

# Cascading Style Sheets

CITS3403 Agile Web Development

---

Unit Coordinator: Tim French

2022 Semester 1

# What is CSS?

- CSS stands for *Cascading Style Sheets*
  - stylesheet language for web
  - used to specify the presentation (layout and style) of markup languages
  - can be applied to any XML document as well as HTML5.
  - superseded many HTML attributes that mixed presentation with content

```
body {  
    background-color: lightblue;  
}  
h1 {  
    color: white;  
    text-align: center;  
}  
p {  
    font-family: verdana;  
    font-size: 20px;  
}
```

# Why CSS?

- Separation of content and presentation
- Advantages for the web
  - *Speed* - stylesheet(s) downloaded once, rather than with each page (if content and style information is intermingled)
  - *Maintainability* - can be “centrally” maintained, easier to update
  - *Accessibility* - can make pages appear similar on different browsers and devices
  - *Portability* - eg. printing, porting to new devices
  - *Reduced work* - eg. don't have to specify alignment every time an element is used
  - *Consistency* - make an organisation's web pages have consistent “look and feel” - corporate ID, brand (and update as brand updates)
    - eg. UWA...




# Faculty of Science

## About us

We are ranked 1st in Australia for Life and Agricultural Sciences (Academic Ranking of World Universities 2015)

Learn more about the Faculty



Faculty Home
Current Students
Staff

Site Search
UWA Website
GO

UWA Home > Faculty of Engineering, Computing and Mathematics > Home

The Faculty

Future students

New Students

Current Students

Research

Alumni

Business and industry

Community


Staff

Contact us


Science has the power to change our world for the better. UWA's Faculty of Science is harnessing this power to improve lives and protect our environment.

Ranked 1st in Australia for Life and Agricultural Sciences, our world-renowned scientists are collaborating with international and local organisations to address global significance.

Providing the very best experience is a priority for the Faculty; we successfully equip our students with the skills needed to compete in the employment market.



Future students



Current students


Science Student Office

A team dedicated to helping you

Schools

Centres

# Engineering, Computing and Mathematics



## Cutting Edge Masters Degrees

Take your career to the next level with the Master of Engineering in Oil and Gas or Master of Information Technology.

The Faculty

Courses

Research

Business and industry

Alumni

Community

Current Students

Staff

Contact us

Research

We are innovators in Engineering for Remote Operations, with teams of transdisciplinary researchers offering integrated solutions to the challenges of remote developments.

Research opportunities

Future Students

Our unique teaching and learning approach creates independent graduates who are empowered to change the world and seek solutions to humanity's greatest challenges.


Courses

EZONE UWA

EZONE UWA is a revolutionary space for innovation, collaboration, and multidisciplinary teaching and research across the disciplines that underpin engineering.

EZONE UWA

## NEWS AND EVENTS



### World-class engineering zone for UWA

The University of Western Australia's vision for a new world-leading engineering zone is moving closer to reality, with approval for the first \$80 million of an estimated \$600 million works to build an engineering hub, known as EZONE UWA.

More news

### Faculty Schools

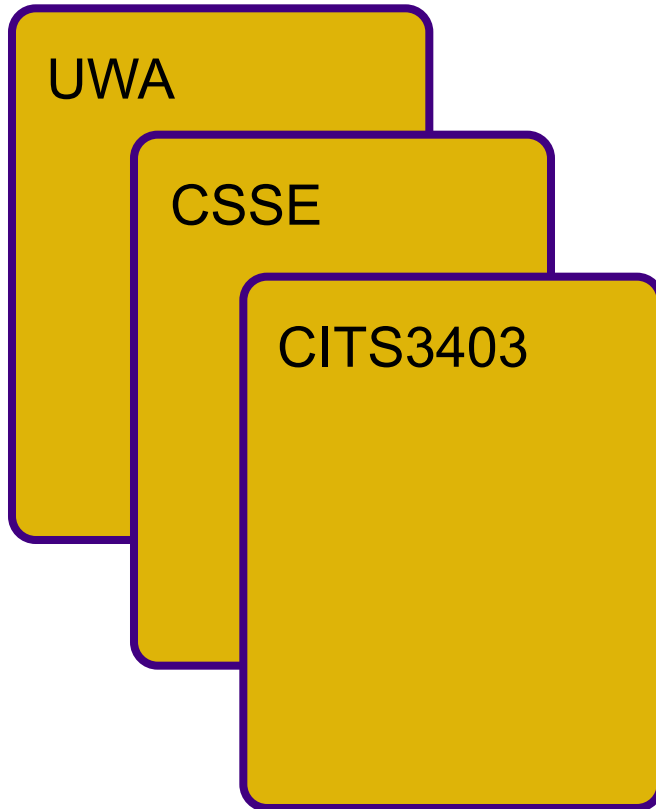
- Civil, Environmental and Mining Engineering
- Computer Science and Software Engineering
- Electrical, Electronic and Computer Engineering
- Mathematics and Statistics
- Mechanical and Chemical Engineering

- **Borders**
  - [border-color](#)
  - [border-image](#)
  - [border-radius](#)
  - [box-shadow](#)
- **Color**
  - [HSL colors](#)
  - [HSLA colors](#)
  - [opacity](#)
  - [RGBA colors](#)
- **Text effects**
  - [text-shadow](#)
  - [text-overflow](#)
  - [word-wrap](#)
- **User-interface**
  - [box-sizing](#)
  - [resize](#)
  - [outline](#)
  - nav-top, nav-right, nav-bottom, nav-left
- **Selectors**
  - [attribute selectors](#)
- **Basic box model**
  - overflow-x, overflow-y
- **Generated Content**
  - content
- **Other modules**
  - [media queries](#)
  - [multi-column layout](#)
  - [Web fonts](#)
  - [speech](#)

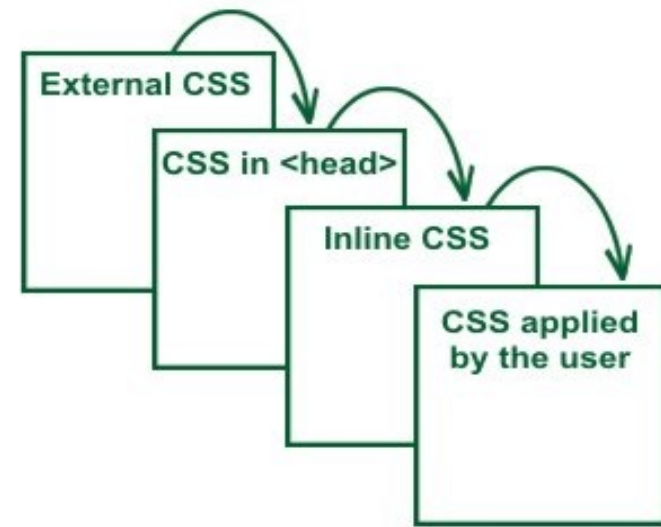
# Why “Cascading”?

- There are three levels of style sheets
  - *Inline* - specified for a specific occurrence of a tag and apply only to that tag
    - This is fine-grain style, which defeats the purpose of style sheets - uniform style
  - *Document-level* style sheets - apply to the whole document in which they appear
  - *External* style sheets - can be applied to any number of documents
- When more than one style sheet applies to a specific tag in a document, the lowest level style sheet has precedence
  - In a sense, the browser searches for a style property spec, starting with inline, until it finds one (or there isn't one)

# Why “Cascading”?



- Inline style sheets appear in the tag itself
- Document-level style sheets appear in the head of the document
- External style sheets are in separate files, potentially on any server on the Internet
  - Written as text files with the MIME type `text/css`



Picture: [http://paulbohman.com/web/css/why\\_cascading](http://paulbohman.com/web/css/why_cascading)

# In-line Style Specification Format

- Style specification appears as the value of the `style` *attribute*

- General form:

```
style = "property_1: value_1; property_2: value_2;...  
        property_n: value_n"
```

- Example:

```
<p style="background: purple; color: white;">  
  This paragraph will have white text on a purple  
  background.  
</p>
```



