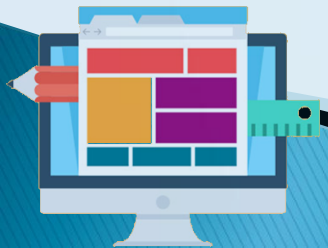
CSS

Designing the world !

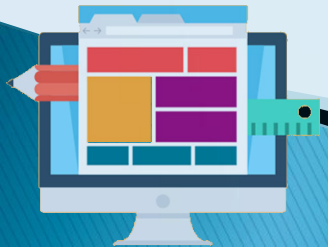
What is CSS

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**



Lecture content

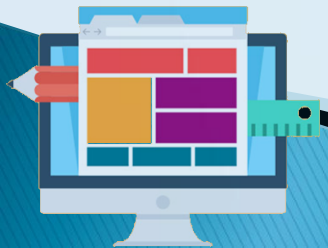
- Linking to the document
- CSS syntax
- Backgrounds
- Borders
- Text
- Box model
- Positioning




CSS
Part 1: Linking

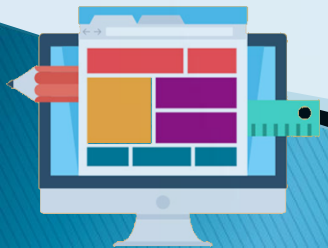
Linking HTML and CSS

- HTML (content) and CSS (presentation) can be linked in three ways:
  - Inline: the CSS rules in the `style` attribute
    - No selectors are needed
  - Internal: in the `<head>` in a `<style>` tag
  - External: CSS rules in separate file (best)
    - Usually a file with `.css` extension
    - Linked via `<link rel="stylesheet" href=…>` tag or `@import` directive in embedded CSS block



Linking HTML and CSS (2)

- Using external files is highly recommended
  - Simplifies the HTML document
  - Improves page load speed as the CSS file is cached



CSS Cascade (Precedence)

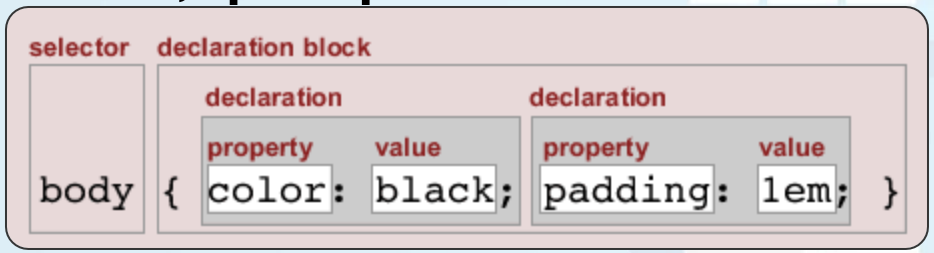
- There are browser, user and author stylesheets with "normal" and "important" declarations
  - Browser styles (least priority)
  - Normal user styles
  - Normal author styles (external, in head, inline)
  - Important author styles
  - Important user styles (max priority)

```
a { color: red !important ; }
```

Part 2: Syntax

Style Sheets Syntax

- Stylesheets consist of rules, selectors, declarations, properties and values



- Selectors are separated by commas
- Declarations are separated by semicolons
- Properties and values are separated by colons

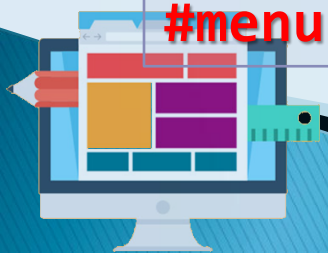
```
h1,h2,h3 { color: green; font-weight: bold; }
```

Selectors

- Selectors determine which element the rule applies to:
  - All elements of specific type (tag)
  - Those that mach a specific attribute (id, class)
  - Elements may be matched depending on how they are nested in the document tree (HTML)
- Examples:

```
.header a { color: green }
```

```
#menu>li { padding-top: 8px }
```



Selectors (2)

- Three primary kinds of selectors:
  - By tag (type selector):

```
h1 { font-family: verdana,sans-serif; }
```

  - By element id:

```
#element_id { color: #ff0000; }
```

  - By element class name (only for HTML):

```
.myClass {border: 1px solid red}
```

- Selectors can be combined with commas:

```
h1, .link, #top-link {font-weight: bold}
```

This will match `<h1>` tags, elements with class `link`, and element with id `top-link`

Selectors (3)

- Pseudo-classes define state
  - **:hover, :visited, :active , :lang**
- Pseudo-elements define element "parts" or are used to generate content
  - **:first-line , :before, :after**

```
a:hover { color: red; }
p:first-line { text-transform: uppercase; }
.title:before { content: "»"; }
.title:after { content: "«"; }
```

Selectors (4)

- Match relative to element placement:

```
p a {text-decoration: underline}
```

This will match all `<a>` tags that are inside of `<p>`

- * – universal selector (avoid or use with care!):

```
* {color: black}
p * {color: black}
```

This will match all descendants of `<p>` element

- + selector – used to match "next sibling":

```
img + .link {float:right}
```

This will match all siblings with class `link` that appear immediately after `<img>` tag

