HTML 5

Marco Torchiano

Version 1.6.2 - April 2020

License

Introduction

What is HTML

HTML (HyperText Markup Language)

- Is a language used to describe the contents of web documents.
- HTML documents consist of a a set of tree-structured elements and text.
- HTML specifies content, separately from presentation (see *CSS*)

History

1991:
  first version of HTML by Tim Berners-Lee

1999:
  HTML 4.01 Recommendation

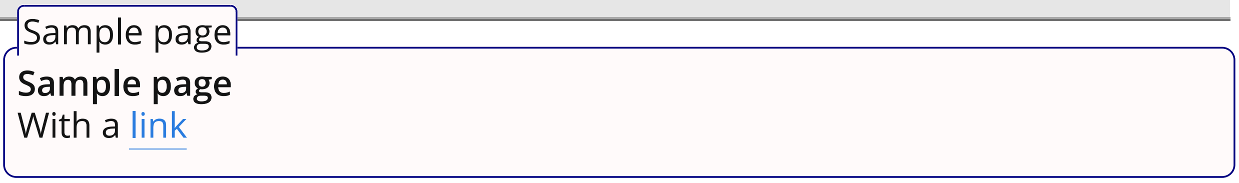
2000:
  XHTML, eXtensible HyperText Markup Language, bridge between HTML and XML

2014:
  W3C HTML 5 Candidate Recommendation

2017:
  HTML 5.2 - W3C Recommendation

Example

```
<!DOCTYPE html>
<html>
  <head lang="en"><meta charset="utf-8">
    <title>Sample page</title>
  </head>
  <body> <!-- this is a comment -->
    <h1>Sample page</h1>
    <p>With a <a href="demo.html">link</a></p>
  </body>
</html>
```


Sample page

Sample page
With a link

HTML Syntax

- Text fragment, e.g. `With a`
- Tag, e.g. `<body>`
  Most tags are defined in pairs:
  - opening, e.g. `<p>`
  - closing, e.g. `</p>`
- Elements, i.e. a pair of opening and closing tags with their contents, e.g. `<h1>Sample page</h1>`
- Attributes of elements, e.g. `href="demo.html"`
  - Consist of a pair `name` `=` `"value"`
  - Defined in the opening tag of an element

Structure and elements

Basic Structure

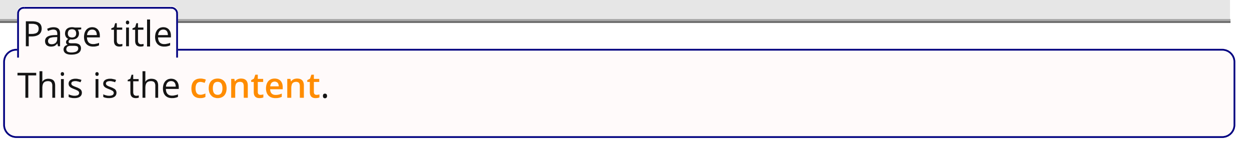
- `!DOCTYPE` declaration
- `html` content, including:
  - `head` block

    includes (meta-)information about the document, which are not shown
  - `body` block

    the wrapper that surrounds the actual content of the page that is shown in the browser window

Example

```html
<!DOCTYPE html>
<html>
 <head lang="en">
  <meta charset="utf-8">
  <title>Page title</title>
 </head>
 <body>
  <p>This is the <strong>content</strong>.</p>
 </body>
</html>
```


Page title

This is the **content**.

Element types

Block:

high level, define the structure of the document.

A block element is rendered starting on a new line, breaking away from the previous content

Inline:

contained within block elements, typically surround only small portions.

Inline elements rendering do not imply a break in the document flow

Head

- Title of the document `<title>` + `</title>`
- Charset `<meta charset="utf-8">`
- Meta information: `<meta name=".." content="..">`, where name can be:
  - `description`
  - `keywords`
  - `author`
- Style `<style type="text/css">` + `</style>`
- Scripts `<script>` + `</script>`

Body

- Any element
  - excluding those in `<head>`
  - including `<script>`
- The body contents rendering is shown inside the browser window

Text block elements

- Headings `<h1>`, `<h2>`, `<h3>`, …
- Paragraph `<p>`
- Quotations `<blockquote>`
- Preformatted text `<pre>`

