Lezione10-Modern HTML

Thursday, March 18, 2021 8:37 AM

<WA1/>
<AW1/>
2021

(Modern) HTML

A fast-paced introduction to HTML5

Fulvio Corno
Luigi De Russis
Enrico Masala

This week we create layout of the page, not js code!
We look directly at html5, modern version, modern approach to design

Goal

- Quick introduction to the essential features of HTML
  - Document structure and syntax
  - Essential elements
  - Semantic elements
  - Page structuring
- Aligned to what HTML 5 looks like in year 2020

Rough timeline of web technologies

Trying to be neutral about what content you want to produce

HTML
- CERN
- 1991

HTML 2.0
- IETF
- 1995

W3C®

HTML 3.2
- W3C
- 1997

W3C: World Wide Web Consortium

HTML 4.01
- 1999

XHTML 1.0
- W3C
- 2000
- XML-based; discontinued

Experiment, nice syntax, many validation rules derived from xml

HTML5
- WHATWG and W3C
- 2014

WHATWG: Web Hypertext Application Technology Working Group

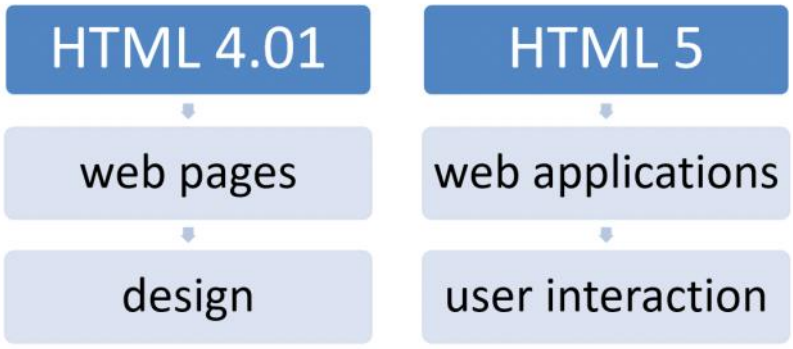
30 years old!

https://html.spec.whatwg.org/multipage/
https://html.spec.whatwg.org/dev/

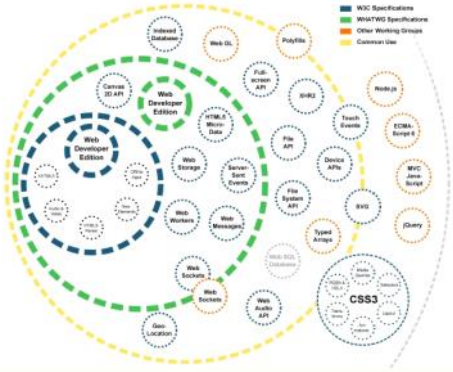
Living standard, dynamic versions, all news into html5 (no v1,2,3,4,)

Reduce verbosity and formalness of the last versions, big step to semplify it!
Initiative against the institute! No neutral to what we create, create features to help programmer to write sites like they are written today!

A different approach



HTML 5 is a family of standards



Big, many standards!
Various browsers decide which standard to include!

Live standard: Browser Compatibility?

- https://caniuse.com/

Understand which users will be able to use the functionality to decide which features of the language we should include! Living updates!

Current version of the browser

A basic HTML document

Nested tree of elements like xml.
Each elemetns can contain elements or attributes.

Easy header:
Doctype
Html;
Head
body

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <title>Sample page</title>
 </head>
 <body>
  <h1>Sample page</h1>
  <p>This is a <a href="demo.html">simple</a> sample.</p>
  <!-- this is a comment -->
 </body>
</html>
```

Name=value

- *Unicode* Text File
- DOCTYPE declaration
- Nested tree of Elements
  - Strict nesting
- Element: Basic element
  - Start tag: `<a>`
  - Attribute: `href="demo.html"`
    - Name=value pairs
  - End tag: `</a>`
  - Inner text: `simple`

Every type of elements has different attributes, functionalityes, behaviours

Mandatory structure

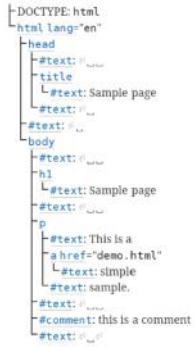
- First line: `<!DOCTYPE html>`
- `<html>` root element, contains the whole file
  - <head> (title, other metadata, load style sheets, load JavaScript code)
  - <body> (actual page content)

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <title>Sample page</title>
 </head>
 <body>
  <h1>Sample page</h1>
  <p>This is a <a href="demo.html">simple</a> sample.</p>
  <!-- this is a comment -->
 </body>
</html>
```


