PROGETTAZIONE E PRODUZIONE MULTIMEDIALE

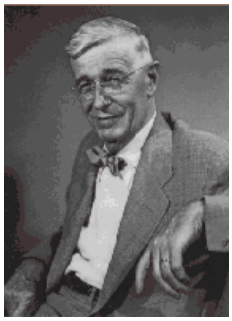
HTML, CSS

Prof. Alberto Del Bimbo

The HTML

HTML stands for *HyperText Markup Language*, i.e. a formatting language that use markers to specify how a document should appear and its relationships with other documents.

Brief history of HTML



- The origin of HTML can be dated back to the original ideas of the US engineer Vannevar Bush. In the early 30's, he described a system to interconnect information stored in microfilms according to the needs of the individual users.
- The system – never implemented – was called *Memex* (*mem*ory *ex*tension), and was aimed to support humans against the growth of information.

- In 1965, Ted Nelson used the term *Hypertext* to refer to a text connected to other bulks of information according to Bush's ideas. The name indicated that that the new text extended the information capability of a plane text through the links to other documents

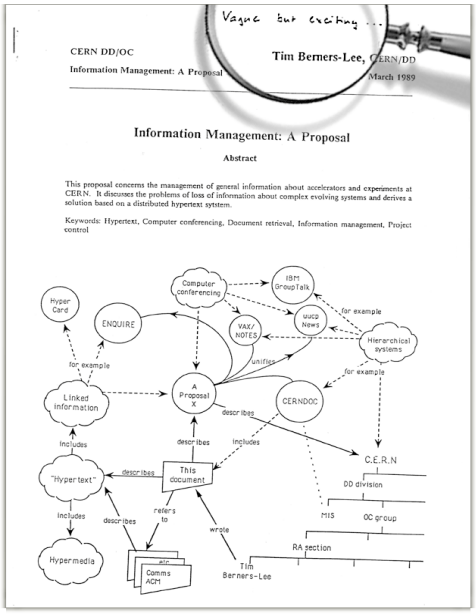
- In 1980 Tim Berners-Lee, a researcher at CERN in Geneve, following the ideas of Bush and Nelson, proposed a system based on hypertext to share information between the researchers of the institute and developed a prototype named *Enquire*.
- Later on, in 1989, he had the idea of joining hypertext with the Internet transmission protocol TCP and specified HTML as a markup language to create web pages.

"…..HyperText is a way to link and access information of various kinds as a web of nodes in which the user can browse at will. It provides a single user-interface to large classes of information (reports, notes, data-bases, computer documentation and on-line help). We propose a simple scheme incorporating servers already available at CERN.

The project has two phases: firstly we make use of existing software and hardware as well as implementing simple browsers for the user's workstations, based on an analysis of the requirements for information access needs by experiments. Secondly, we extend the application area by also allowing the users to add new material.

Phase one should take 3 months with the full manpower complement, phase two a further 3 months, but this phase is more open-ended, and a review of needs and wishes will be incorporated into it.

The manpower required is 4 software engineers and a programmer, Each person will require a state-of-the-art workstation. These will cost …. totalling 50k… and foresee an expense of 30k during development for one-user licences, visits to existing installations and consultancy……."



- His proposal at CERN was accepted. This started the story of Web and HTML.
- He wrote the browser and server software in late 1990. The first website built was first put online at CERN on 6 August 1991

- HTML became of public domain with the X-Windows Mosaic browser in 1993, developed by NCSA (National Center for Supercomputing Applications) under the lead of Marc Andersen who later started Netscape.
- In 1993 Tim Berners Lee left CERN and joined MIT where he founded the World Wide Web Consortium (W3C) in cooperation between MIT and CERN. From 1994 on, any modifications of HTML have been defined under the control of the W3C.
- Today the consortium includes the main IT research centers and companies worldwide and has standardized HTML as the language for the Web

HTML syntax

Tags

- Content definition of a HTML document is obtained using *tags*.
  According to this a HTML page contains a set of tags with different names depending on the funtion that each of them must play.

Tags are identified with a tag name enclosed in special parentheses `<TAG>` followed by content. Tags are closed with `"/"`: `</TAG>`.

Tag attributes and content

- Tags have *attributes.* Attributes have *values. Tag content* is included between the start and the end of a tag

Tag syntax:
<TAG attributes>content</TAG>.
Attribute syntax:
attribute= "value" "" no more required in HTML5

Example (right alignment of text) :
```
<P align="right">testo</P>
```

- General syntax of a tag:
<TAG attributo_1="valore1" attributo_2="valore2"…> contenuto</TAG>

- Some tags have no content. For example this happens when the tag indicates the position of elements such as images. These tags have no closure as well.
They are referred to as empty tags.

Empty tag syntax:
<TAG attributes>.

Example of empty tag for images:
```
<IMG width="20" height="20" SRC="miaImmagine.gif" ALT="alt">
```

- Tags can be nested:

```
<TAG1 attributi>
     contenuto 1
     <TAG2>
          contenuto 2
     </TAG2>
 </TAG1>
```

- Nested tags are useful for text formatting:

```
<P align="right">
     testo 1
    <P align="left">
     testo 2
    </P>
 </P>
```

- HTML is *case unsensitive*.
  `<P ALIGN="RIGHT">` in HTML is regarded as the same as `<p align="right”>`

Comments

- Comments can be included to make the HTML document more readable

Comment syntax is as follows:

```
<!- this is a comment -->
```

The <!DOCTYPE> instruction and the <html> tag

- **<!DOCTYPE ….>.** Is the first line of any HTML document. It is used to tell the browser the type of document it is reading. `Doctype` stands for Document Type Declaration (DTD) and includes declarations of:
  - the language used (HTML) and its version (ex. 4.0)
  - the world language (ex.EN)
  - Whether the document is public
- **<!DOCTYPE>** is not an HTML tag but instead an instruction for the browser
- Following the **<!DOCTYPE>** the **<html>** tag tells the browser that this is an HTML document. The <html> tag represents the root of an HTML document and is the container for all other HTML elements (except for the **<!DOCTYPE>**).