Selectors (5)

- `>` selector – matches direct child nodes:

```
p > .error {font-size: 8px}
```

This will match all elements with class **error**, direct children of **<p>** tag

- **[ ]** – matches tag attributes by regular expression:

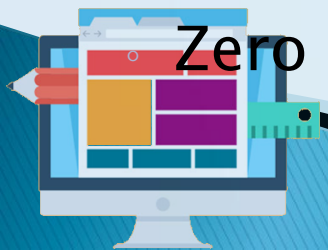
```
img[alt~=logo] {border: none}
```

This will match all **<img>** tags with **alt** attribute containing the word **logo**

- **.class1.class2** (no space) – matches elements with both (all) classes applied at the same time

Values in the CSS Rules

- Colors are set in RGB format (decimal or hex):
  - Example: `#a0a6aa = rgb(160, 166, 170)`
    - Predefined color aliases exist: `black, blue, etc.`
- Numeric values are specified in:
  - Pixels, ems, e.g. `12px , 1.4em`
    - Points, inches, centimeters, millimeters
  - E.g. `10pt , 1in, 1cm, 1mm`
    - Percentages, e.g. `50%`
      - Percentage of what?...
  - Zero can be used with no unit: `border: 0;`



Part 3: Backgrounds

Backgrounds

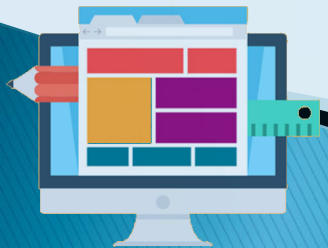
- image to be used as background, e.g.:

```
background-image:url("back.gif");
```

- Using color **background-color**

- Additionaly :

- **background-repeat**
  - **repeat-x, repeat-y, repeat, no-repeat**
- **background-attachment**
  - **fixed / scroll**

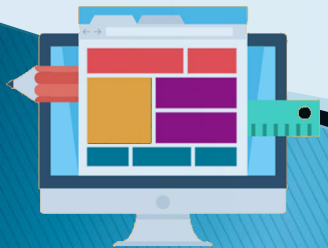


Backgrounds (2)

- **background-position:** specifies vertical and horizontal position of the background image
  - Vertical position: **top, center, bottom**
  - Horizontal position: **left, center, right**
  - Both can be specified in percentage or other numerical values
  - Examples:

```
background-position: top left;
```

```
background-position: -5px 50%;
```



Background Shorthand Property

- background: shorthand rule for setting background properties at the same time:

```
background: #FFF0C0 url("back.gif") no-repeat
fixed top;
```

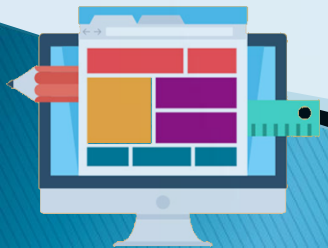
is equal to writing:

```
background-color: #FFF0C0;
background-image: url("back.gif");
background-repeat: no-repeat;
background-attachment: fixed;
background-position: top;
```

  - Some browsers will not apply BOTH color and image for background if using shorthand rule

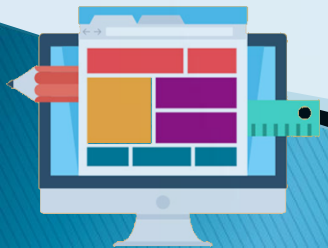
Background-image or <img>?

- Background images allow you to save many image tags from the HTML
  - Leads to less code
  - More content-oriented approach
- <img> allows for more CSS control because it can be used as a selector with all the possible declarations

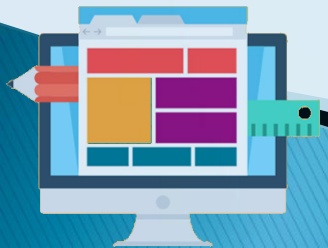


Backgrounds (3): CSS3 Background Size

- The CSS3 `background-size` property allows you to specify the size of background images.
- Values may be `contain, cover` or numerical value



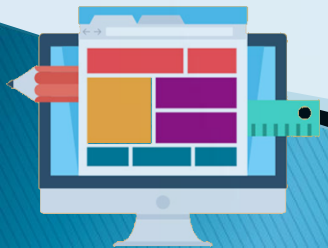

CSS
Part 4: Borders



Borders

- `border-width: thin, medium, thick` or numerical value (e.g. `10px`)
- `border-color`: color alias or RGB value
- `border-style: none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset`

- Each property can be defined separately for left, top, bottom and right
  - `border-top-style, border-left-color, …`



Border Shorthand Property

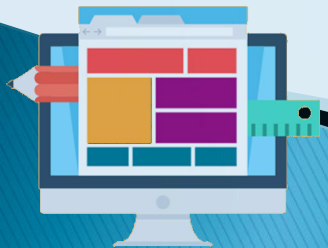
- **border**: shorthand rule for setting border properties at once:

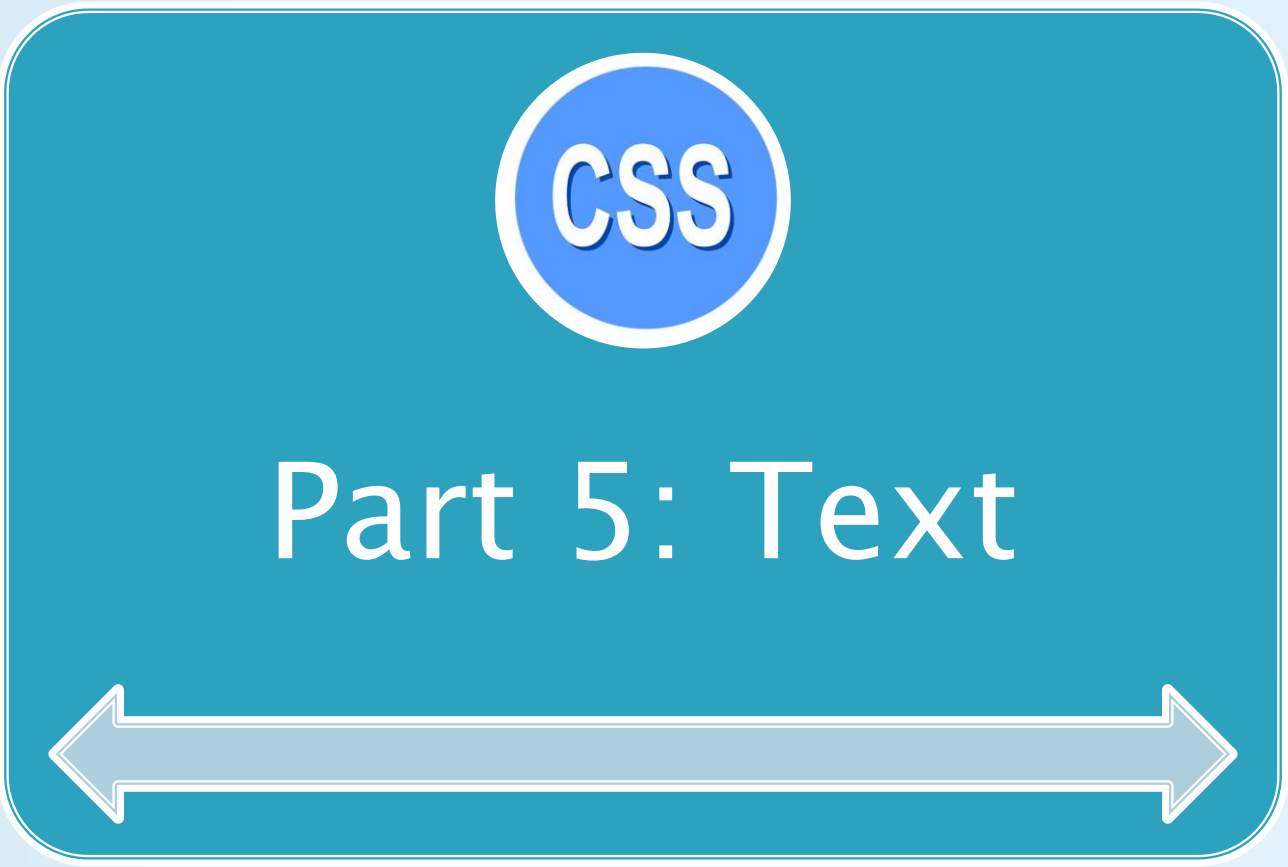
```
border: 1px solid red
```

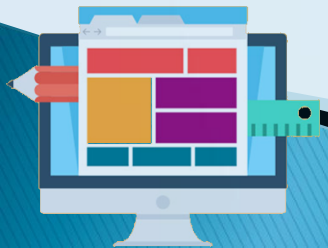
is equal to writing:

```
border-width:1px;
border-color:red;
border-style:solid;
```

- Specify different borders for the sides via shorthand rules: **border-top, border-left, border-right, border-bottom**
- When to avoid **border:0**

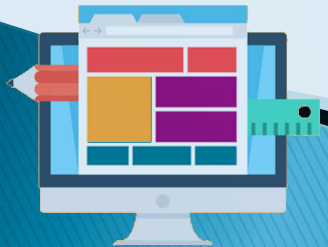



CSS
Part 5: Text



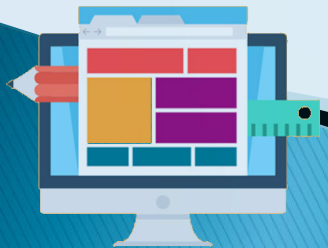
Text-related CSS Properties

- **color** – specifies the color of the text
- **font-size** – size of font: **xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger** or numeric value
- **font-family** – comma separated font names
  - Example: **verdana, sans-serif, etc**.
  - The browser loads the first one that is available
  - There should always be at least one generic font
- **font-weight** can be **normal, bold, bolder, lighter** or a number in range [100 … 900]



CSS Rules for Fonts (2)