NOTE: Multiple spaces are collapsed into a single space

Text block examples

```
<h3>Level 3          heading</h3>
<p>Paragraph, may include several lines...</p>
<pre>
   formatted
     exactly as in source     code
</pre>
```

Level 3 heading

Paragraph, may include several lines...

```
   formatted
     exactly as in source     code
```

Text inline elements

- Emphasis `<em>` (and older *italics* `<i>`)
- Strong `<strong>` (and older **bold** `<b>`)
- Line break `<br>`

Note: line breaks in code are ignored: you should use either `<p>` or `<br>`

Text inline examples

```
Normal code <em>with emphasis</em>
or <i>italics</i>.
<br>
<strong>strong emphasis</strong> or <b>bold</b>
<br>
<p>And other paragraphs</p>
```

Normal code *with emphasis* or *italics*.
strong emphasis or **bold**
And other paragraphs

Lists

- Unordered `<ul>`
- Ordered `<ol>`
- Definition `<dl>`

List items for ordered and unordered list are defined with `<li>` elements.

Definition items consist in a term `<dt>` and a definition `<dd>`

Lists (also of different types) can be nested.

Unordered List

```
<ul>
  <li>an item</li>
  <li>another item, and </li>
  <li>yet another item</li>
</ul>
```

- an item
- another item, and
- yet another item

Ordered List

```
<ol>
  <li>first item</li>
  <li>second item</li>
  <li>third item</li>
</ol>
```

1. first item
2. second item
3. third item

Definition List

```
<dl>
  <dt>List</dt>
  <dd>A sequence of items</dd>
  <dt>Item</dt>
  <dd>An element of a list</dd>
</dl>
```

List
  A sequence of items

Item
  An element of a list

Links

Links are parts of a document pointing to other resources.

- Element `<a>` marks a fragment as a link.
  The attributes of the element are:
  - `href`: the locator for the resource the link points to
  - `title`: additional information shown on *hover*

Link example

```html
<p>This is a link to PoliTo
<a href="http://www.polito.it"
   title="PoliTo" >home</a>.</p>
<p>...and this is a link to
   <a href="/VIQ"
      title="Only works online"
   >course home page</a>.</p>
```

This is a link to PoliTo home.

...and this is a link to course home page.

URL

Uniform Resource Locator

`scheme :// host / path ? query # fragment`

- Not all components are required
- Different subsets are possible, e.g.:
  - *scheme* + *host*
  - *scheme* + *host* + *path*
  - *host* + *path*
  - *path* (relative)

URL components (1)

- `scheme` defines the protocol to retrieve the resource:
  - can be `http`, `https`, `file`, `ftp`, etc.
- `host` corresponds to a valid internet domain name
  - `localhost` refer to the browser’s host
- `path` is mapped to a path on the server file system
  - a folder (e.g. `/VIQ`)
  - a file (e.g. `/VIQ/labs17/Lab2.html`)

URL components (2)

- `query` is used to pass parameters to an application server using the HTTP `GET` method
  - typically mapped from the fields of a form
  - e.g. `key1=value1&key2=value2`
- `fragment` refers to positions of elements inside a page
  - elements are identified by means of a unique `id` attribute

Fragment links

A link to a fragment needs a target fragment identified by means of an `id` parameter:

```
<section id="info-section"> ... </section>
```

A link can refer to that fragment using the id preceded by the `#` character:

```
<a href="#info-section">link to Info section</a>
```

Images

Images can be inserted using a **`<img>`** element.

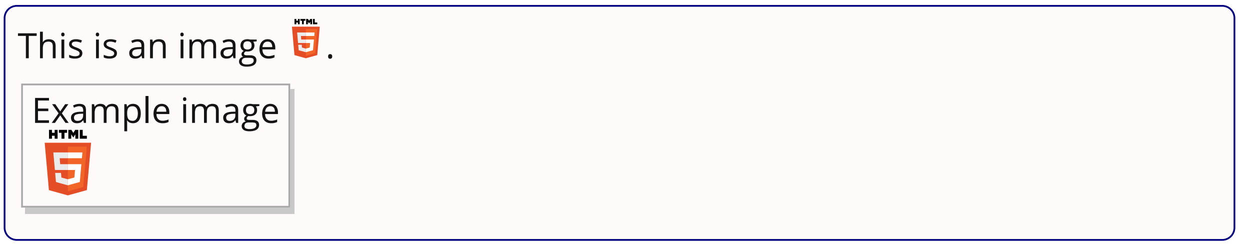
- `src` specifies the location of the image as a URL
- `title` provides a tooltip
- `width` and `height` define the size of the image if defined page rendering is faster

Note: images (`<img>`) are *inline* elements!