# Document-level Format

- Style specification appears as a list of rules that are the *content* of a `<style>` tag
- Contained in the document `<head>`
- General form:  
    `<style>`  
        rule list  
    `</style>`
- Form of the rules:  
    **selector** {list of property/values}
  - Each property/value pair has the form:  
        **property: value**
  - Pairs are separated by semicolons, just as in the value of a `<style>` tag

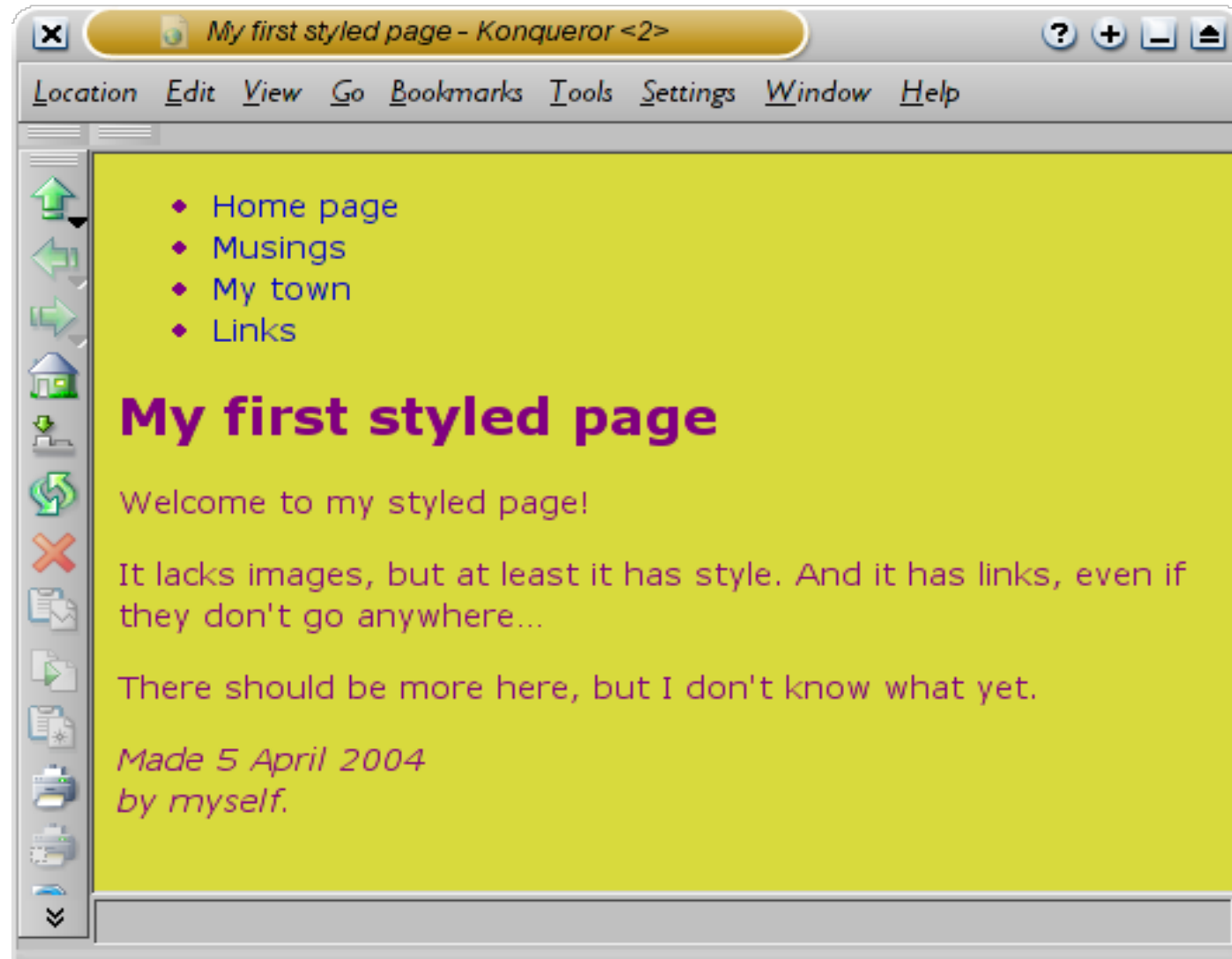
- Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>My first styled page</title>
  <style>
    body {
      color: purple;
      background-color: #d8da3d
    }
  </style>
</head>
<body>

[etc.]
```

–from: WC3 CSS Tutorial,  
<http://www.w3.org/Style/Examples/011/firstcss>

# Document-level Format



# External style sheet format

- A `<link>` tag is used to specify that the browser is to fetch and use an external style sheet file, E.g. Wikipedia style sheet

<http://en.wikipedia.org/skins-1.5/common/shared.css?165>

- External style sheets can be validated

<http://jigsaw.w3.org/css-validator/>

- Form is a list of style rules

**selector {list of property/values}**

as in the content of a `<style>` tag for document-level style sheets

```
<link rel="stylesheet"
      type="text/css"
      href="http://tiny.url/some.css">
</link>
```

```
body {
    color: purple;
    background-color: #d8da3d
}
```

# Selector Forms: Simple

There are numerous ways of specifying to which elements style rules apply. Here are examples of some of the more commonly used:

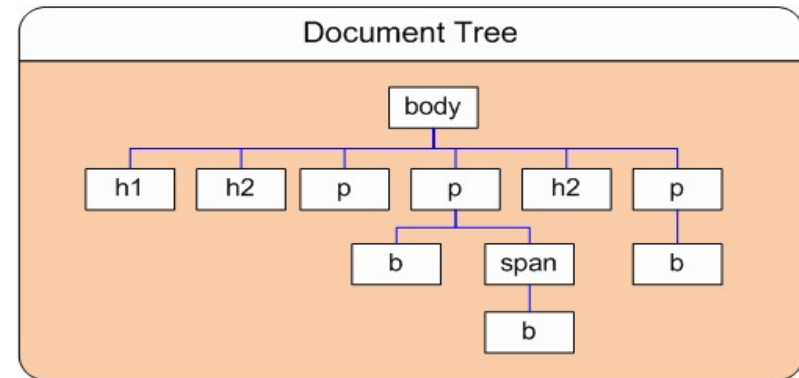
<code>p {color:red}</code>	Every p element
<code>h1,h2,h3 {...}</code>	Group selector
<code>strong em {...}</code>	Contextual selector
<code>div[secret="yes"] {...}</code>	Attribute selector
<code>span.important {...}</code>	Class selector
<code>p#1234 {...}</code>	ID selector

- The *selector* is a tag name or a list of tag names, separated by commas, eg:  
`h1, h2 {font-size: 24pt}`
- Contextual (or descendant) selectors, eg:*  
`body b em {font-size: 14pt}`

Selector	Matches
<code>*</code>	Any element in the hierarchy
<code>e</code>	The specified element in the hierarchy, where <i>e</i> is the specified element
<code>e1, e2, e3, ...</code>	The group of elements <i>e1, e2, e3, ...</i>
<code>e f</code>	The element <i>f</i> when it is a descendant of the element <i>e</i>
<code>e &gt; f</code>	The element <i>f</i> when it is a direct child of the element <i>e</i>
<code>e + f</code>	The element <i>f</i> when it is immediately preceded by the sibling element <i>e</i>

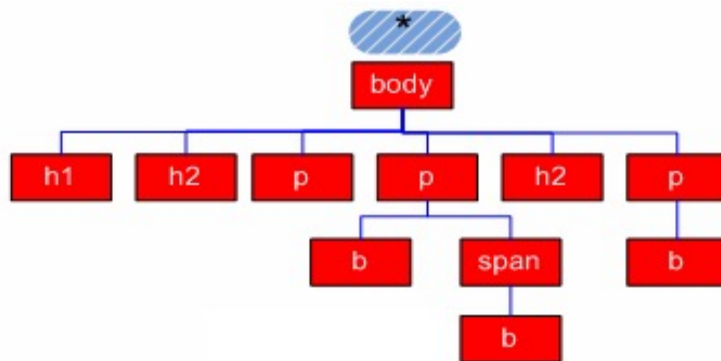
# Selector: Any element

