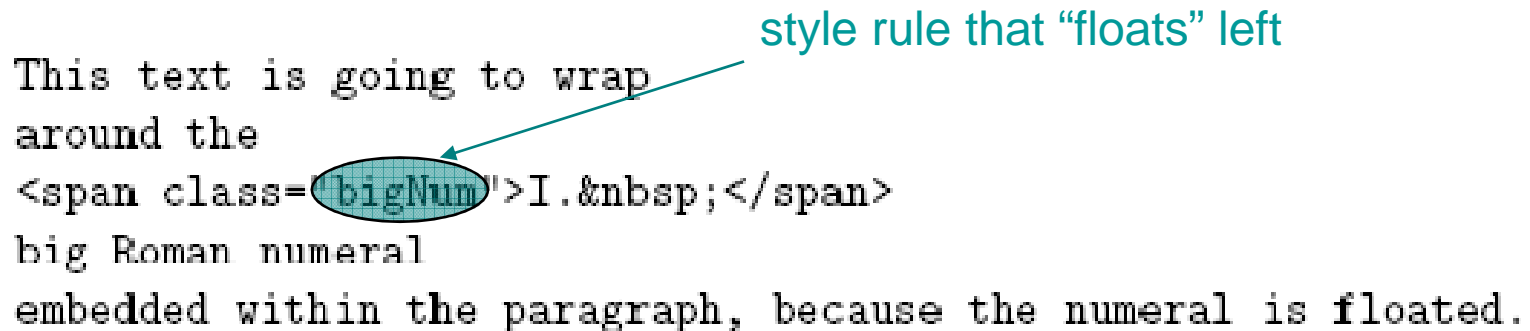
## style rules that move span's away from normal-flow right edge

# Beyond Normal Flow

- CSS allows for boxes to be positioned outside the normal flow:
  - **Float** positioning

This text is going to wrap  
around the  
`<span class="bigNum">I.&nbsp;</span>`  
big Roman numeral  
embedded within the paragraph, because the numeral is floated.

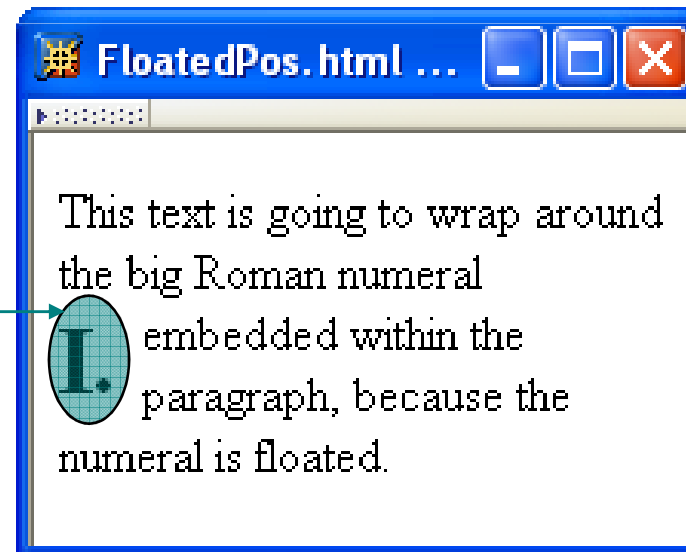
style rule that “floats” left

A diagram illustrating CSS float positioning. The text "This text is going to wrap around the" is positioned to the left of a code snippet. The code snippet is "<span class='bigNum'>I.&nbsp;</span>". The word "bigNum" in the code is circled in blue. A blue arrow points from the text "style rule that 'floats' left" to the circled "bigNum". Below the code snippet, the text "big Roman numeral" and "embedded within the paragraph, because the numeral is floated." is shown. The text "This text is going to wrap around the" is positioned to the left of the code snippet, and the text "big Roman numeral" is positioned below the code snippet.