HTML document structure

- An HTML document is organized into two distinct sections :
  - Head `<head>`
    includes any information that regards how the document must be read and interpreted. Namely includes meta-tags for search engines, scripts in JavaScript, stylesheets, …
  - Body `<body>`
    contains the document contents

Example of HTML document:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>

<head>
 <meta http-equiv="Content-Type"content="text/html;charset=iso8859-1">
 <title>HTML.it</title>
</head>

<body>
  <!– qui il contenuto del documento -->
  Qui il nostro contenuto
</body>

</html>
```

Head

Tag <meta> in the head tells the browser to load the western char set

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
```

Tag <title> in the head indicates the name of the page that will appear in the browser bar

```
<title>Nome del sito</title>
```

Body

bgcolor attribute of tag body sets the background color :

```
<body bgcolor="blue">
```

Images can be used as background using the **background** attribute:

```
<body background="imgSfondo.gif">
```

the background image is repeated in both horizontal and vertical directions

background images and color can be combined:

```
<body bgcolor="#0000ff"  background="imgSfondo.gif">
```

- Color can be selected using hexadecimal coding:

```
<body bgcolor="#0000FF">
```

Hexcodes and colors:

Color Name	RGB Triplet	Hexadecimal	Color Name	RGB Triplet	Hexadecimal
Aqua	(0,255,255)	00FFFF	Navy	(0,0,128)	000080
Black	(0,0,0)	000000	Olive	(128,128,0)	808000
Blue	(0,0,255)	0000FF	Purple	(128,0,128)	800080
Fuchsia	(255,0,255)	FF00FF	Red	(255,0,0)	FF0000
Gray	(128,128,128)	808080	Silver	(192,192,192)	C0C0C0
Green	(0,128,0)	008000	Teal	(0,128,128)	008080
Lime	(0,255,0)	00FF00	White	(255,255,255)	FFFFFF
Maroon	(128,0,0)	800000	Yellow	(255,255,0)	FFFF00

The browser's margin does not exactly fits with page margins. To eliminate borders the **leftmargin** and **topmargin** body attributes can be used:

```
<body leftmargin="0" topmargin="0">
```

The **lang** body attribute indicates the language used to the browser and search engines :

```
<body lang="it">
```

- Document example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>

<head>
   <meta http-equiv="Content-Type"content="text/html;charset=iso-88591">
   <title>HTML.it</title>
</head>

<body leftmargin="0" topmargin="0" background="imgs/sfondo00006.gif"
      bgcolor="#66CCFF" lang="it">
     Testo di prova
</body>

</html>
```

Header tag

- Six different levels of header:

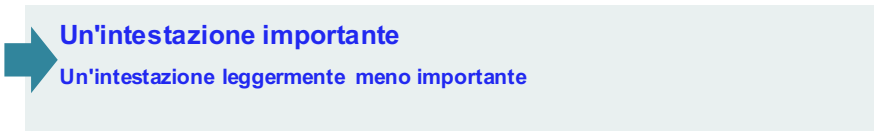
h1 is the most important, followed by **h2** and so on until **h6**

The highest level header:

```
<h1>Un'intestazione importante</h1>
```

An immediately less important header:

```
<h2>Un'intestazione leggermente meno importante</h2>
```



Paragraph tag

Paragraphs start with tag `<p>` and end with `</p>`.

```
<p>Questo è il primo paragrafo.</p>
<p>Questo è il secondo paragrafo.</p>
```

Line breaks are made with the empty tag `<br>`

```
<p>Questo è il terzo paragrafo.<br> Adesso siamo a capo di una riga </p>
```

Un'intestazione importante

Un'intestazione leggermente meno importante

Questo è il primo paragrafo.

Questo è il secondo paragrafo.

Questo è il terzo paragrafo.
Adesso siamo a capo di una riga

Logical character style tags

- Words are emphasized with special tags:

`<strong>` and `<b>` make text bold

`<em>` and `<i>` make text italics

`<strong>` is used with a different meaning in HTML5

```
<p>Con il testo in grassetto, <strong>voglio attirare
l'attenzione</strong> mentre di seguito faccio una citazione
<em>"Thinking HTML"</em></p>
```

Un'intestazione importante

Un'intestazione leggermente meno importante

Questo è il primo paragrafo.

Questo è il secondo paragrafo.

Questo è il terzo paragrafo.
Adesso siamo a capo di una riga

Con il testo in grassetto, **voglio attirare l'attenzione** mentre di seguito faccio una citazione *"Thinking HTML"*

```
<EM>Emphasized</EM><BR>
<STRONG>Strongly Emphasized</STRONG><BR>
<CODE>Code</CODE><BR>
<SAMP>Sample Output</SAMP><BR>
<KBD>Keyboard Text</KBD><BR>
<DFN>Definition</DFN><BR>
<VAR>Variable</VAR><BR>
<CITE>Citation</CITE><BR>
<EM><CODE>Emphasized Code</CODE></EM><BR>
<FONT COLOR="GRAY"><CITE>Gray Citation</CITE></FONT><BR>
<ACRONYM TITLE="JavaDevelopmentKit">JDK Acronym</ACRONYM>
```

Emphasized
Strongly Emphasized
`Code`
`Sample Output`
`Keyboard Text`
Definition
Variable
Citation
`Emphasized Code`
Gray Citation
JDK Acronym

Physical character style tags