```
<html>
  <head>
    <style type="text/css">
      * {color:red;}
    </style>
  </head>
  <body>
    <h1>Heading 1</h1>
    <h2>Heading 2.a</h2>
    <p>First paragraph. </p>
    <p>Second paragraph has a <b>bold</b> and a <span>span with
    another <b>bold</b></span>. </p>
    <h2>Heading 2.b</h2>
    <p>Third paragraph has a <b>bold</b> also. </p>
  </body>
</html>
```

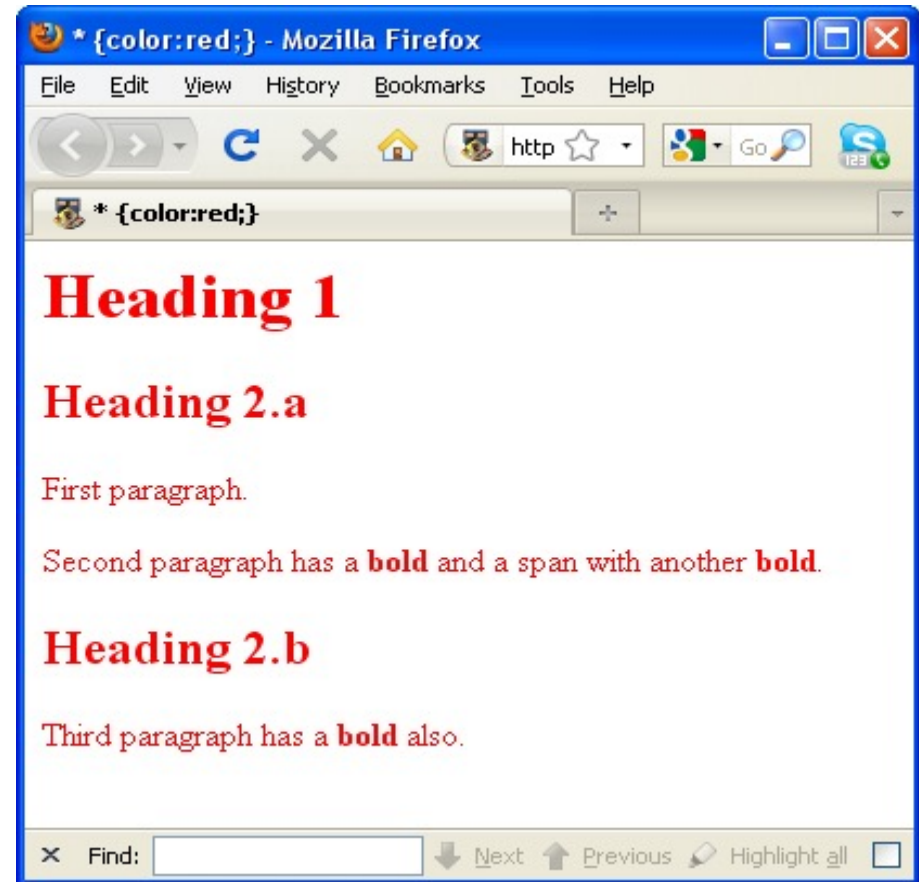




# Selector: Any element

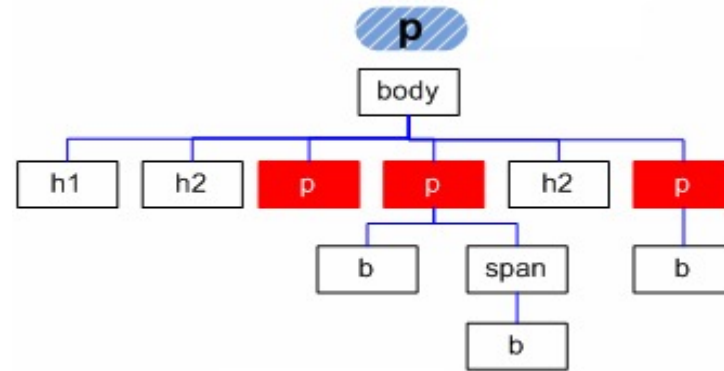


Any selector example

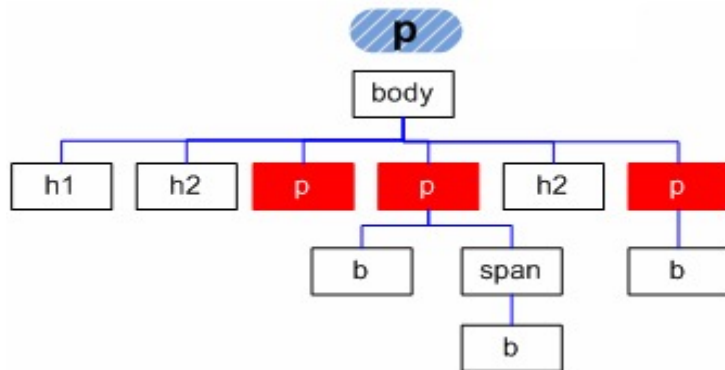


# Example: Selector p

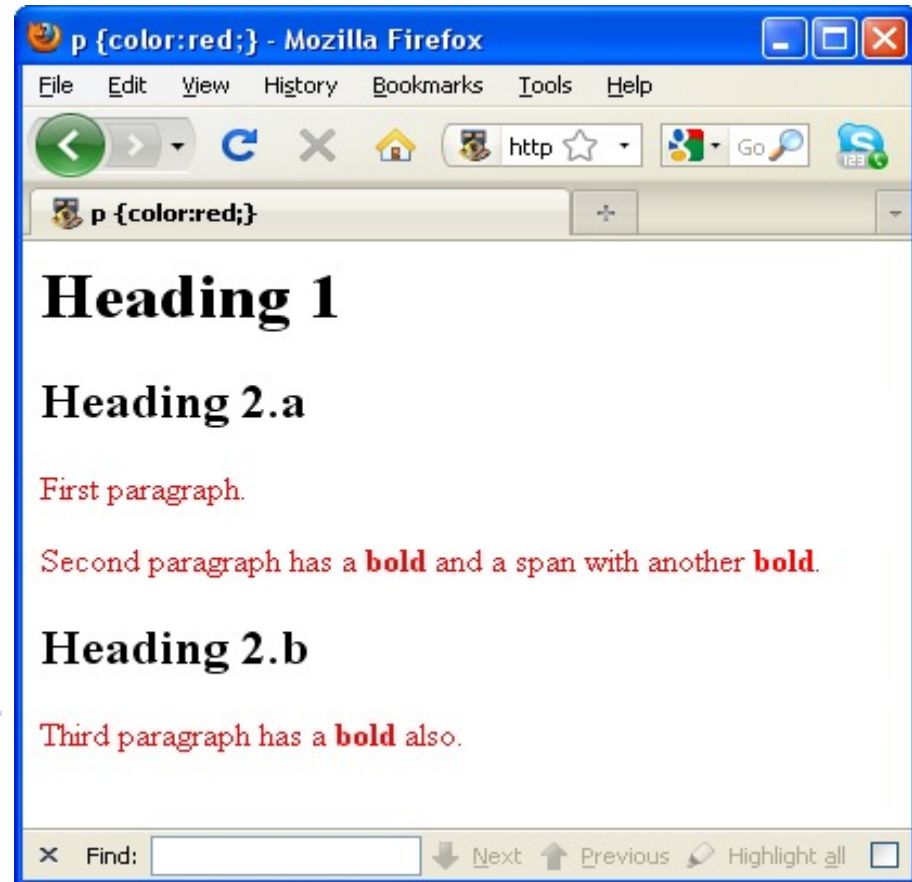
```
<html>
  <head>
    <style>
      p {color:red;}
    </style>
  </head>
  <body>
    <h1> Heading 1</h1>
    <h2> Heading 2.a</h2>
    <p> First paragraph. </p>
    <p> Second paragraph has a <b>bold</b> and a <span>span with another<b>bold</b></span>. </p>
    <h2> Heading 2.b</h2>
    <p> Third paragraph has a <b>bold</b> also. </p>
  </body>
</html>
```



# Example: Selector p

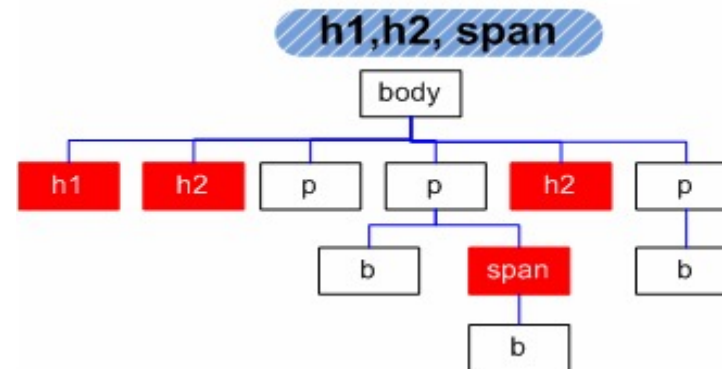


Selector p example

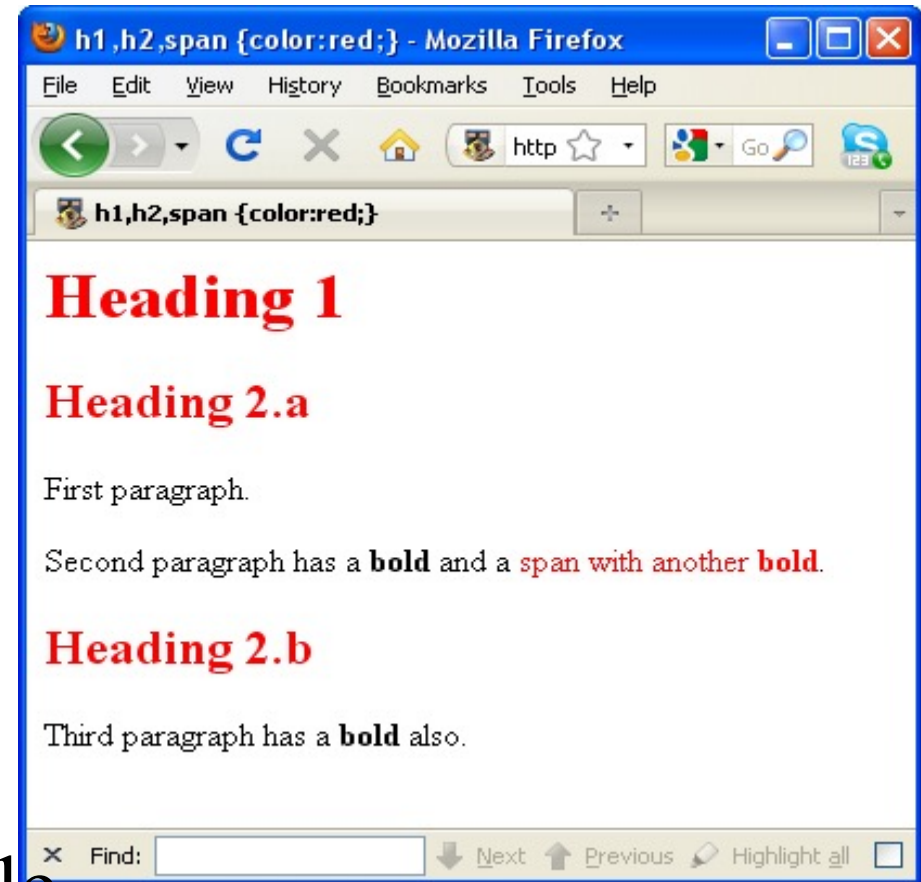
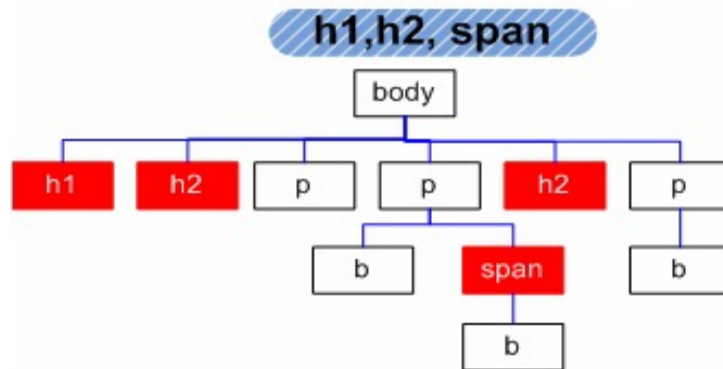


# Example: Selector h1,h2,span