# Beyond Normal Flow

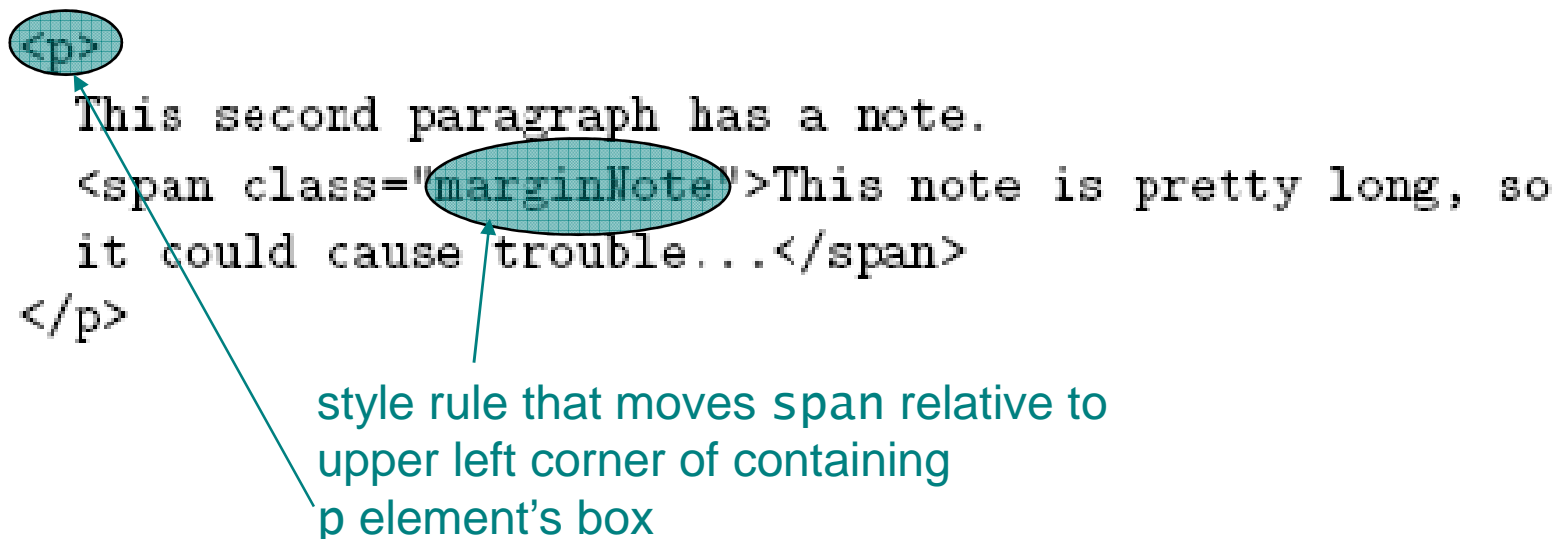
- CSS allows for boxes to be positioned outside the normal flow:
  - **Float** positioning

span taken out of normal flow and “floated” to the left of its line box



# Beyond Normal Flow

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



The diagram illustrates absolute positioning in CSS. It shows a code snippet for a paragraph and a note. The opening paragraph tag `<p>` is circled in blue. The note is enclosed in a `<span class="marginNote">` tag, where the class name `marginNote` is also circled in blue. A blue arrow points from the `<p>` tag to the `marginNote` class, indicating a style rule that positions the span relative to the paragraph's box. The text of the note is "This note is pretty long, so it could cause trouble...".