```
<B>Bold</B><BR>
<I>Italic</I><BR>
<TT>Teletype (Monospaced)</TT><BR>
<U>Underlined</U><BR>
Subscripts: f<SUB>0</SUB> + f<SUB>1</SUB><BR>
Superscripts: x<SUP>2</SUP> + y<SUP>2</SUP><BR>
<SMALL>Smaller</SMALL><BR>
<BIG>Bigger</BIG><BR>
<STRIKE>Strike Through</STRIKE><BR>
<B><I>Bold Italic</I></B><BR>
<BIG><TT>Big Monospaced</TT></BIG><BR>
<SMALL><I>Small Italic</I></SMALL><BR>
<FONT COLOR="GRAY">Gray</FONT><BR>
```

Bold
Italic
`Teletype (Monospaced)`
Underlined
Subscripts: $f_0 + f_1$
Superscripts: $x^2 + y^2$
Smaller
Bigger
~~Strike Through~~
Bold Italic
`Big Monospaced`
Small Italic
Gray

Still formally valid for compatibility with former HTML versions but no longer in use with HTML5

List tag

- Lists supported in HTML are of three different types:

Non-ordered lists (pointed items) use tags `<ul>` `</ul>` and `<li>` for list and list items respectively

Example :

```
<ul>
  <li>il primo elemento della lista</li>
  <li>il secondo elemento della lista</li>
  <li>il terzo elemento della lista</li>
</ul>
```

Fruit
- Banana
- Grape

Ordered lists (numbered items) use tags `<ol>` `</ol>` and `<li>` for list and list items respectively

Example:

```
<ol>
  <li>il primo elemento della lista</li>
  <li>il secondo elemento della lista</li>
  <li>il terzo elemento della lista</li>
</ol>
```

Fruit
1. Banana
2. Grape

Lists that specify list items and their definition use tags `<dl> </dl>`,`<dt> </dt>` and `<dd> </dd>` respectively for list, items and item definition

```
<dl>
<dt>il primo termine</dt> <dd>la sua definizione</dd>
<dt>il secondo termine</dt> <dd>la sua definizione</dd>
<dt>il terzo termine</dt> <dd>la sua definizione</dd>
</dl>
```

Table tag

- Tables are document components that are used for structured presentation of data.

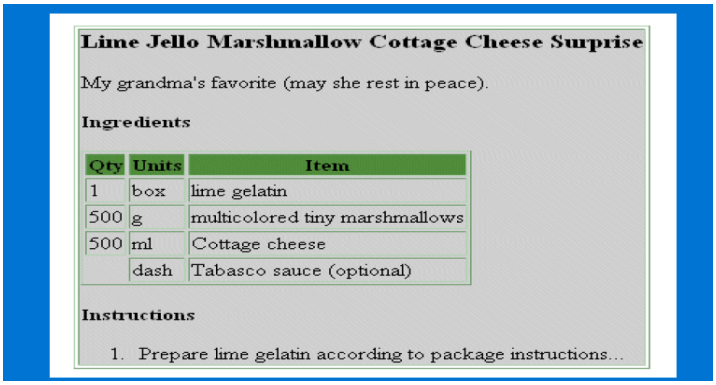
Tables are defined using tag `<table>`.
Tags `<tr></tr>`, `<th></th>`, `<td></td>` are used to define respectively rows, headers and data.

- Attribute *`border`* permits to align table borders to page borders. It either fixes the table width or let the browser to perform width alignment with table size adaptation

Example:

```
<table border="1">
   <tr><th>Anno</th><th>Vendite</th></tr>
   <tr><td>2000</td><td>$18M</td></tr>
   <tr><td>2001</td><td>$25M</td></tr>
   <tr><td>2002</td><td>$36M</td></tr>
   </table>
```

- Table example:



Lime Jello Marshmallow Cottage Cheese Surprise

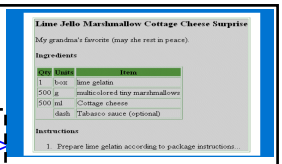
My grandma's favorite (may she rest in peace).

Ingredients

Qty	Units	Item
1	box	lime gelatin
500	g	multicolored tiny marshmallows
500	ml	Cottage cheese
	dash	Tabasco sauce (optional)

Instructions

1. Prepare lime gelatin according to package instructions...



```
<HTML>
 <HEAD>
  <TITLE>Lime Jello Marshmallow Cottage Cheese Surprise</TITLE>
 </HEAD>
 <BODY>
  <H3>Lime Jello Marshmallow Cottage Cheese Surprise</H3>
       <P>My grandma's favorite (may she rest in peace).</P>
  <H4>Ingredients</H4>
  <TABLE BORDER="1">
  <TR BGCOLOR="#308030"><TH>Qty</TH><TH>Units</TH><TH>Item</TH></TR>
          <TR><TD>1</TD><TD>box</TD><TD>lime gelatin</TD></TR>
          <TR><TD>500</TD><TD>g</TD><TD>multicolored tiny marshmallows</TD></TR>
          <TR><TD>500</TD><TD>ml</TD><TD>cottage cheese</TD></TR>
          <TR><TD></TD><TD>dash</TD><TD>Tabasco sauce (optional)</TD></TR>
  </TABLE>
  <H4>Instructions</H4>
        <OL>
          <LI>Prepare lime gelatin according to package instructions...</LI>
          <!-- and so on -->
        </OL>
 </BODY>
</HTML>
```

Note: there is no data type support, so types of data are not specified. Elements in the first column are not quantities but only free text

- Table tag attributes:
  - Internal cell spacing, the *distance between cell external margin and cell content*, is applied through the ***cellpadding*** attribute.

Example (10 pixel internal cell spacing):

```
<table border="1" cellpadding="10">
```

  - The ***cellspacing*** attribute sets spacing between cells.

Example (between cell spacing 10 pixel):

```
<table border="1" cellspacing="10">
```

Still formally valid for compatibility with former HTML versions but **no longer in use with HTML5**

- As a default, browsers set cell header as centered and align left cell data. Default alignments are modified using the ***align, valign***, attributes for individual cells and rows.
- The ***bgcolor*** attribute define the row background color

```
Align: left, center, right
Valign: top, middle, bottom
Bgcolor: background color
```

- Example:


```
<TABLE ALIGN="center" WIDTH="300" HEIGHT="200">
<TR ALIGN="left" VALIGN="top" BGCOLOR="red"><TD>One</TD><TD>Two</TD>
<TR ALIGN="center" VALIGN="middle" BGCOLOR="lightblue"><TD>Three</TD><TD>Four</TD>
<TR ALIGN="right" VALIGN="bottom" BGCOLOR="yellow"><TD>Five</TD><TD>Six</TD>
</TABLE>
```

Still formally valid for compatibility with former HTML versions but no longer in use with HTML5

Image tag

Images are specified using tag **<img>**

Image attributes **width** and **height** specify respectively image width and height in pixels

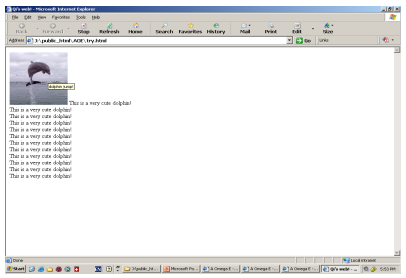
Src attribute is used to load the image file.
Alt attribute is used to associate some text/audio description to the image so that also visually disabled people can have some feeling of image content

Example (image "libro.jpg" in the same directory as the HTML with 200 pixel width and 150 pixel height.

```
<img src = "libro.jpg" width="200" height="150"
 alt="Il Dizionario Sabatini-Colletti">
```

- Alignment of images and text in HTML documents can be modified using ***align*** attribute:
  - ***left*** image on left edge; text flows to right of image
  - **middle** words align with middle of image
  - **right** image on right edge; text flows to left
  - **top** image is left; words align with top of image
  - **bottom** image is left; words align with bottom of image

```
<BODY>
    <img src="dolphin.jpg" align="left" width="150" height="150" alt="dolphin jump!">
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
   This is a very cute dolphin on the left!<br>
    </BODY>
</HTML>
```



Still formally valid but no longer in use with HTML5

Div tag

`<Div>` tag is used for blocks of text, e.g., paragraphs, block quotes, headers, or lists. Blocks are defined when a specific formatting option must be applied to that part of the document. The formatting option is typically defined using the ***class*** attribute of Cascade Style Sheet (**see CSS in the following**) :

```
<Div class=class_name>
    Block-level elements
</Div>
```

`Class_name` is the name of the class.

Span tag

The `<span>` tag provides a way to add a hook to a part of a text or a part of a document.

It can be used to group elements for styling purposes (using the ***class*** or ***id*** or ***style*** attributes of CSS explained later). It should be used only when no other semantic element is appropriate. `<span>` is very much like a `<div>` element, but `<div>` is block-level element whereas a `<span>` is typically an *inline* element (see later).

Example:

```
<p><span>Some text</span></p>
```

Results:

- If nothing specified in the CSS: Some text
- if `p span {color: blue}` specified in the CSS : **Some text**

Multimedia content

- Multimedia content can be distinguished between internal and external.
  - External contents require additional software to be displayed, that is not offered with the browser. External contents are MPEG videos, audio sources…
    Native in HTML5
  - Internal contents are directly shown by the browser. Typical internal contents are animated GIFs, *Java applets*….

HTML5 defines new elements (the `<audio>` and the `<video>` element) which specify a standard way to embed respectively an audio file and a video file on a web page

External reference tag

External contents are distinguished using tag **`<a>`** followed by the ***`href`*** attribute with the explicit reference to the multimedia file.

Example (film `"filmato.mov"`):

```
<a href="filmato.mov"> Vedi il Filmato </a>
```

- The *href* attribute is also used to link to a different page.

  Example (link to page "libro.html"):

```
<p>Questo è un collegamento <a href="libro.html"> alla
pagina del libro</a></p>
```

- To link to a page on a different web server the full Internet address is needed.
  Example (link to www.w3c.org):

```
<p>Questo è un link al <a href="http://www.w3c.org/"> W3C
</a> </p>
```

Object tag

Inclusion of external content that requires special plug-in to be displayed requires the special tag **<object>** recommended by W3C.

- For example, Adobe Flash animations in the HTML page must use tag **<object>** with appropriate parameters

CSS (Cascade Style Sheets)

- **CSS (Cascade Style Sheets)** are a set of rules that tell the browser how the page must be displayed. It is the means to separate fromatting instructions that define the web page aspect from content.
- CSS refers to logical blocks (*block-level* elements) :
  - `<body>`
  - `Heading tags, e.g., <H1>, <H2>`
  - `<p>`
  - `List tags, e.g., <ul>, <ol>, <dl>`
  - `<img>`
  - `<div>`
  - …
- Each CSS element is referred as ***style***: it permits to modify font, background, color, layout…..
  - …

Cascading order

- CSS styles can be defined in different ways: ***inline, internal and external***. Precedence order for the application of styles is defined as follows (from the lowest to highest) :

1. **Browser default**
2. **External style sheets**
3. **Internal style sheets**
4. **Inline styles**

-

+

Inline style

The simplest method for definition of styles is *inline styles*. It is implemented with the attribute **`style`** applied to a tag and adds information on the style of any single element of the document

Inline style sheet example:

```
<p style ="font:1.00em/1.20em  verdana,helvetica,sans-serif;">  titolo1</p>
```

- Use it when you need to format just a single section in a web page

Internal style sheet

Internal style sheet are used instead to define styles that apply to the entire document.
Rules are defined using the tag **`<style>`** in the HEAD part of the HTML document.

- In the tag **`<style>`** it is possible to define rules for any specific element of the document.
  In this case properties are applied to all the instances of the element throughout the document.
  Attribute *`type`* is set to **`"text/css"`**.

```
<head>
  [...]
  <style type="text/css">
        <!-- body { font:80% Verdana,Helvetica,  sans-serif;  } -->
        p{font:  1.00em/1.20em verdana,  helvetica,  sans-serif}
        [...]
  </style>
[...]
</head>
```

- Use it when you need to modify all instances of particular element (e.g., `p`) in a web page

External style sheet

External style sheets are a different mode in which style sheets are saved in a specific external file with `.css` suffix. Linkage to the HTML code is obtained using the tag `<link>` in the HEAD part of the HTML document.