- **font-style** – styles the font
  - Values: **normal, italic, oblique**
- **text-decoration** – decorates the text
  - Values: **none, underline, line-through, overline, blink**
- **text-align** – defines the alignment of text or other content
  - Values: **left, right, center, justify**



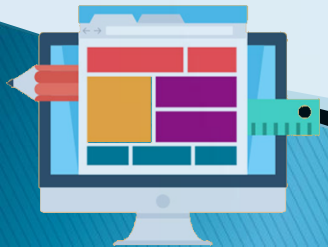
Shorthand Font Property

- font
  - Shorthand rule for setting multiple font properties at the same time

```
font:italic normal bold 12px/16px verdana
```

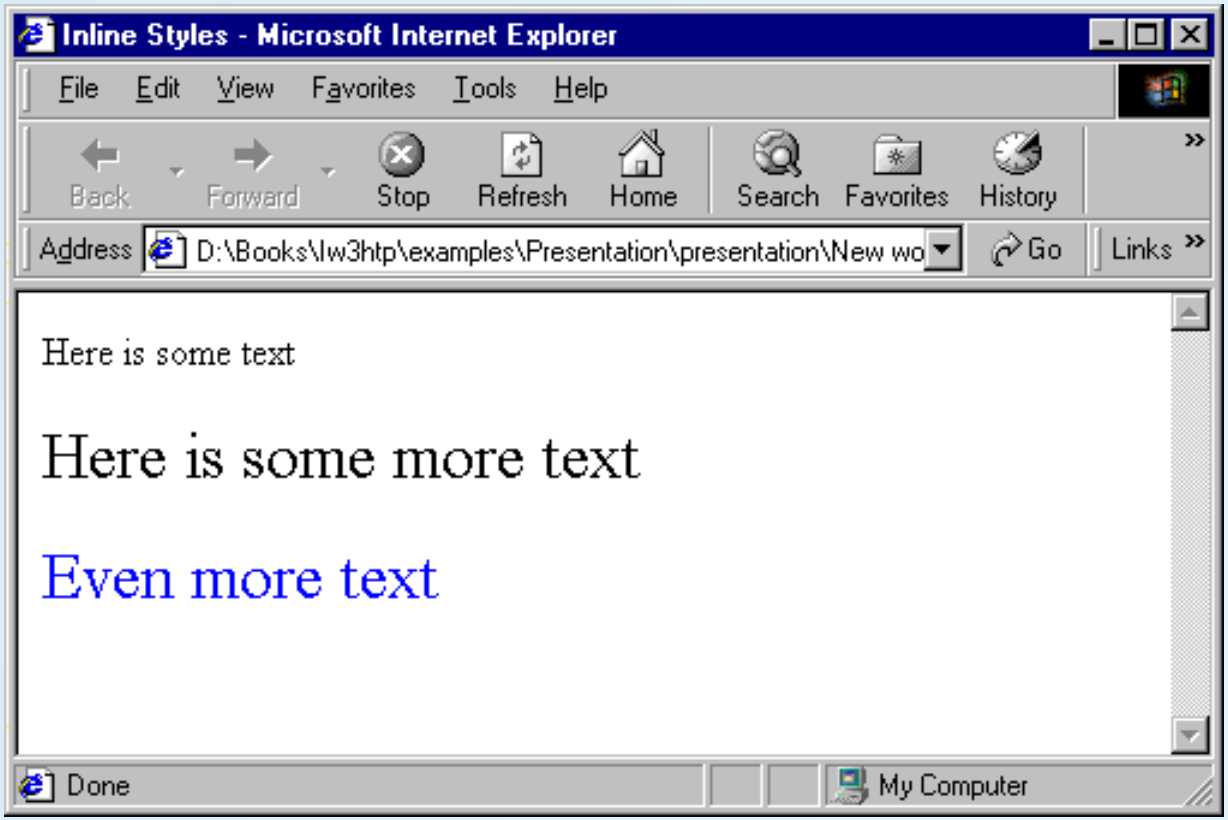
is equal to writing this:

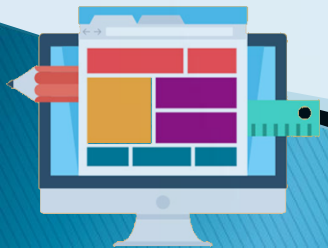
```
font-style: italic;
font-variant: normal;
font-weight: bold;
font-size: 12px;
line-height: 16px;
font-family: verdana;
```



Inline Styles: Example

```
<html>
<head>
  <title>Inline Styles</title>
</head>
<body>
  <p>Here is some text</p>
<p style="font-size:20pt">Here is some
    more text</p>
  <p style="font-size: 20pt;color:
    #0000FF" >Even more text</p>
</body>
</html>
```


Inline Styles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History
Address D:\Books\Iw3htp\examples\Presentation\presentation\New wo
Go Links
Here is some text
Here is some more text
Even more text
Done
My Computer