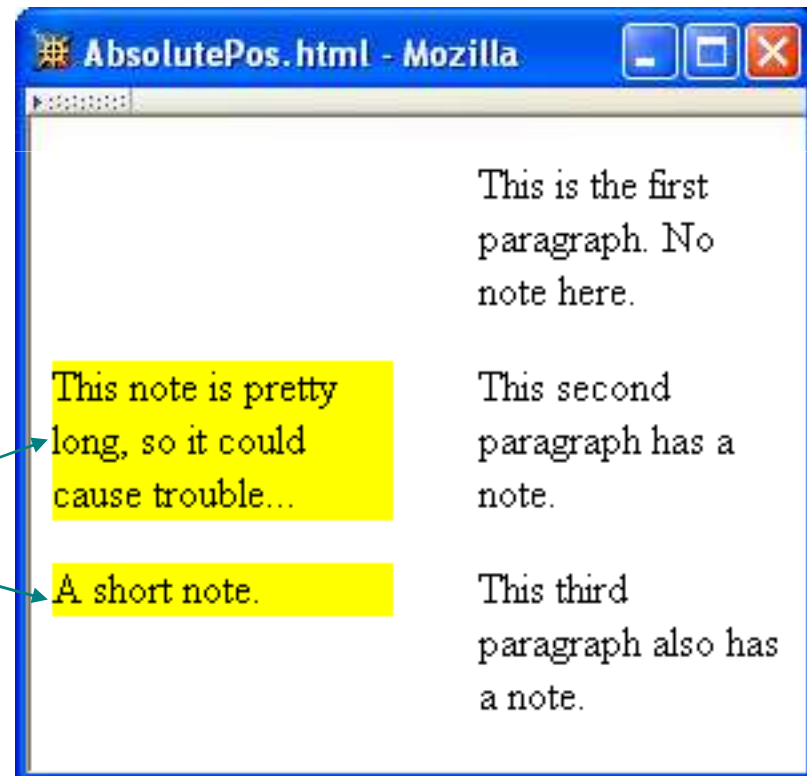
```
<p>
  This second paragraph has a note.
  <span class="marginNote">This note is pretty long, so
  it could cause trouble...</span>
</p>
```

style rule that moves span relative to  
upper left corner of containing  
p element's box

# Beyond Normal Flow

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

span's removed from normal flow and positioned relative to another box



# Beyond Normal Flow

- Properties used to specify positioning:
  - **position**: `static` (initial value), `relative`, or `absolute`
    - Element is **positioned** if this property not `static`
    - Properties `left`, `right`, `top`, `bottom` apply to positioned elements
      - Primary values are `auto` (initial value) or CSS length
  - **float**: `none`, `left`, or `right`
    - Applies to elements with `static` and `relative` positioning only





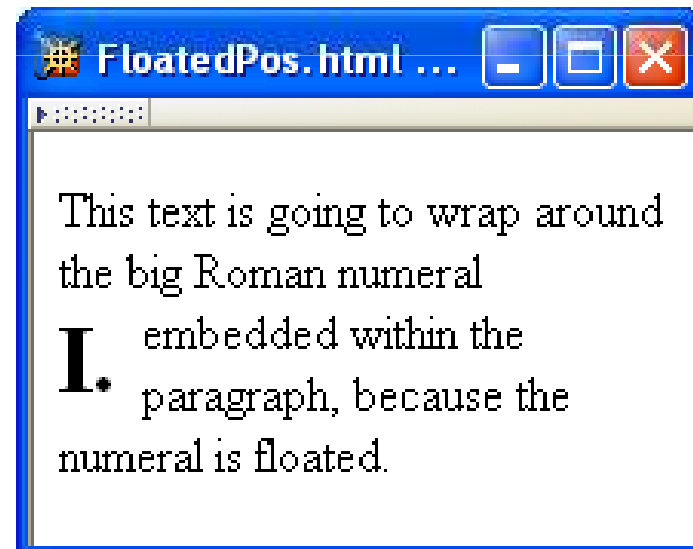


# Beyond Normal Flow

- Float positioning
  - Specify value for float property

```
.bigNum { float:left; font-size:xx-large; font-weight:bold }
```

This text is going to wrap  
around the  
<span class="bigNum">I.&nbsp;  </span>  
big Roman numeral