```
<head>
  [...]
  <link rel="stylesheet" type="text/css" media="screen"
  href="/css/foglio_di_stile.css" />
  [...]
</head>
```

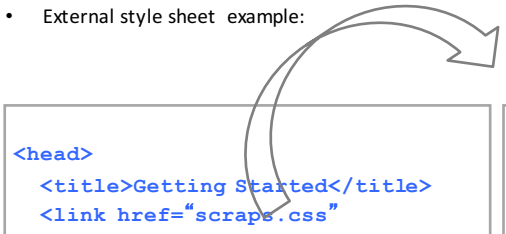
Attributes of the `link` tag are:

- ***href*** is set to the stylesheet file URL
- ***rel*** is set to "stylesheet" to specify it is a stylesheet file
- ***type*** is set to "text/css"
- ***media*** indicates whether content is displayed or printed:

```
<link rel="stylesheet"  type="text/css"  media="screen"
  href="/css/view.css"  />
<link rel="stylesheet"  type="text/css"  media="print"
  href="/css/print.css"  />
```

- Use it when you need to control the style for an entire web site. Wherever possible, place your styles in external style sheets

- External style sheet example:



scraps.css

```
<head>
  <title>Getting Started</title>
  <link href="scraps.css"
  rel="stylesheet" type="text/css"/>
</head>
[...]
```

HTML file

```
h1 {font-family: sans-serif;
   color: orange}
b {color: blue}
```

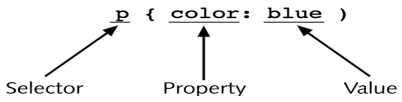
Text file of CSS "stylesheet"

CSS rule syntax

- CSS styles are applied to elements of the HTML document using ***selectors*** and include two separate components:
  - **property** (refers to the specific element)
  - **value** (determines the visual features of the style).

  Property and values are referred to as "*declaration*" or "*style declaration*"

CSS rule syntax is: `selector {property:value; property: value}`



CSS example (formatting the element `<titolo>`):

```
titolo {font-family: Arial; font-size: 18pt; color: #0000FF;
text-align: center}
```

CSS example (formatting the element `<titolo>` inside the element `<argomento>`):

```
argomento titolo {font-family: Arial}
```

Selectors for style application

- Selectors specify the element to which the style is applied. Main selector types are:
  - **Tags**
  - **Identifiers**
  - **Classes**
  - **Pseudo-classes and Pseudo-elements**
  - **Cascade selectors**
  - **Groups**

Tag selector

- In the HTML document any tags can be used as **selector**.

- In the CSS file, tag selectors are represented as:

```
p { text-indent: 2em; }
a { font-weight: bold; }
```

It is indicated to the browser that paragraphs must be indented of 2 characters and links must be bold

Identifier selector

- Identifiers are applied to tags and indicate a unique instance of the tag. Such instance cannot be repeated in the document.

  - In the HTML document identifiers are referred to with the **`id`** attribute followed by the **unique name** of the identifier that is associated with the tag:

```
<p id="nota-copyright">Questo paragrafo è la nota per il
  copyright</p>
```

  - In the CSS file, the name of the identifier to which properties should be applied must be preceded by character **'#'** :

```
#nota-copyright {font-size: xx-small}
```

- In the HTML text two distinct tags associated to the same identifier are not allowed:

  Example (not allowed):

```
<p id="nota-copyright">Questo paragrafo è la nota per il
   copyright</p>
<p id="nota-copyright">E' errato inserire questo paragrafo</p>
```

Class selector

- The same tag can be associated to different styles. This is obtained using **classes.**

  – In the HTML document, Classes are specified using the attribute ***class*** of a tag:

```
<p class="nota">Questo paragrafo è una nota</p>
<p class="evidenziato">Questo paragrafo è evidenziato in
   giallo</p>
```

  – In the CSS file, classes are specified with **tag.class** so to indicate that such class is associated to the tag. Their properties follow:

```
/* carattere piccolo */
   p.nota {font-size: small}
/* area con sfondo giallo */
   p.evidenziato {background-color: yellow;}
```

- Classes associated to different tags may have the same name:

HTML

```
<p class="nota"> Questo paragrafo è una nota </p>
<p>Testo normale (<span class="nota">nota nel testo</span>) testo
   normale ..</p>
```

CSS

```
p.nota {font-size: small} /* carattere piccolo */
span.nota {color: #999999;} /* testo in grigio */
```

- Every tag is associated to one class only:
  `p.classe1.classe2 {…}` is not valid

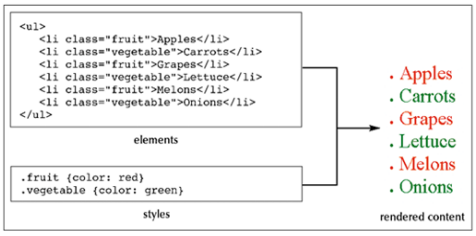
- Classes can be defined with no specific association to a tag. In this case, in the CSS classes are specified with **.class** so to indicate that such class can be associated to any tag in the HTML code:

HTML

```
<p class="nota">Questo paragrafo è una nota</p>
<div class="nota">Questa sezione è una nota</div>
```

CSS

```
.nota {font-size: small} /* carattere piccolo */
```



Pseudo-class selector

- Pseudo-classes are special classes that are directly recognized by browsers. In this case there is no specification in the HTML.

Pseudoclass syntax is therefore specified only in the CSS:

```
selector:pseudoclass  {property:  value}
```

- Pseudo-classes determine distinct elements. The most typical case of use is to specify how should appear links visited from active links (tag **a**).

Link definitions in CSS:

```
a:link {color:  blue;}
a:visited {color: purple;}
a:hover {color: red;}
a:active {color: dark-red;}
```

in this case CSS tells the browser that:

- links must be blue
- visited links must be violet
- links where the mouse is rolling over must be red
- active links must be dark red

Selector cascading

In the CSS file, Selectors can be cascaded :