- **`<figure>`** is a block-element intended to contain images
- **`<figcaption>`** element defines the caption

Example image

```
This is an image <img src="HTML5Logo.svg"
      title="The HTML logo"
      width=30 height=30>.
<figure><figcaption>Example image</figcaption>
<img src="HTML5Logo.svg" width=50 height=50>
</figure>
```


This is an image .

Example image

- The actual rendering depends on style.

Structural elements (HTML 4)

- Headers: `<h1>` - `<h6>`
- Divs: `<div>` block container

  Used to add blocks in the page with special formatting
- Spans: `<span>` inline container

  Used to apply special formatting to a span of text

Sectioning elements (HTML5)

- `<article>`: a complete, or self-contained, composition in a document
- `<section>`: a generic section of a document, typically with a heading.
- `<nav>`: a section with navigation links
- `<aside>`: content that is tangentially related to the content around
- `<h1>` - `<h6>`: a section heading
- `<header>`: a group of introductory or navigational aids
- `<footer>`: a footer for its nearest ancestor

Tables

Tables arrange elements in rows and columns.

The element `<table>` includes

- Row elements, `<tr>` that include
  - Cells elements `<td>` that include
    - the cell content (i.e. any html)

  The elements are described *by row*
- Header cells are represented by element `<th>`
- Table caption is defined in a `<caption>` element

Table example

```
<table>
<caption>Departments staff size</caption>
<tr><th>Department</th><th>Staff</th></tr>
<tr><td>Computer science</td><td>68</td></tr>
<tr><td>Chemistry</td><td>120</td></tr>
</table>
```

Departments staff size

Department	Staff
Computer science	68
Chemistry	120

Note: the exact rendering depends on the style.

Characters

Entities

- Entities are used to represent special characters and characters not present or *typeable* with a keyboard
  - Special characters (e.g. `<`, `>`, and `&`) cannot appear directly inside an html document
- Entities:
  - start with `&`
  - terminate with `;`
  - in between there is a name or a code value

Entities examples

- Named entities:
  - `&lt;` : `<`
  - `&gt;` : `>`
  - `&amp;` : `&`
- Code entities
  - `&#169;` : ©

    Note: the actual character depends on the *charset* of the document.

Character set and encoding

- A character set is the a set of characters
  - e.g., ASCII defines 127 character (letters + numbers + symbols)
- The character set encoding is a mapping from the character set to a sequence of bytes
  - e.g., ASCII letter `I` corresponds to 0x41 (65 decimal)
- Encoding is used by the browser to convert the stream of bytes into characters

Unicode

Standard that assigns a unique code to every character in any language

- Core specification gives the general principles
- Code charts show representative glyphs for all the Unicode characters.
- Annexes supply detailed normative information
- Character Database normative and informative data for implementers

Characters, Glyphs

- **Character**: the abstract concept

  e.g. `LATIN SMALL LETTER I`

- **Glyph**: the graphical representation of a character

  e.g. i *i* `i` i

  Font: a collection of glyphs

Codepoints

- **Codepoint**: the numeric representation of a character

  Included in the range 0 to $10FFFF_{16}$ (23 bits)

  Represented with `U+` followed by the hexadecimal code

  e.g. `U+0069` for i (`LATIN SMALL LETTER I`)
  e.g. `U+16B5` for ᚵ (`RUNIC_LETTER_G`)
  e.g. `U+0913` for ओ (`DEVANAGARI LETTER O`)

Encoding

Mapping from a byte sequence to a code point.

- UTF-32: fixed width, high memory occupation (4 bytes)

  e.g., '*I*', `U+0069` is represented as `0x00000069`
- UTF-16: variable width, represents
  - codepoints from `U+0` to `U+ffff` on 16 bits (2 bytes)
  - codepoints from `U+10000` to `U+10ffff` on 32 bits (4 bytes)

  e.g., '*è*', `U+00E8` is represented as `0x00E8`

Encoding UTF-8

- UTF-8: variable width,
  - codepoints `U+00` to `U+7f` are mapped directly to bytes, i.e. *ASCII transparent*
  - most non-ideographyc codepoints are represented by 2 bytes

    e.g. '*è*', `U+00E8` is mapped to `0xC3` + `0xA8`.