# Beyond Normal Flow

- Float positioning
  - Specify value for float property

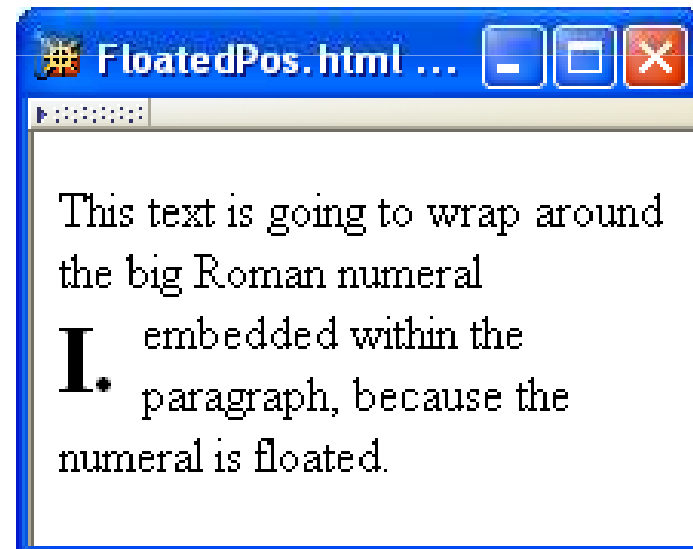
```
.bigNum { float:left; font-size:xx-large; font-weight:bold }
```

This text is going to wrap  
around the

`<span class="bigNum">I.&nbsp;  </span>`

big Roman numeral

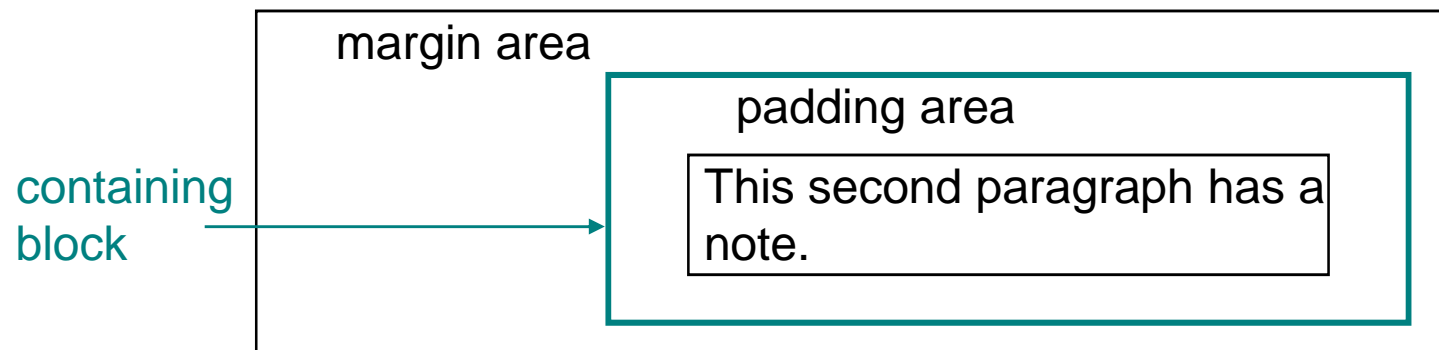
Floated element becomes a CSS block  
element (e.g., can set height and width)



# Beyond Normal Flow

- Absolute positioning
  - Specify location for corner of box relative to **positioned containing block**

p elements are positioned (but don't move!)  
`p { position: relative; margin-left: 10em }`



# Beyond Normal Flow

- Absolute positioning
  - Specify location for edges of box relative to positioned containing block

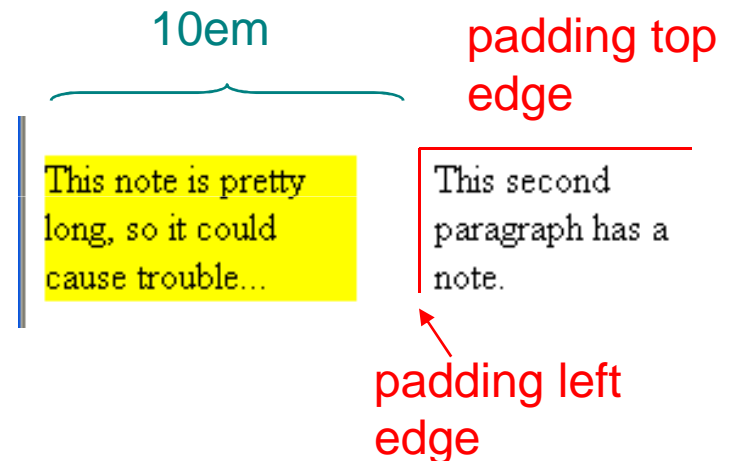
```
.marginNote { position:absolute;  
               top:0; left:-10em; width:8em;  
               background-color:yellow }
```