```
<html>
  <head>
    <style type="text/css">
      h1,h2,span {color:red;}
    </style>
  </head>
  <body>
    <h1>Heading 1</h1>
    <h2>Heading 2.a</h2>
    <p>First paragraph. </p>
    <p>Second paragraph has a <b>bold</b> and a <span>span with
    another <b>bold</b></span>. </p>
    <h2>Heading 2.b</h2>
    <p>Third paragraph has a <b>bold</b> also. </p>
  </body>
</html>
```



# Example: Selector h1,h2,span

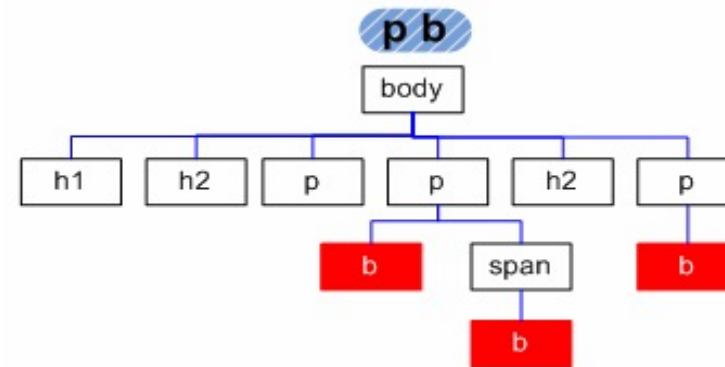


Selector h1,h2,span example

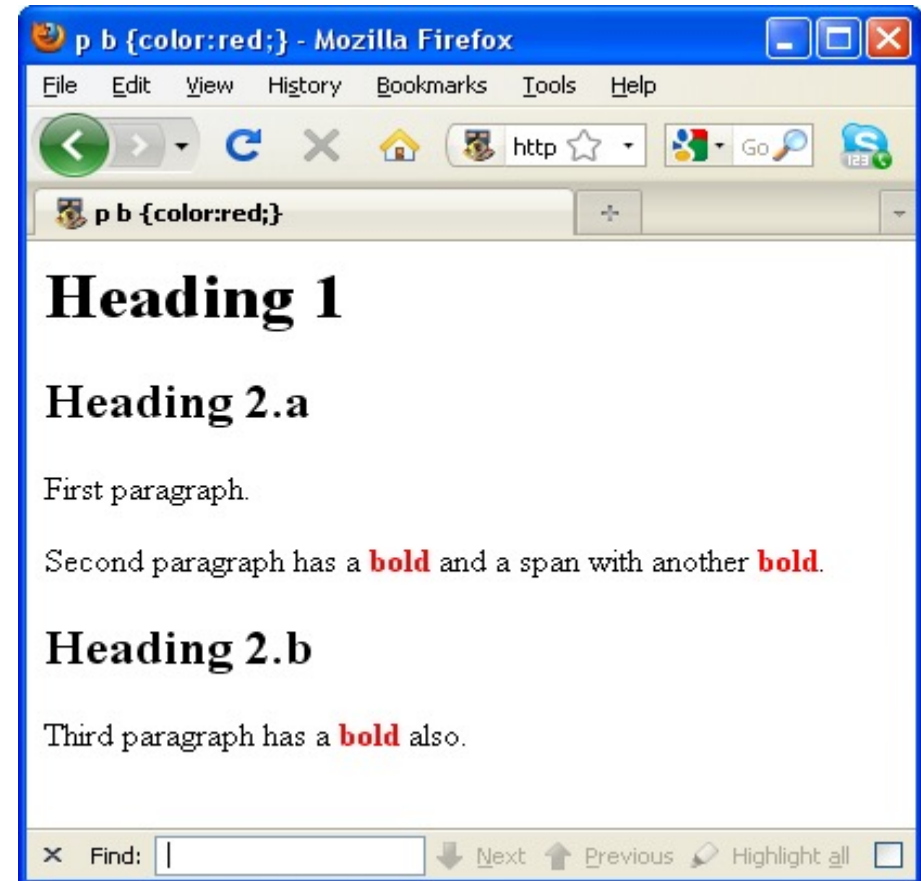
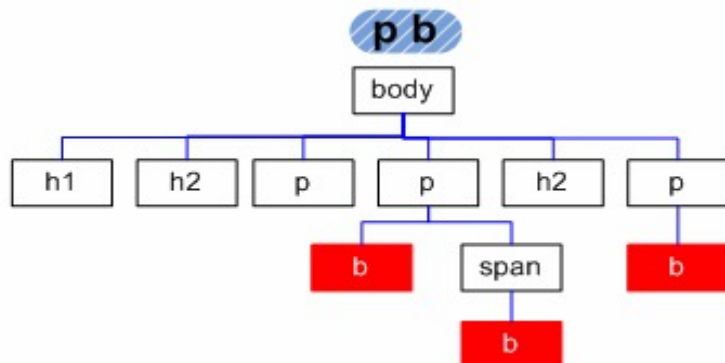


# Example: Selector p b