```
selector1 selector2 { propriety: value }
```

In this case rule applies to selector2 if it is contained in selector1

- Example:

CSS:

```
code i { color: red; }
```

it specifies that tag "`i`", if included in tag "`code`", must have text colored in red

HTML:

```
<p><code>testo normale <i>testo corsivo</i>testo normale</code></p>
<p>testo normale <i>testo corsivo</i> testo normale</p>
```

testo normale *testo corsivo* testo normale

testo normale *testo corsivo* testo normale

Selector specificity

- Selectors have specificity:
  - Identifiers: specificity 100
  - Classes: specificity 10
  - Tags: specificity 1

- Selector cascading is overridden considering the specificity of the selector

Selector grouping

If several selectors have the same rules then grouping is possible as follows:

```
h1, h2, h3, h4, h5, h6 { font-family: sans-serif }
```

- This is the same as:

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
h4 { font-family: sans-serif }
h5 { font-family: sans-serif }
h6 { font-family: sans-serif }
```

Property and value of styles

Property and value are specified with reference to selectors:

```
selector {property:value; property: value}
```

property		value	
background-color	*	red,green, blue, #666633,..	indica il colore dello sfondo di un elemento
background-image	*	url	specifica l'immagine da visualizzare come sfondo di un elemento
border-style		none, dotted, double,..	specifica lo stile del bordo da applicare a un elemento
color		red,green, blue, #666633,...	specifica il colore di un elemento
display		block, inline, none	specifica le modalità di visualizzazione di un elemento
font-family	*	times, helvetica, arial	indica il tipo di carattere da utilizzare
font-size	*	small, large, 10pt, 12pt,..	indica le dimensioni dei caratteri
font-weight		normal, bold, 100, 200, ..	indica lo spessore dei caratteri
height, width	*	30px, 40px, 75%, ..	indicano l'ampiezza e la larghezza di un elemento
position	*	absolute, relative, static	indica il tipo di posizionamento da applicare
text-decoration	*	none, underline, blink, ..	indica alcune decorazioni da applicare al testo, come ad esempio la sottolineatura
top, left	*	30px, 40px, 50px	indicano le coordinate dell'angolo superiore sinistro di un elemento
z-index	*	1,2,3,..	specifica il posizionamento degli elementi nello spazio (sovrapposizione)

W3C specification syntax for properties and values

- W3C specification syntax (following pages)
  - **<Abc>: value of Abc**
  - **A B C: keywords requested**
  - **A|B: either A or B**
  - **A||B: either A or B or both**
  - **[Abc]: grouping**
  - **Abc*: Abc will appear zero o several times**
  - **Abc+: Abc appears once at least**
  - **Abc?: Abc is optional**
  - **Abc{n,m}: Abc must appear n times minimum and m times maximum**

font-family property

- Syntax

```
font-family: <font name>* || <generic font>
```

  - `<font name>`
    - any font name (e.g. verdana, helvetica, georgia)
  - `<generic font>`
    - `serif`
    - `sans-serif`
    - `monospace`
    - `cursive` [not suggested]
    - `fantasy` [not suggested]

- Example:

```
p { font-family: verdana, helvetica, "Trebuchet MS", sans-serif }
```

in a chain of attributes the font that is applied is the first available from left

font-size property

- Syntax:

```
font-size: <absolute dimension > | <relative dimension> |
<value> | <percentage>
```

  - **<absolute dimension>**
    xx-small | x-small | small | medium | large | x-large | xx-large
  - **<relative dimension>**
    larger | smaller
  - **<value>**
  - **<percentage>** ….

- Percentage is recommended by W3C: it is *"equal to the computed value of the font-size"*. It is reasonably assumed that browsers set font size value to 16px.
  p {font-size: 0.875em} determines a font size of 14px

font-style property

- Syntax:

```
font-style: normal | italic | oblique;
```

  Value **italic** makes text oriented. value **oblique** provides the same effect in the opposite direction. **normal** does not determine any changes.

- Example:

```
quote { font-style: italic }
```

font-variant *property*

- Syntax:

```
font-variant: normal | small-caps
```

- Example:
  In the CSS

```
.maiuscoletto { font-variant: small-caps }
```

Note that `.maiuscoletto` in the CSS corresponds to the class definition “maiuscoletto” in the HTML.

The `Small-caps` attribute creates ”*maiuscoletto*", `normal` does not introduce any changes.

font-weight *property*

- Syntax:

```
font-weight: normal | bold | bolder | lighter | 100 | 200
| 300 | 400 | 500 | 600 | 700 | 800 | 900
```

provides font weighting. Value `bolder` and `lighter` are set with reference to default value. The other values are absolute values in a scale 100-900. Not any value is available for any font. In this case the closest value is applied.

- Example:

```
strong { font-weight: 900 }
```

font *property*

The `font` property can be employed as a quick method to define properties like **`font-family`**, **`font-style`**, **`line-height`** (for line spacing)..... With font, values for properties font-size and font-family must be specified at least. Syntax:

```
font: <value>

   <value>
      [ <font-style> || <font-variant> || <font-weight> ]?
      <font-size>
      [ <line-height> ]? <font-family>
```

Esempio:

```
p.paragrafo-speciale { font: bold 1.00em/1.50em verdana,
helvetica, "Trebuchet MS", sans-serif }
```

color *property*

Property **`color`** permits the specification of the color for the text of a selector. Syntax is:

```
color: <color>
```

`<color>` same values as color property

Example:

```
a {color: #009 }
em {color: red }
```

background-color *property*

Property **background-color** permits the specification of the background color and background image. Syntax is:

background-color: <value>

<value>
<color> || <image> || <repetition> || <scrolling> || <position>

<color> background color can be specified using the same values as color property
<image> the image URL
<repetition>: repeat | repeat-x | repeat-y | no-repeat
the background image can be repeated, repeated horizontally or vertically only, or not repeated
<scrolling>: scroll | fixed background image moves with text or remain still
<position>: [<percentage> | <lenght>]{1,2} | [top | center | bottom] || [left | center | right] the position of the background image can be specified either with reference to the top left corner, or as percentages, or unit lenght or using keywords