# Beyond Normal Flow

- Absolute positioning

```
p { position:relative; margin-left:10em; }  
.marginNote { position:absolute;  
               top:0; left:-10em; width:8em;  
               background-color:yellow }
```

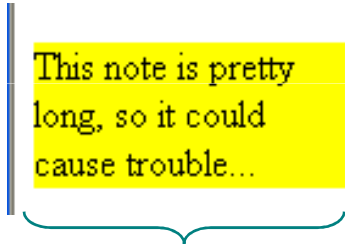
```
<p>  
  This second paragraph has a note.  
  <span class="marginNote">This note is pretty long, so  
  it could cause trouble...</span>  
</p>
```



# Beyond Normal Flow

- Absolute positioning

```
p { position:relative; margin-left:10em }  
.marginNote { position:absolute;  
               top:0; left:-10em; width:8em;  
               background-color:yellow }
```



This note is pretty  
long, so it could  
cause trouble...

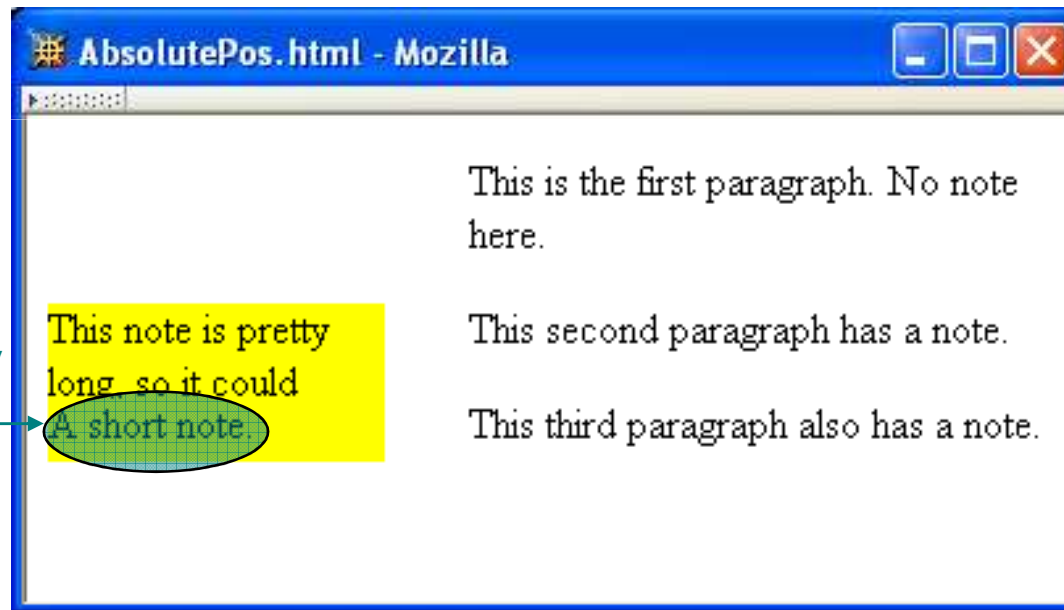
This second  
paragraph has a  
note.

```
<p>  
  This second paragraph has a note.  
  <span class="marginNote">This note is pretty long, so  
  it could cause trouble...</span>  
</p>
```

# Beyond Normal Flow

- Absolutely positioned box does *not* affect positioning of other boxes!

Second absolutely positioned box obscures first





# CSS Position-Related Properties

- `z-index`: drawing order for overlaid boxes (largest number drawn last)

```
#text { position:absolute; top:10px; left:10px;  
        font-family:"Courier",monospace; letter-spacing:0.1em;  
        background-color:yellow;  
        z-index:1 }  
#overlay { position:absolute; top:10px; left:10px;  
           width:1.1em; height:4.5em;  
           border:solid red 1px;  
           z-index:2 }
```



# CSS Position-Related Properties

- `display`: value `none` means that element and its descendants are not rendered and *do not* affect normal flow
- `visibility`: value `hidden` (initial value is `visible`) means that element and its descendants are not rendered but still *do* affect normal flow