```
<html>
  <head>
    <style>
      p b {color:red;}
    </style>
  </head>
  <body>
    <h1> Heading 1</h1>
    <h2> Heading 2.a</h2>
    <p> First paragraph. </p>
    <p> Second paragraph has a <b>bold</b> and a <span>span with
    another <b>bold</b></span>. </p>
    <h2> Heading 2.b</h2>
    <p> Third paragraph has a <b>bold</b> also. </p>
  </body>
</html>
```

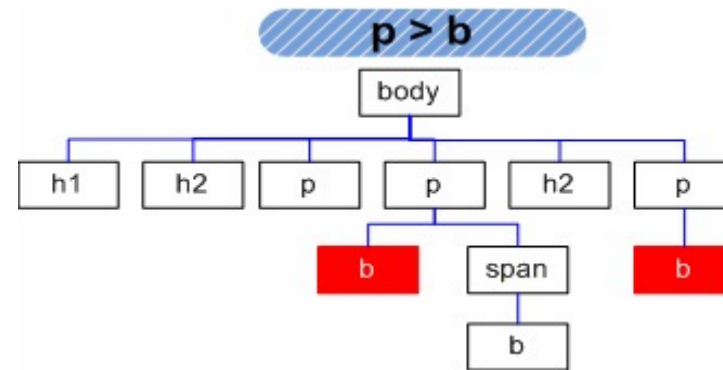


# Example: Selector p b

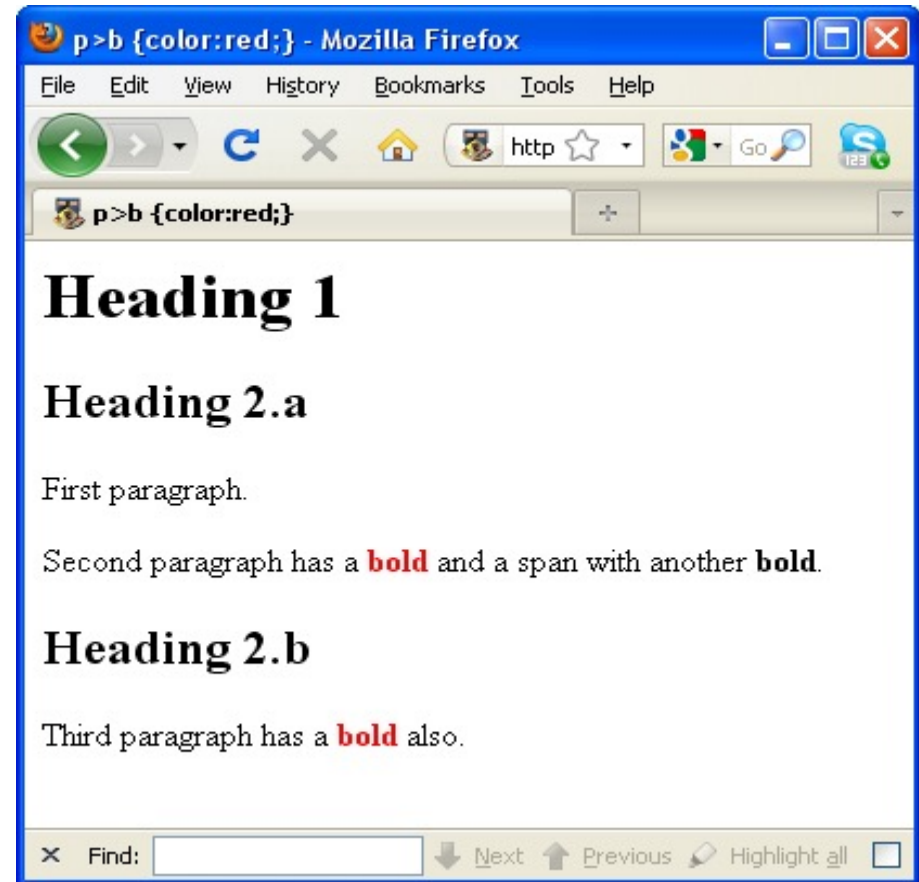
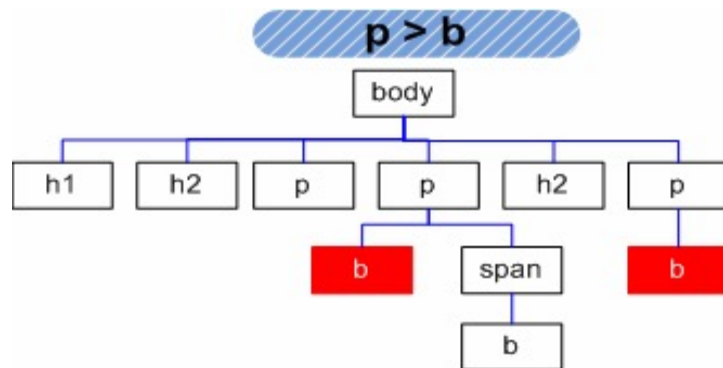


# Example: Selector `p>b`

```
<html>
  <head>
    <style type="text/css">
      p>b {color:red;}
    </style>
  </head>
  <body>
    <h1>Heading 1</h1>
    <h2>Heading 2.a</h2>
    <p>First paragraph. <b>bold</b></p>
    <p>Second paragraph has a <b>bold</b>
      and a <span>span with another <b>bold</b></span>. </p>
    <h2>Heading 2.b</h2>
    <p>Third paragraph has a <b>bold</b> also. </p>
  </body>
</html>
```



# Example: Selector p>b



## Example: Selector h2+p

```
<html>
```

```
  <head>
```

```
    <style type="text/css">
```

```
      h2+p {color:red;}
```

```
    </style>
```

```
  </head>
```

```
  <body>
```

```
    <h1>Heading 1</h1>
```

```
    <h2>Heading 2.a</h2>
```

```
    <p>First paragraph. </p>
```

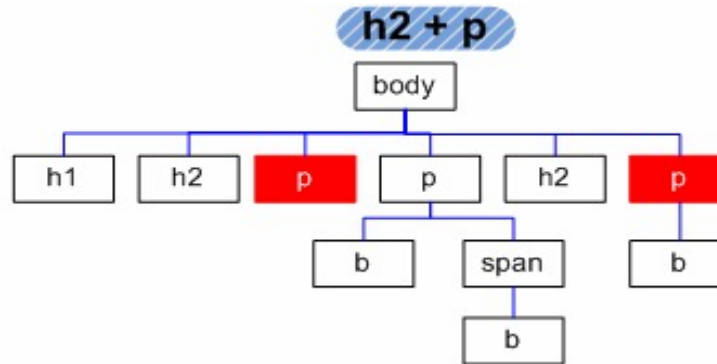
```
    <p>Second paragraph has a <b>bold</b> and a <span>span with  
    another <b>bold</b></span>. </p>
```