- Example:

```
body { background:#FFFFFF url(/images/sfondo.gif) no-repeat fixed
top right;}
```

word-spacing *property*

Property **word-spacing** permits setting of the space between words. Syntax is:

word-spacing: <value>

<value>
normal | <lenght>

- Example:
  - **.parole-distanziate { word-spacing: 1.00em }**
  - **.parole-ravvicinate { word-spacing: -0.20em }**

letter-spacing property

Property **letter-spacing** defines spacing between letters of a word. Syntax is:

```
letter-spacing: <value>

 <value>
     normal | <lenght>
```

- Example:
  - `.lettere-distanziate { letter-spacing: 0.50em }`
  - `.lettere-ravvicinate { letter-spacing: -0.15em }`

line-height property

Property **line-height** permits to define interlining. Syntax is:

```
line-height: <value>

 <value>
     normal | <lenght> | <percentage>
```

- Example :

```
.interlinea-ridotta { line-height: 100% }
```

Percentage refers to font size. Negative values are not allowed

text-indentation property

Property **text-indentation** permits to set indentation at linebreaks. Syntax is:

```
text-indentation: <value>

<value>
    <lenght> | <percentage>
```

- Example:

```
p { text-indentation: 2em; }
```

text-decoration property

Property **text-decoration** permits decoration of text. Syntax is:

```
text-decoration: <value>

<value>
   none | [ underline|| overline|| line-through|| blink ]
```

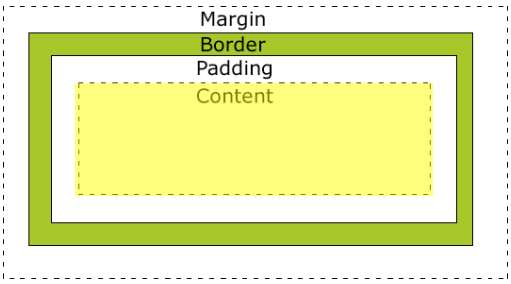
- Example:

```
#barra-di-navigazione A { text-decoration: none }
/* non-underlined link */
```

CSS box model

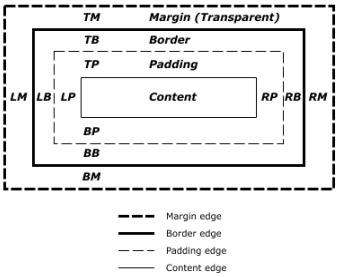
- The **CSS Box Model** allows us to place a border around *block-level* elements and space elements in relation to other elements.
- The CSS box model is essentially a box that wraps around HTML block-level elements, and it consists of:
  - ***margins,***
  - ***borders,***
  - ***padding,***
  - **the *actual content.***

- *Margin:* clears an area around the border; it is completely transparent
- *Border* : a border that goes around the padding and content
- *Padding*: clears an area around the content
- *Content:* where text and images appear



Box edges

- Each box has four edges:
  - ***content edge* or *inner edge*** The content edge surrounds the rectangle given by the width and height of the box, which depend on the rendered content.
    The four content edges define the box's *content box*.
  - ***padding edge*** The padding edge surrounds the box padding. If the padding has 0 width, the padding edge is the same as the content edge.
    The four padding edges define the box's *padding box*.
  - ***border edge*** The border edge surrounds the box's border. If the border has 0 width, the border edge is the same as the padding edge.
    The four border edges define the box's *border box*.
  - ***margin edge*** The margin edge surrounds the box margin. If the margin has 0 width, the margin edge is the same as the border edge.
    The four margin edges define the box's *margin box*.



Box size

- The dimensions of the content area of a box, the ***content width*** and ***content height***, depend on several factors:
  - whether the element generating the box has the 'width' or 'height' property set,
  - whether the box contains text or other boxes,
  - whether the box is a table,
  - ……

- When setting the width and height properties of an element with CSS, just the width and height of the content area is set. To calculate the full size of an element, the paddings, borders and margins must also be added.

  - The total width of an element should be calculated like this:
    Total element width = **width + left padding + right padding + left border + right border + left margin + right margin**

  - The total height of an element should be calculated like this:
    Total element height = **height + top padding + bottom padding + top border + bottom border + top margin + bottom margin**

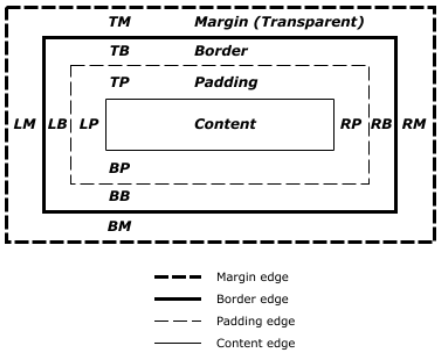
- Example:

  Content width:250px; padding:10px; border:5px solid gray; margin:10px;

  Doing the math the total width of the element is 300px:

  width: 250px
  + 20px (left and right padding)
  + 10px (left and right border)
  + 20px (left and right margin)

  = 300px



Box background

- The background style of the content, padding, and border areas of a box is specified by the 'background' property of the generating element:
  - The margin does not have a background color, it is completely transparent
  - The border is affected by the background color of the box
  - The padding is affected by the background color of the box