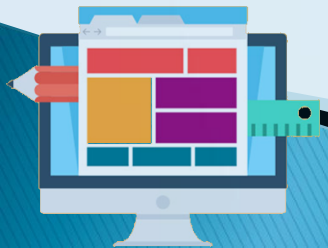
External Styles: Example

styles.css

```
/* CSS Document */

a       { text-decoration: none }

a:hover { text-decoration: underline;
          color: red;
          background-color: #CCFFCC }

li em   { color: red;
          font-weight: bold }

ul      { margin-left: 2cm }

ul ul   { text-decoration: underline;
          margin-left: .5cm }
```

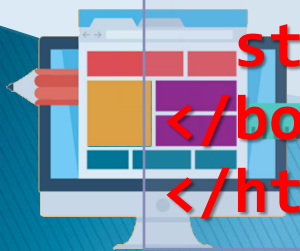
External Styles: Example (2)

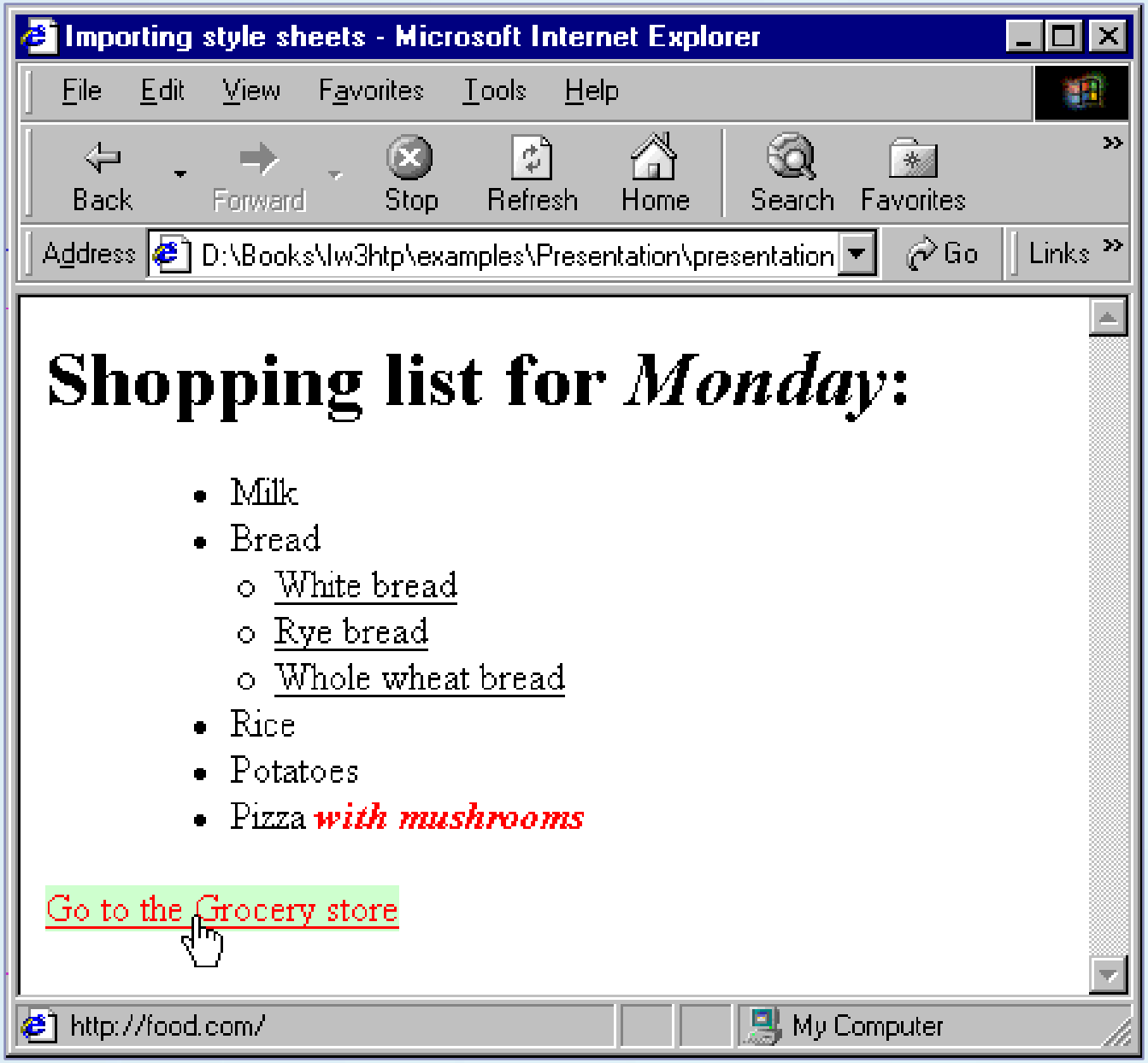
```
<html>
<head>
  <title>Importing style sheets</title>
  <link type="text/css" rel="stylesheet"
    href="styles.css"  />
</head>
<body>
  <h1>Shopping list for <em>Monday</em>:</h1>
  <li>Milk</li>
  …
```