HTML parsing into a DOM tree

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <title>Sample page</title>
 </head>
 <body>
  <h1>Sample page</h1>
  <p>This is a <a href="demo.html">simple</a> sample.</p>
  <!-- this is a comment -->
 </body>
</html>
```

Parsing



Html document is just a text file/ text inside a string (received from the network)
We don't work on a string, but first we parse it into a tree (DOM).
Look at text and look at the tree or viceversa.
Each element can contain other elements, but there are elements ("a") which can contain things that are not element (special type of node #text contains free text (even white space!), it is not another element!) (special type of node #attribute which contains that element's attribute)
"a" node is a node with a list of children (#text) and a list of attributes (href).

DOM inspector can show it

HTML elements

- Elements are used to define the *meaning* of a portion of a document
  - Semantic markup
- The meaning will be rendered graphically, according to the style sheets
  - No HTML code should deal with presentation issues

- Each element has a default *display* value
  - `display:block`
    - Full-width
    - Starts on a new line
    - Top-to-bottom layout
  - `display:inline`
    - Does not start a new line
    - Occupies just the necessary space
    - Left-to-right layout
    - Wraps around at the end of line

Meaning of the various elements.
Some creates particular effects on the page (title, paragraph..)
Other just give an intended use for that section: specify layout of the page, while the content will be determined by the css!
In html we use semantic markup, not presentation topics!

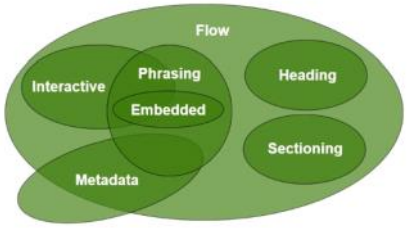
Block elements
(paragraphs,headings)
Takes a whole row of space!

Inline elements (text, images, buttons,..).
Just take necessary space
Typically inside a block element!

Rules to place elements: create block, then fill it with inline elements.
Element has a default behaviour, but they can be changed using css

Categories of HTML elements

Flow elements: Contains content of the page that flows from top to bottom



Structure of page

- *Sectioning* content defines headings and footers
- *Heading content* defines the header of a section
- *Phrasing* content is the *text* of the document, as well as elements that mark up that text *at the intra-paragraph level*.
  - Runs of phrasing content form paragraphs.
- *Flow* content includes most types of elements

Outline of the page (header level1, level2, level3)

Banners at top or bottom of a section, titles of the different sections..

Text + everything inside the paragraph: groups of elements, icons, buttons, that compose the individual elements inside an inline row

https://html.spec.whatwg.org/dev/dom.html#kinds-of-content

Sections and Headings (block)

Divide/markup parts of the page to indicate if this is part of the main part of the page (article) or to the aside part (lateral column) of the page, or to the navigation menu (nav)

Titles (inside various sections)

Sectioning content

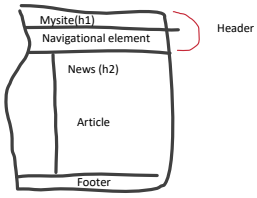
- `article`
- `aside`
- `nav`
- `section`
- `header`
- `footer`

Heading content

- `h1`
- `h2`
- `h3`
- `h4`
- `h5`
- `h6`
- `hgroup`

Aside: related content or navigational element

Invisible containers! Just group other elements



Usage guide

Suggestion about how to use the different elements

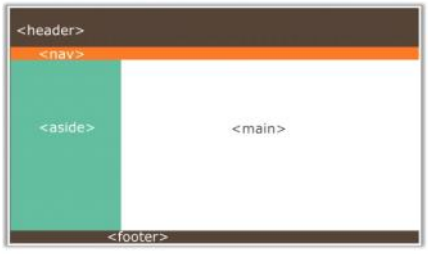
Element	Usage
<article>	complete, or self-contained, composition in a document, page, application, or site and that is, in principle, independently distributable or reusable
<section>	A generic section of a document or application. A thematic grouping of content, typically with a heading
<nav>	A section of a page that links to other pages or to parts within the page: a section with navigation links
<aside>	A section of a page that consists of content that is tangentially related to the content around the aside element, and which could be considered separate from that content (such as sidebars)
<h1>-<h6>	A section heading
<hgroup>	The heading of a section, that groups multiple h1-h6 elements, e.g., in case of subheadings
<header>	A group of introductory or navigational aids
<footer>	typically contains information about its section such as who wrote it, links to related documents, copyright data, and the like

You decide how you want to structure it.
Behaviour is the same as div, it is just to clarify usage!

<main>

https://html.spec.whatwg.org/dev/sections.html#usage-summary-2

Typical layout