Interactive Box Model sizing example

- http://guyroutledge.github.io/box-mod

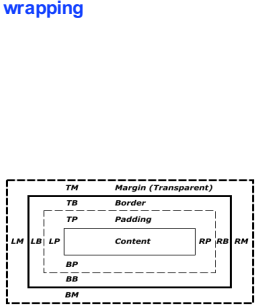
Box model example

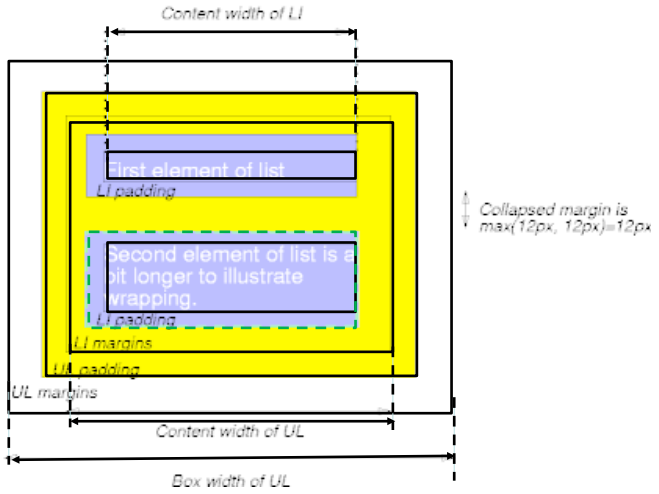
- Box model example:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<HTML>
  <HEAD>
    <TITLE>Examples of margins, padding, and borders</TITLE>
    <STYLE type="text/css">
      UL { background: yellow; margin: 12px 12px 12px 12px; padding: 3px 3px 3px 3px; /* No
      borders set */ }
       LI { color: white; /* text color is white */ background: blue; /* Content, padding will be blue */
       margin: 12px 12px 12px 12px; padding: 12px 0px 12px 12px; /* Note 0px padding right */ list-
       style: none /* no glyphs before a list item */ /* No borders set */ }
       LI.withborder { border-style: dashed; border-width: medium; /* sets border width on all sides
       */ border-color: lime; }
 </STYLE>
  </HEAD>
  <BODY>
      <UL>
        <LI>First element of list
         <LI class="withborder">Second element of list is a bit longer to illustrate wrapping.
      </UL>
  </BODY>
</HTML>
```

```
UL { background: yellow;
     margin: 12px 12px 12px 12px; padding: 3px 3px 3px 3px; /* No borders set */ }
LI { color: white; /* text color is white */ background: blue; /* Content, padding will be blue */
    margin: 12px 12px 12px 12px; padding: 12px 0px 12px 12px; /* No borders set */
LI.withborder { border-style: dashed; border-width: medium; /* sets border width on all sides */ border-
color: lime;}

<LI>First element of list
<LI class="withborder"> Second element of list is a bit longer to illustrate
wrapping
```





Block element positioning

- Default positioning of a block element is static, i.e. the block element follows the normal flow of the page
- The default positioning can be altered using either the ***position*** or the ***float*** property

Position property

property	value	description
position	static	default position
	relative	offset from its normal static position
	absolute	a fixed position within its containing element
	fixed	a fixed position within the browser window
top, bottom, left, right	positions of box's corners	

Absolute positioning

```
#menubar {position: absolute;left: 400px;top: 50px;}
```

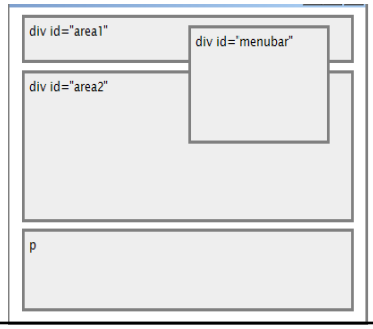
CSS

- Absolute-positioned elements are normally positioned at an offset from the corner of the overall web page.
  - removed from normal flow
  - actual position determined by `top`, `bottom`, `left`, `right`
  - should often specify a `width` property as well

Original HTML
.....

.....

...



Relative positioning

```
#menubar {position: relative; top: 20px; left: 400px }
```

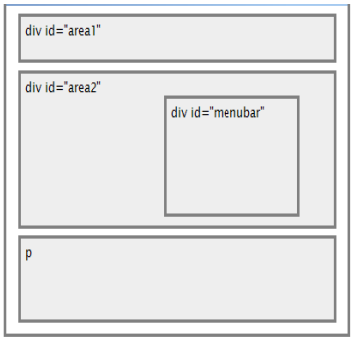
CSS

- To make the absolute element to position itself relative to some other element's corner, wrap the absolute element in an element whose position is relative:
  - positioned relative to the block element containing it

Original HTML
.....

..... <p
.....>

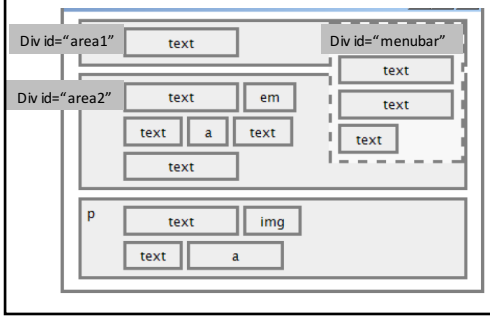
 .....
.



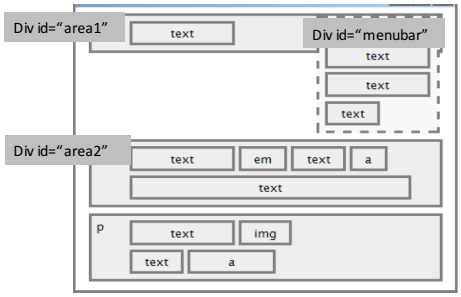
Float property

- The float property specifies whether or not a box (an element) should float. Absolutely positioned elements ignores the float property. Elements after a floating element will flow around it. To avoid this, use the clear property
- To float (wrap) a block-level element, use:
  `selector {float:value;}` where *value* = `right, left, none`
- To prevent an element from wrapping, use:
  `selector {clear:value;}` where `value=right, left, both, none`

```
#menubar {float: right;}
```




```
#menubar {float: right;}
#area2 {clear: right;}
```



Interactive positioning example

Validating CSS

- W3C provides W3C CSS Validation Service utility to validate CSS external style sheets