```
    <h2>Heading 2.b</h2>
```

```
    <p>Third paragraph has a <b>bold</b> also. </p>
```

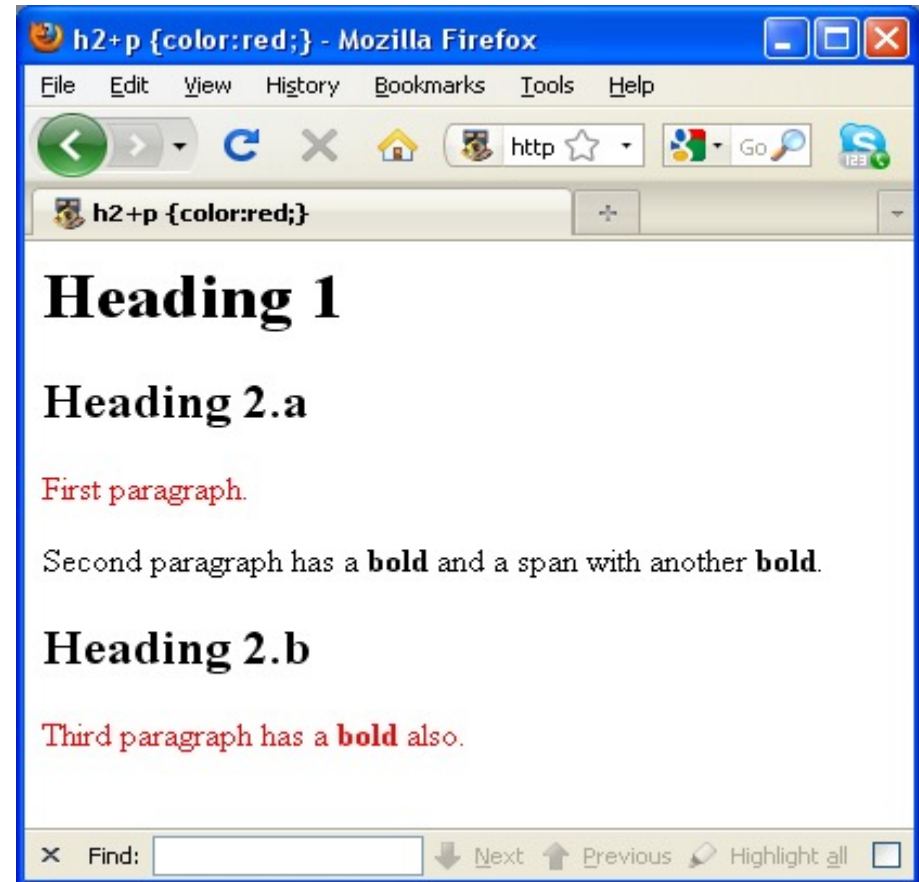
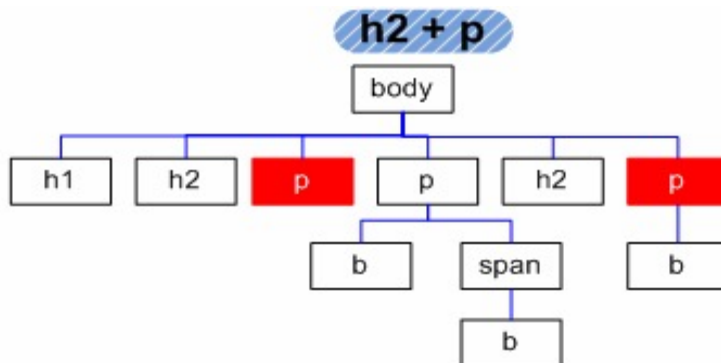
```
  </body>
```

```
</html>
```

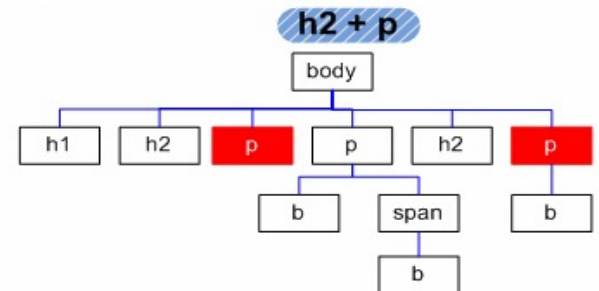
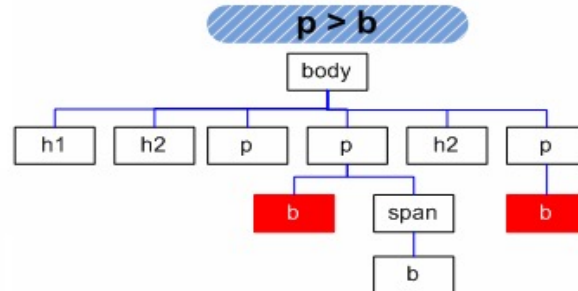
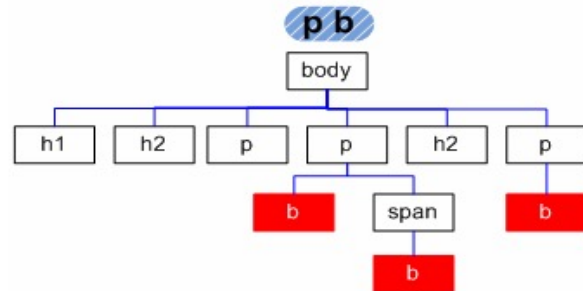
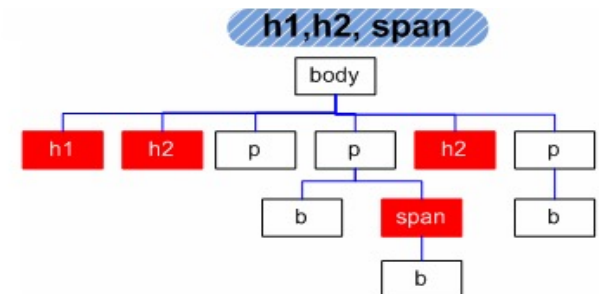
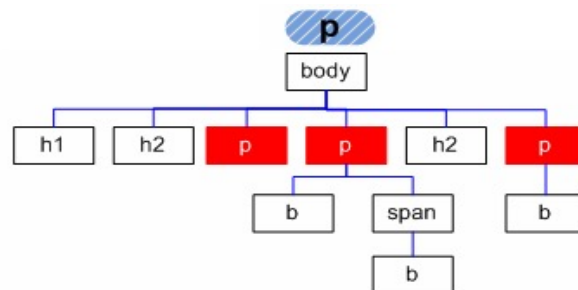
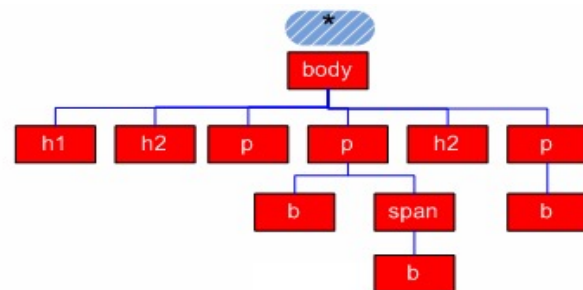




## Example: Selector h2+p



# Examples Summary



# Class Selectors

- Used to allow different occurrences of the same tag to use different style specifications
- A style *class* has a name, which is attached to a tag name

```
p.narrow {property/value list}  
p.wide {property/value list}
```

- The class you want on a particular occurrence of a tag is specified with the class attribute of the tag
- For example:

```
<p class = "narrow">  
    ...  
</p>  
...  
<p class = "wide">  
    ...  
</p>
```

# Generic Selectors

- A *generic class* can be defined if you want a style to apply to more than one kind of tag
- A generic class name must begin with a period
- Example,

```
.really-big {font-size: 60pt; ...}
```

- Use it as if it were a normal style class

```
<h1 class = "really-big"> ... </h1>
```

```
...
```

```
<p class = "really-big"> ... </p>
```

# id Selectors

- An *id* selector allows the application of a style to one specific element

- General form:

```
#specific-id {property/value list}
```

- Example:

```
#breadcrumbs {  
  top: 60px; width: 100%; height: 23px;  
  text-indent: 15px; padding-top: 1px;  
  color: white;  
}
```

```
<p id="breadcrumbs">
```

```
  <a href="http://web.csse.uwa.edu.au/">School Home</a> |
```

```
    <a href="http://web.csse.uwa.edu.au/current/">Current  
      Students</a> |
```

```
    <a href="http://undergraduate.csse.uwa.edu.au/units/  
      CITS4230/">Internet Technologies</a>
```