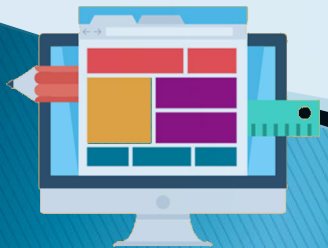


External Styles: Example (3)

```
  …
  <li>Bread
    <ul>
      <li>White bread</li>
      <li>Rye bread</li>
      <li>Whole wheat bread</li>
    </ul>
  </li>
  <li>Rice</li>
  <li>Potatoes</li>
  <li>Pizza <em>with mushrooms</em></li>
</ul>
<a href="http://food.com" title="grocery
    store">Go to the Grocery store</a>
</body>
</html>
```



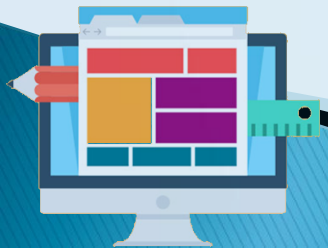

Importing style sheets - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites
Address D:\Books\Iw3htp\examples\Presentation\presentation Go Links
Shopping list for Monday:
Milk
Bread
White bread
Rye bread
Whole wheat bread
Rice
Potatoes
Pizza with mushrooms
Go to the Grocery store
http://food.com/
My Computer



Part 6: box model

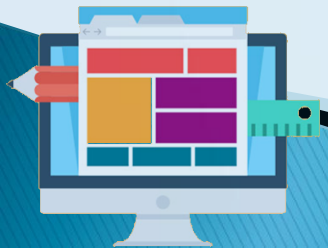
Width and Height

- **`width`** – defines numerical value for the width of element, e.g. **`200px`**
- **`height`** – defines numerical value for the height of element, e.g. **`100px`**
  - By default the height of an element is defined by its content
  - Inline elements do not apply height, unless you change their **`display`** style.



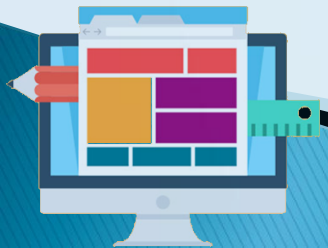
Margin and Padding

- **`margin`** and **`padding`** define the spacing around the element
  - Numerical value, e.g. **`0px`** or **`-5px`**
    - Can be defined for each of the four sides separately – **`margin-top`**, **`padding-left`**, …
    - **`margin`** is the spacing outside of the border
    - **`padding`** is the spacing between the border and the content
    - What are collapsing margins?



Margin and Padding: Short Rules

- **`margin: 5px;`**
  - Sets all four sides to have margin of 5 px;
- **`margin: 10px 20px;`**
  - top and bottom to 10px, left and right to 20px;
- **`margin: 5px 3px 8px;`**
  - top 5px, left/right 3px, bottom 8px
- **`margin: 1px 3px 5px 7px;`**
  - top, right, bottom, left (clockwise from top)
- Same for **`padding`**



The Box Model

Default Browser Styles

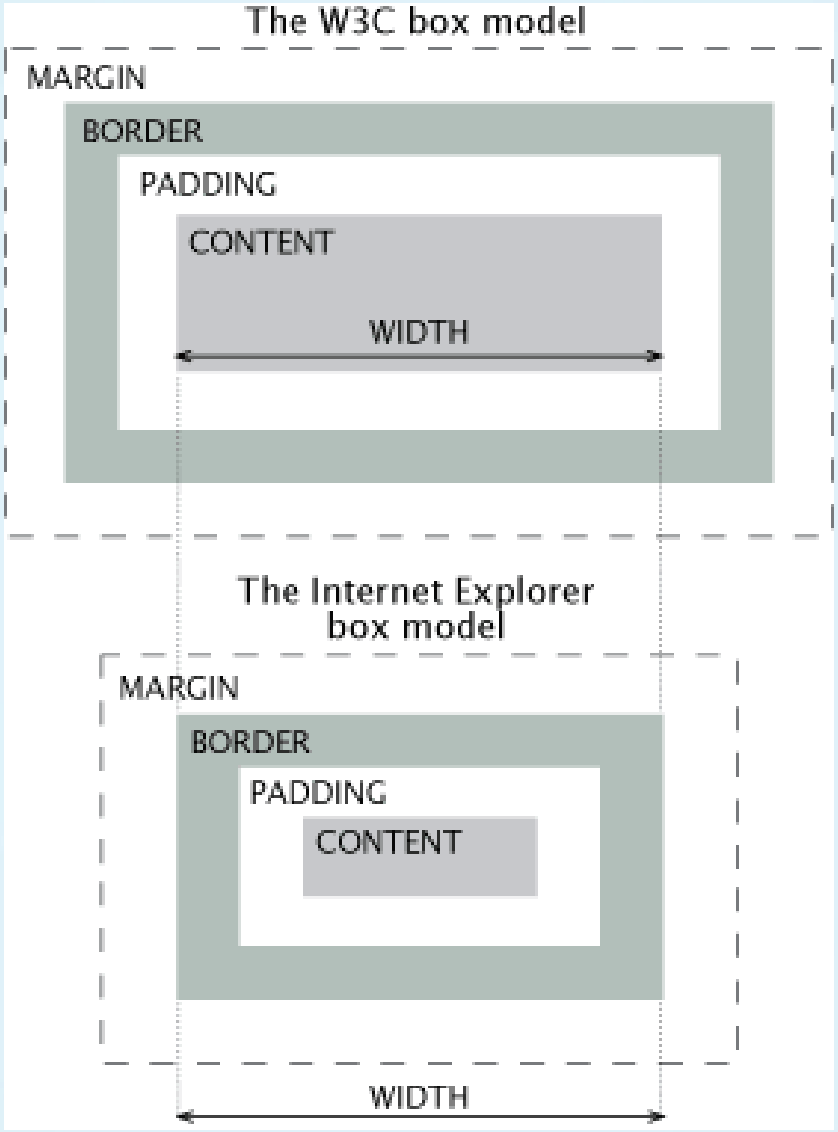
- Browsers have default CSS styles
  - Used when there is no CSS information or any other style information in the document
- Caution: default styles differ in browsers
  - E.g. margins, paddings and font sizes differ most often and usually developers reset them