Enconding mismatch

- A mismatch occurs when the encoding used to convert a byte sequence to characters is note the same used to perform the coding.
- When character ‘*è*’ (`U+00E8`) is encoded using UTF-8 it results into two bytes: `0xC3` + `0xA8`. IS0-8859-1 encoding interprets the above two bytes as the characters ‘*Ã¨*’.
- Viceversa, ‘*è*’ is encoded by IS0-8859-1 as `0xE8` which is an invalid character in UTF-8 encoding (usually represented as �)

Forms

Forms

Forms are html structures that allow the user to enter information mean to be sent to the web server

- The form definition is enclosed in a `<form>` block element
- The `<form>` element has two attributes:
    - `method` specifies how data is transmitted
        - either `GET` or `POST`
    - `action` specifies the location (URL) that will receive the data

Methods: `GET` vs. `POST`

Differ in the way they encode the from values

- `GET` encodes them as a query portion in the URL

```
GET /api/get.jsp?USER=John&PASS=53cr3t HTTP/1.0
```

- `POST` encodes them as additional elements in the HTTP Request

```
POST /api/post.jsp HTTP/1.0

USER=John
PASS=53cr3t
```

Form items

- Items in the form are represented by
  - `<input>`
  - `<textarea>`
  - `<select>`

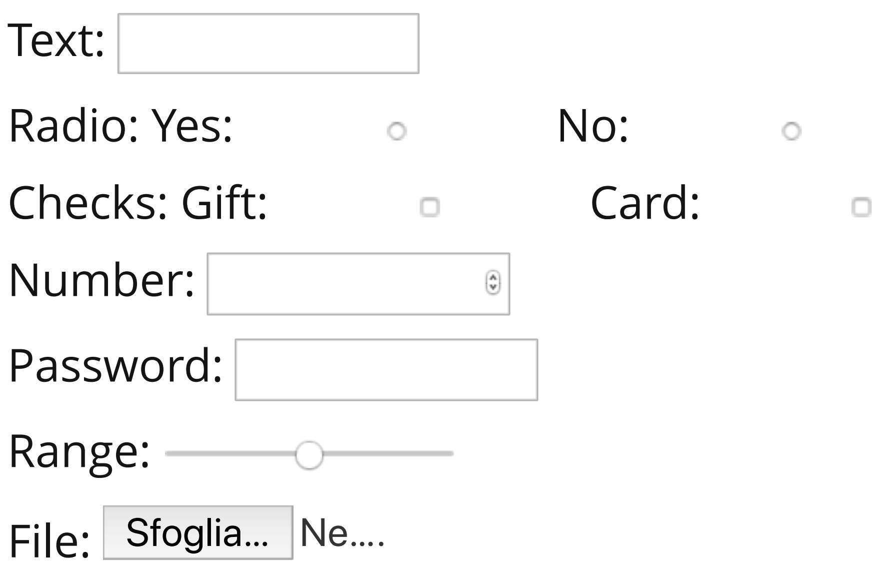
The elements have two main attributes:

- `name` name of the data item
- `value` value of the variable (initial or predefined for e.g. a button or hidden element)

`<Input>` element

- Attribute `type` defines the type of input:
  - `text`: text field
  - `submit`: button
  - `checkbox`: a checkbox
  - `radio`: a radio button (several with the same name)
  - `file`: file selection
  - `password`: text field with masked characters
  - `hidden`: invisible element

Example types


Text:

Radio: Yes: No:

Checks: Gift: Card:

Number:

Password:

Range:

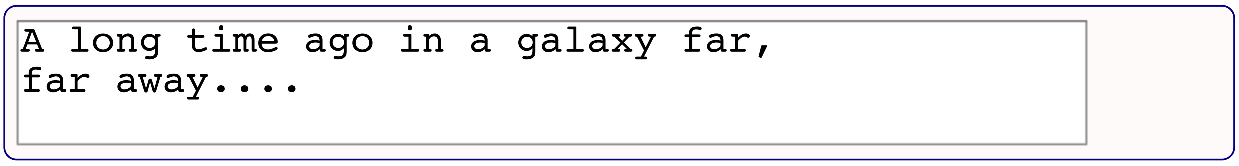
File: Sfoglia... Ne....