```
</p>
```



# Pseudo Classes

- Pseudo classes are styles that apply when something happens, rather than because the target element simply exists
- Names begin with colons
  - `hover` classes apply when the mouse cursor is over the element
  - `focus` classes apply when an element has focus

```
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title> Checkboxes </title>
    <style type = "text/css">
      input:hover {color: red;}
      input:focus {color: green;}
    </style>
  </head>
  <body>
    <form action = "">
      <p> Your name:    <input type = "text" />    </p>
    </form>
  </body>
</html>
```

# Conflict Resolution

- When two or more rules apply to the same tag there are rules for deciding which rule applies
- Document level
  - In-line style sheets have precedence over document style sheets
  - Document style sheets have precedence over external style sheets
- Within the same level there can be conflicts
  - A tag may be used twice as a selector
  - A tag may inherit a property and also be used as a selector
- Style sheets can have different sources
  - The author of a document may specify styles
  - The user, through browser settings, may specify styles
- Individual properties can be specified as important

# Precedence Rules

- From highest to lowest
  1. Important declarations with user origin     (`{key: value !important;}`)
  2. Important declarations with author origin
  3. Normal declarations with author origin
  4. Normal declarations with user origin
  5. Any declarations with browser (or other user agent) origin

## Tie-Breakers

- Specificity
  1. id selectors
  2. Class and pseudo-class selectors
  3. Contextual selectors
  4. General selectors
- Position
  - Essentially, later has precedence over earlier

# CSS Properties

- There are many CSS properties and the list is continually growing.
- The basic ones to know are text, background, borders, the box model, colors, tables and lists.

## CSS Property Groups

- Color
- Background and Borders
- Basic Box
- Flexible Box
- Text
- Text Decoration
- Fonts
- Writing Modes
- Table
- Lists and Counters
- Animation
- Transform
- Transition
- Basic User Interface
- Multi-column
- Paged Media
- Generated Content
- Filter Effects
- Image/Replaced Content
- Masking
- Speech
- Marquee

# Font Properties

- `font-size`
  - Possible values: a length number or a name, such as `smaller`, `xx-large`, etc.
- `font-style`
  - `italic`, `oblique` (useless), `normal`
- `font-weight` - degrees of boldness
  - `bolder`, `lighter`, `bold`, `normal`
    - Could specify as a multiple of 100 (100 – 900)
- `font`
  - For specifying a list of font properties
  - `font: bolder 14pt Arial Helvetica`
  - Order must be: style, weight, size, name(s)
- Examples: [fonts.html](#), [fonts2.html](#)
- The `text-decoration` property
  - `line-through`, `overline`, `underline`, `none`
  - `letter-spacing` – value is any length property value

# List properties

- `list-style-type`
- *Unordered lists*
  - Bullet can be a disc (default), a square, or a circle
  - Set it on either the `<ul>` or `<li>` tag
    - On `<ul>`, it applies to list items...

```
<h3> Some Common Single-Engine Aircraft
</h3>
<ul style = "list-style-type: square">
  <li> Cessna Skyhawk </li>
  <li> Beechcraft Bonanza </li>
  <li> Piper Cherokee </li>
</ul>
```

- On `<li>`, `list-style-type` applies to just that item

```
<h3> Some Common Single-Engine Aircraft
</h3>
<ul>
  <li style = "list-style-type: disc">
    Cessna Skyhawk </li>
  <li style = "list-style-type: square">
    Beechcraft Bonanza </li>
  <li style = "list-style-type: circle">
    Piper Cherokee </li>
</ul>
```

## **Some Common Single-Engine Aircraft**

- Cessna Skyhawk
- Beechcraft Bonanza
- Piper Cherokee

## **Some Common Single-Engine Aircraft**

- Cessna Skyhawk
- Beechcraft Bonanza
- Piper Cherokee

# Colors

- The color property specifies the foreground colour of elements

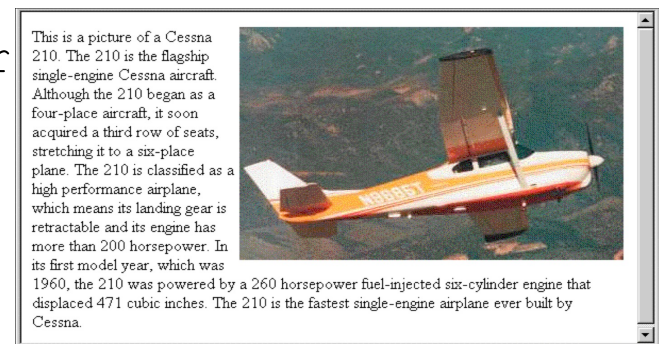
```
<style type = "text/css">
  th.red {color: red}
  th.orange {color: orange}
</style>
...
<table border = "5">
  <tr>
    <th class = "red"> Apple </th>
    <th class = "orange"> Orange </th>
    <th class = "orange"> Screwdriver </th>
  </tr>
</table>
```

Name	Hexadecimal Code	Name	Hexadecimal Code
black	000000	green	008000
silver	C0C0C0	lime	00FF00
gray	808080	olive	808000
white	FFFFFF	yellow	FFFF00
maroon	800000	navy	000080
red	FF0000	blue	0000FF
purple	800080	teal	008080
fuchsia	FF00FF	aqua	00FFFF

- There are three color collections
  - There is a set of 16 colors that are guaranteed to be displayable by all graphical browsers on all color monitors
  - There is a much larger set, the *Web Palette*
    - 216 *named* colors [http://www.w3schools.com/html/html\\_colornames.asp](http://www.w3schools.com/html/html_colornames.asp)
  - Any one of 16 million different colors
    - #000000, #000001, #000002, . . . , #FFFFFFE, #FFFFFFF

# Alignment of Text

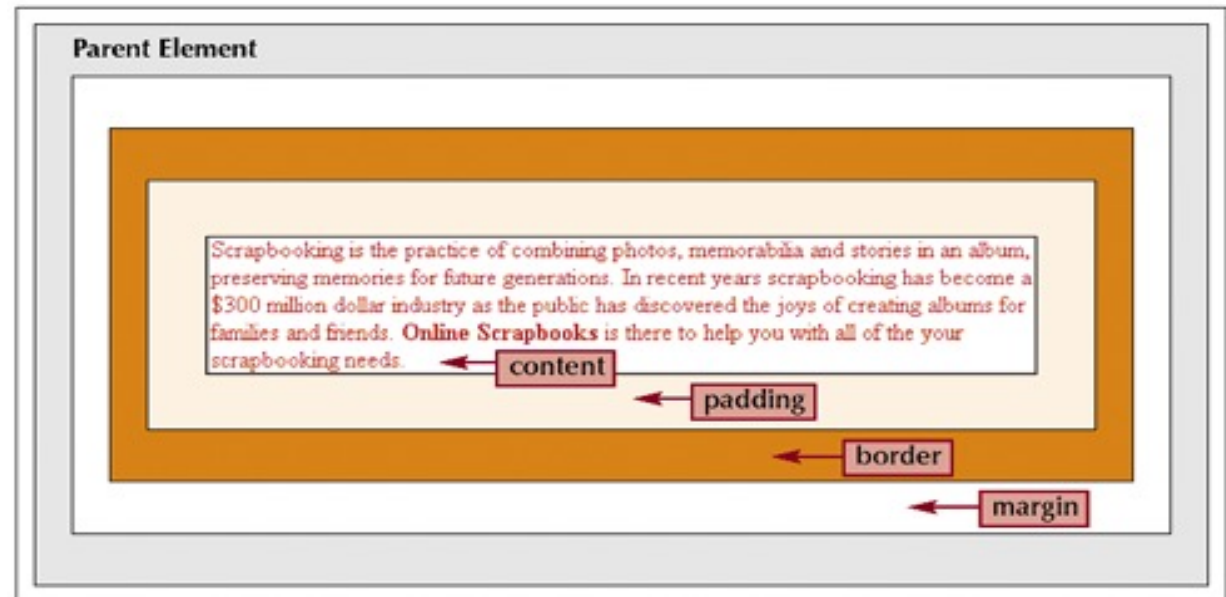
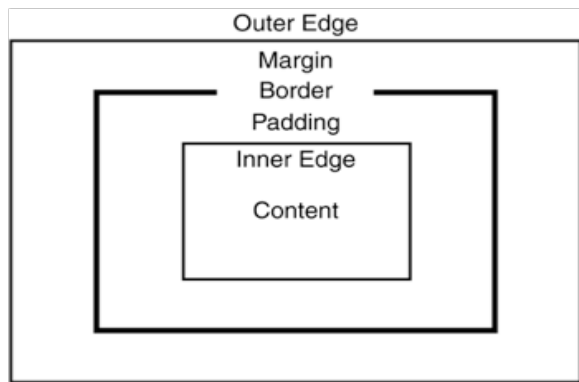
- The `text-indent` property allows indentation
    - Takes either a length or a % value
  - The `text-align` property has the possible values, `left` (the default), `center`, `right`, or `justify`
  - Sometimes we want text to flow around another element - the `float` property
    - The `float` property has the possible values, `left`, `right`, and `none` (the default)
    - If we have an element we want on the right, with text flowing on its left, we use the default `text-align` value (`left`) for the text and the `right` value for `float` on the element we want on the right
- ```
<img src = "c210.jpg" style = "float: right"
```
- Some text with the default alignment - `left`





# Working with the Box Model

- The **box model** is an element composed of four sections:
  - Margin
  - Border
  - Padding
  - content

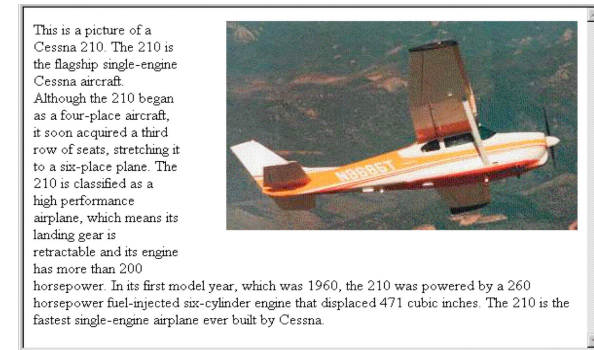


- Borders – every element has a `border-style` property
  - Controls whether the element has a border and if so, the style of the border
    - `border-style` values: none, dotted, dashed, and double
    - `border-width`: thin, medium (default), thick, or a length value in pixels
    - `border-color`: any color
  - → Example: [borders.html](#)

# The Box Model

- Margin – the space between the border of an element and its neighbor element
- The margins around an element can be set with `margin-left`, etc. - just assign them a length value