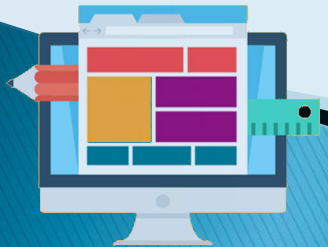
```
* { margin: 0; padding: 0; }
```

```
body, h1, p, ul, li { margin: 0; padding: 0; }
```

IE Quirks Mode

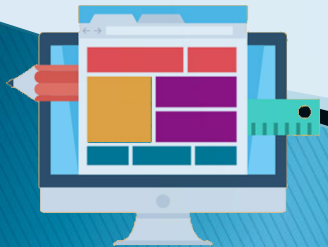
- When using quirks mode (pages with no DOCTYPE or with a HTML 4 Transitional DOCTYPE), Internet Explorer violates the box model standard





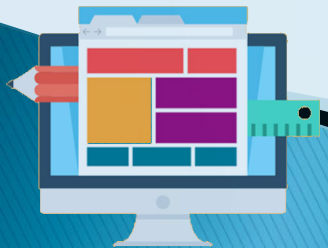
Positioning

- **position:** defines the positioning of the element in the page content flow
- The value is one of:
  - **static** (default)
  - **relative** – relative position according to where the element would appear with static position
  - **absolute** – position according to the innermost positioned parent element
  - **fixed** – same as absolute, but ignores page scrolling



Positioning (2)

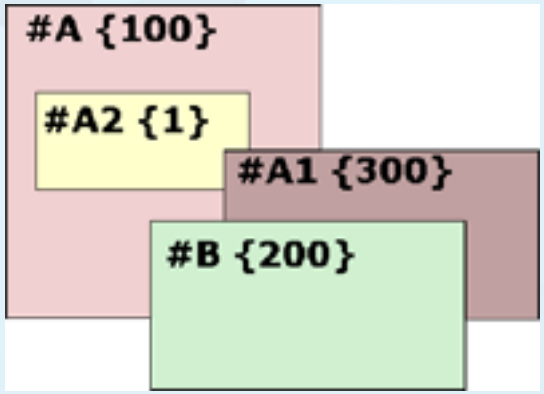
- Margin VS relative positioning
- Fixed and absolutely positioned elements do not influence the page normal flow and usually stay on top of other elements
  - Their position and size is ignored when calculating the size of parent element or position of surrounding elements
  - Overlaid according to their z-index
  - Inline fixed or absolutely positioned elements can apply height like block-level elements

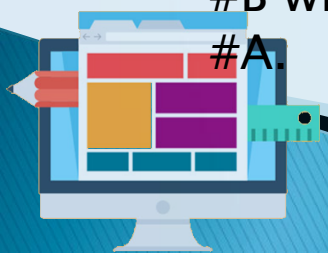


Positioning (3)

- **top, left, bottom, right**: specifies offset of absolute/fixed positioned element as numerical values
- **z-index** : specifies the stack level of positioned elements
  - Understanding stacking context

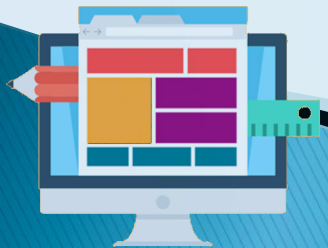
Each positioned element creates a stacking context.
Elements in different stacking contexts are overlapped according to the stacking order of their containers. For example, there is no way for #A1 and #A2 (children of #A) to be placed over #B without increasing the z-index of #A.