```
<body>
    <header>
        <nav>

        </nav>
    </header>
    <aside>

    </aside>
    <main>

    </main>
    <footer>

    </footer>
</body>
```


Grouping content (block)

Block elements content inside structural elements

Element	Usage
<p>	a paragraph of text
<hr>	horizontal rule (represents a paragraph-level thematic break)
<pre>	block of preformatted text
<blockquote>	a section that is quoted from another source
	a list of items, where the items have been intentionally ordered. The items of the list are elements
	a list of items, where the order of the items is not important. The items of the list are elements
<menu>	an unordered list , whose elements are commands that may be executed
<dl>	a list of definitions. Each definition has a name (<dt>, definition term) and a value (<dd>, definition description)
<figure>	a sectioning element that can be used to annotate illustrations, diagrams, photos, code listings, etc. May include <img> or other content. May include <figcaption>
<main>	represents the dominant contents of the document
	the element has no special meaning at all. It can be used with the class, and title attributes to mark up semantics common to a group of consecutive elements. Use as a last resort.

Ordered list
Unnumbered list (bullets)

https://html.spec.whatwg.org/dev/grouping-content.html

Phrasing content (inline)

Text which can contain formatting (italics, bold, links), images

- a
- abbr
- audio
- b
- br
- button
- canvas
- cite
- code
- data
- datalist
- del
- dfn
- em
- embed
- i
- img
- input
- ins
- kbd
- label
- mark
- output
- picture
- progress
- q
- s
- samp
- select
- slot
- small
- span
- strong
- sub
- sup
- template
- textarea
- time
- var
- video
- wbr

- Insert content or format content inside a single paragraph
- Normally formatted left-to-right, and wrap at the end of line

https://html.spec.whatwg.org/dev/text-level-semantics.html

https://html.spec.whatwg.org/multipage/text-level-semantics.html

Interactive content (inline)

Can generate events, be dynamically programmed!

- a
- audio
- button
- details
- embed
- iframe
- input
- label
- select
- textarea
- video

Forms, checks, buttons

dropdown

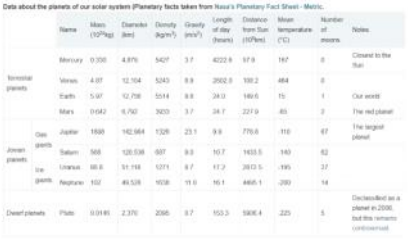
Big insert

- Elements specifically intended for user interaction
- May be used to compose a form
- May include multimedia interactive content
- In the simplest form, they may just be a link
  - `<a href="URL">anchor</a>`

Tabular data

Tabular set of rules

- `<table>` container
- `<thead>` groups header rows, `<tbody>` groups data rows, `<tfoot>` groups summary rows
- `<tr>` table row, includes
  - `<td>` cell with table data
  - `<th>` cell with table heading
- May have a `<caption>`
- `<colgroup>` may apply common attributes to a set of `<col>` columns
- `rowspan` and `colspan` attributes for creating complex grids