```
<img src = "c210.jpg " style = "float: right;  
margin-left: 0.35in;  
margin-bottom: 0.35in" />
```



- Padding – the distance between the content of an element and its border
  - Controlled by `padding`, `padding-left`, etc.
- Example: [marpads.html](#)
- The `background-image` property
  - Can also specify [background-image](#)
    - Repetition can be controlled
      - `background-repeat` property
      - Possible values: `repeat` (default), `no-repeat`, `repeat-x`, or `repeat-y`

# The <span> and <div> tags

- One problem with the font properties is that they apply to whole elements, which are often too large
  - Solution: a tag to define an element within a larger element - <span>

- Use <span> to apply a document style sheet to its content

```
<style type = "text/css">
```

```
    .bigred {font-size: 24pt; font-family: Ariel;  
            color: red}
```

```
</style>
```

```
<p>
```

```
    Now is the
```

```
        <span class = "bigred"> best time </span> ever!
```

```
</p>
```



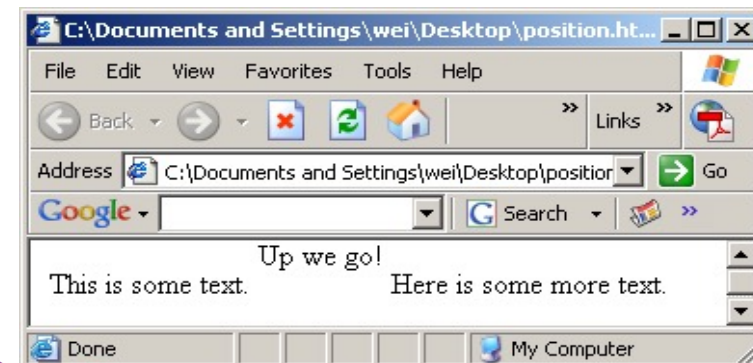
# Positioning

- *Normal Flow* - block formatting of block boxes, inline formatting of inline boxes, relative positioning of block or inline boxes
- *Floats* - laid out according to normal flow, then shifted
- *Absolute positioning* - box is removed entirely from normal flow
- Values: static, relative, absolute, fixed
- Offsets: top, right, left, bottom
- Each CSS box is laid out on the screen (or page) in one of the three ways: in its *normal* position, *relative* position or at an *absolute* position.
- *Relative Positioning*
  - If no `top` and `left` properties are specified, the element is placed exactly where it would have been placed if no `position` property were given
  - But it can be moved later using JavaScript

<p> This is some text.

<span style="position:relative; top: -1em>  
Up we go!</span> Here is some more text.

</p>



# Absolute Positioning

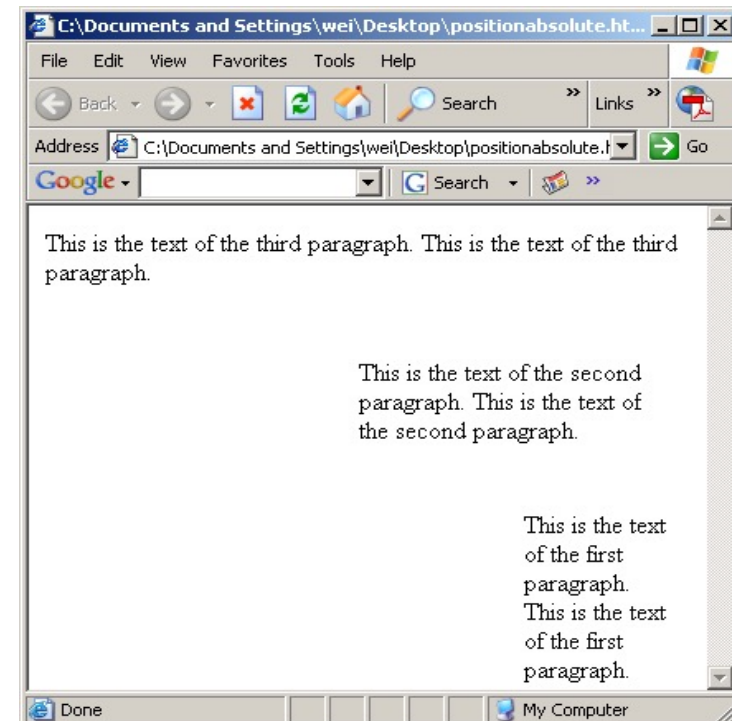
- *Absolute Positioning*

- The element is positioned relative to its first positioned (not static) ancestor

```
<p style = "position: absolute; left: 50px; top: 100px;">
```

- If an element is nested inside another element and is absolutely positioned, the top and left properties are relative to the enclosing element

```
<html><head><style type="text/css">
  .one {position:absolute; top: 200px;
left:300px}
  .two {position:absolute; top: 100px;
left:200px}
  .three {position:static}
</style></head><body>
<p class="one">This is the text of the first
  paragraph. This is the text of the first
  paragraph. </p>
<p class="two">This is the text of the second
  paragraph. This is the text of the second
  paragraph. </p>
<p class="three">This is the text of the third
  paragraph. This is the text of the third
  paragraph. </p>
</body></html>
```









# Vendor Prefix

A positive catalyst for the evolution to exciting technologies

*“... force the vendors and the Working Group to work together to devise the tests necessary to determine interoperability. Those tests can then guide those who follow, helping them to achieve interoperable status much faster. They could literally ship the prefixed implementation in one public beta and drop the prefix in the next.”*

```
.foo {  
-webkit-border-radius: 10px;  
-moz-border-radius: 10px;  
border-radius: 10px;  
}
```

 WebKit	-webkit-
 Mozilla	-moz-
 Opera	-o-
 Konqueror	-khtml-
 Microsoft	-ms-
 Chrome	-chrome-