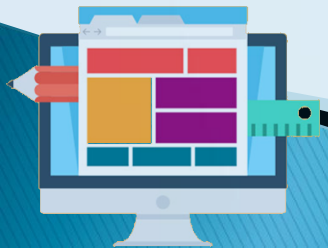
Inline element positioning

- `vertical-align`: sets the vertical-alignment of an inline element, according to the line height
  - Values: `baseline, sub, super, top, text-top, middle, bottom, text-bottom` or numeric
- Also used for content of table cells (which apply `middle` alignment by default)



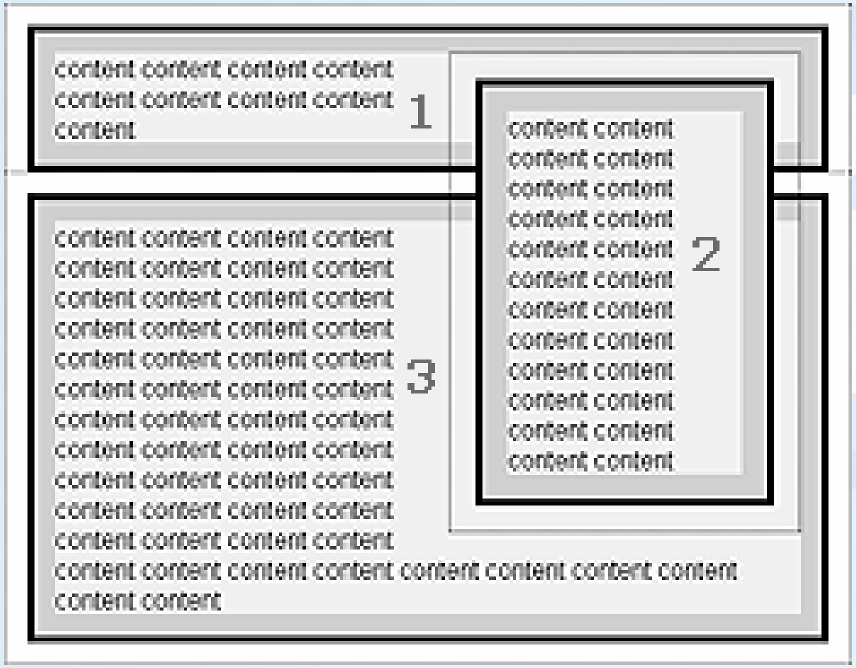
Float

- `float`: the element "floats" to one side
  - `left:` places the element on the left and following content on the right
  - `right:` places the element on the right and following content on the left
  - floated elements should come before the content that will wrap around them in the code
  - margins of floated elements do not collapse
  - floated inline elements can apply height



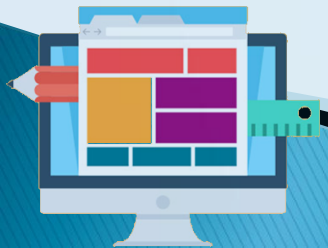
Float (2)

- How floated elements are positioned



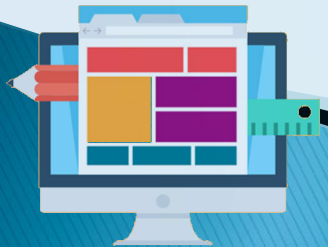
Clear

- **clear**
  - Sets the sides of the element where other floating elements are NOT allowed
  - Used to "drop" elements below floated ones or expand a container, which contains only floated children
- Possible values: **left, right, both**
- Clearing floats
  - additional element **( )** with a clear style



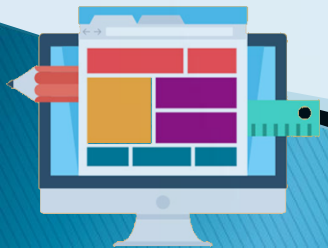
Clear (2)

- Clearing floats (continued)
  - **:after { content: ""; display: block; clear: both; height: 0; }**
    - Triggering **hasLayout** in IE expands a container of floated elements
      - display: inline-block;
      - zoom: 1;



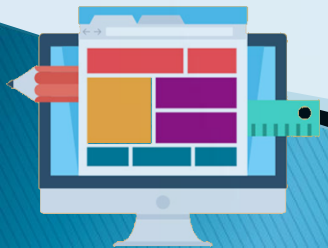
Opacity

- **opacity**: specifies the opacity of the element
  - Floating point number from 0 to 1
  - For old Mozilla browsers use **-moz-opacity**
  - For IE use **filter:alpha(opacity=value)** where value is from **0 to 100**; also, "binary and script behaviors" must be enabled and **hasLayout** must be triggered, e.g. with zoom:1



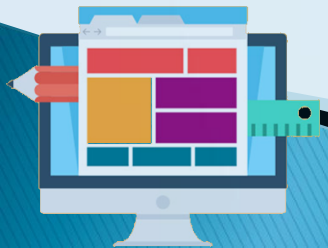
Visibility

- **visibility**
  - Determines whether the element is visible
  - **hidden:** element is not rendered, but still occupies place on the page (similar to **opacity:0**)
  - **visible:** element is rendered normally



Display

- **display:** controls the display of the element and the way it is rendered and if breaks should be placed before and after the element
  - **inline:** no breaks are placed before and after (is an inline element)
  - **block:** breaks are placed before AND after the element (** ** is a block element)



Display (2)

- **`none:`** element is hidden and its dimensions are not used to calculate the surrounding elements rendering (differs from `visibility: hidden`!)
- There are some more possible values, but not all browsers support them