Other inputs

- `<textarea>`
  - attributes `cols` and `rows`
  - can enclose the initial value of the text
- `<select>` encloses a list of elements
  - `<option>` elements define the items of the list
    - the label is enclosed in the element
    - attribute `value` defines the relative value

Text area example

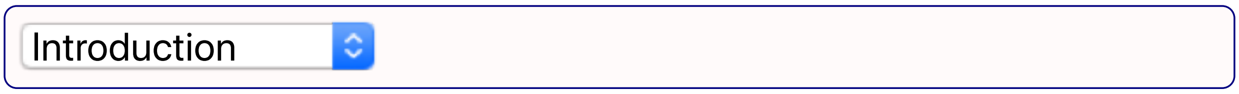
```
<textarea name="description" cols="45" rows="3">
A long time ago in a galaxy far,
far away....

It is a period of civil war.
Rebel spaceships, striking
from a hidden base, have won
their first victory against
the evil Galactic Empire.</textarea>
```



```
A long time ago in a galaxy far,
far away....
```

Select example

```
<select name="lecture">
<option value="L1">Introduction </option>
<option value="L2">Data Visualization</option>
<option value="L3">Measurement</option>
<option value="L4">Descriptive Stats</option>
<option value="L5">Perception</option>
</select>
```



Introduction

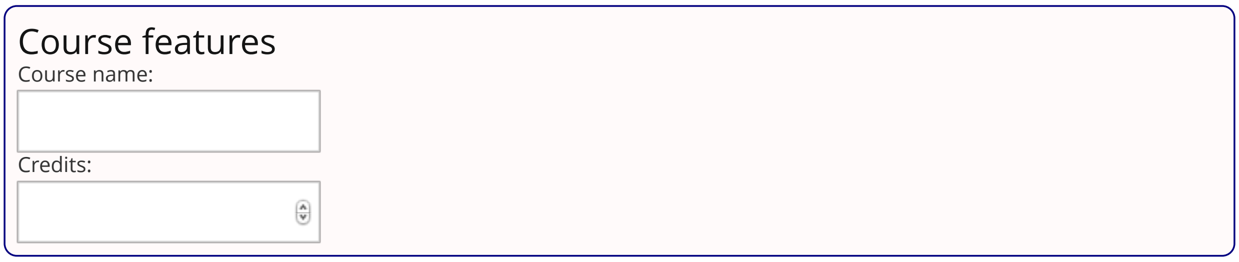
Labels and fieldsets

For accessibility define labels explicitly

- `<label>` defines elements
  - `for` refers to an input via its `id`
- `<fieldset>` groups related inputs
  - element `<legend>` encloses a name for the fieldset

Labels and fieldsets

```html
<fieldset><legend>Course features</legend>
<label for="course">Course name:</label>
<input type="text" name="course" id="course"><br>
<label for="cfu">Credits:</label>
<input type="number" name="cfu" id="cfu"
                        min="3" max="24">
</fieldset>
```


Course features
Course name:
Credits:

Terminology

Mime type

- Multipurpose Internet Mail Extensions

  Specify the nature of the data in the body of a MIME entity, by giving media type and subtype identifiers

  RFC2046

- Encoded using the format:

top-level* `/` *subtype* `;` *param* `=` x *value

Top-level mime types

- `text` textual information
- `image` image data (binary)
- `audio` audio data (binary)
- `video` video data (binary)
- `application` some kind of data interpretable by an application

All mime types must be registered with the IANA.

Mime types catalog

- `text/plain` typically require a *charset* parameter: `Content-type: text/plain; charset=iso-8859-1`
- `text/html`

Common image types:

- `image/jpeg`
- `image/png`
- `image/svg+xml`

General binary content `application/octet-stream`.

References

W3C HTML 5.2 Recommendation: https://www.w3.org/TR/html5/

WHATWG - Web Hypertext Application Technology Working Group: https://html.spec.whatwg.org

W3C: https://docs.webplatform.org/wiki/guides/the_basics_of_html

RFC2046 - MIME - Multipurpose Internet Mail Extensions: https://tools.ietf.org/html/rfc2046

UNICODE: http://www.unicode.org/versions/latest/

MDN Web technology reference: https://developer.mozilla.org/en-US/docs/Web/Reference

W3Schools HTML5 Tutorial: https://www.w3schools.com/html/default.asp