Data about the planets of our solar system (Planetary facts taken from Nasa's Planetary Fact Sheet - Metric.

		Name	Mass (10^{24}kg)	Diameter (km)	Density (kg/m^3)	Gravity (m/s^2)	Length of day (hours)	Distance from Sun (10^6km)	Mean temperature (°C)	Number of moons	Notes
Terrestrial planets		Mercury	0.330	4,879	5427	3.7	4222.6	57.9	167	0	Closest to the Sun
		Venus	4.87	12,104	5243	8.9	2802.0	108.2	464	0	
		Earth	5.97	12,756	5514	9.8	24.0	149.6	15	1	Our world
		Mars	0.642	6,792	3933	3.7	24.7	227.9	-65	2	The red planet
Jovian planets	Gas giants	Jupiter	1898	142,984	1326	23.1	9.9	778.6	-110	67	The largest planet
		Saturn	568	120,536	687	9.0	10.7	1433.5	-140	62	
	Ice giants	Uranus	86.8	51,118	1271	8.7	17.2	2872.5	-195	27	
		Neptune	102	49,528	1638	11.0	16.1	4495.1	-200	14	
Dwarf planets		Pluto	0.0146	2,370	2095	0.7	153.3	5906.4	-225	5	Declassified as a planet in 2006, but this remains controversial.

Key attributes (applicable to all elements)

Identifiers

Many to many. Any element may have 0 or 1 or many class.
Each class can contain 0 or 1 or many elements. (Generic label)

Element selected by id: unique inside the page!
Each element has only 0 or 1 ids

Give extra attributes to make elements recognizable to css and js which will manipulate them
These can be applied to ANY element! (Also together)

`class="aa bb cc"`

- Space-separated list of class identifiers
- Represents all the classes that this elements belongs to
- Heavily used in CSS and JS for matching portions of contents

`id="unique"`

- Represents an element's unique identifier
- Must be unique within this specific HTML document
- Heavily used in CSS and JS for finding/matching this specific element

(No effect on the page if we don't use those elements in css or js)

' = "

Id to every element may be too much, too much clutter!
For example i give a style to the header, it will include all the title, we don't neeed to give id to that h1
Incremental selection of ids

Key attributes (applicable to all elements)

Do not use! Apply style to one specific element!
<h3 style="color:red">
Inject css code in a particular element! NO! Define the style in css so that every time we use it we can use the same style and change it only once instead of changing styles one element by one element

`style="css fragment"`

- Apply a set of CSS declarations to this specific HTML element
- Use sparingly, prefer CSS rules that map to element `id` and `class`

Goal: clean html, dirty work on css

"Wildcard" elements

Empty block, typically has class/id to customize what happens inside that!

Empty inline element!
 fsfsdfsd ciao fdssd

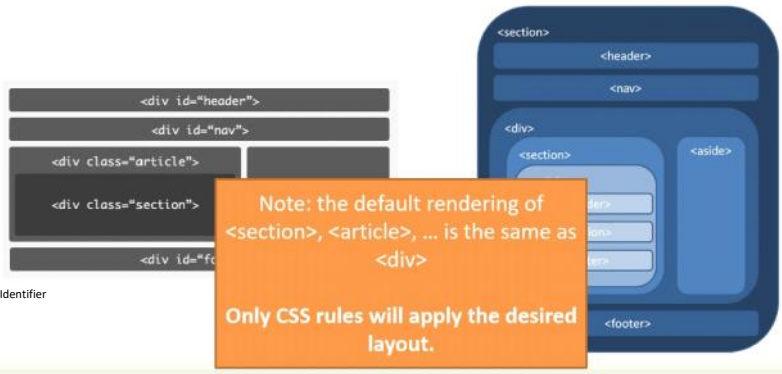
`<div>…</div>`

- Block-level container element
- No predefined semantics
- Invisible layout
- Used to mark-up blocks or groups of blocks, with specific classes

`<span>…</span>`

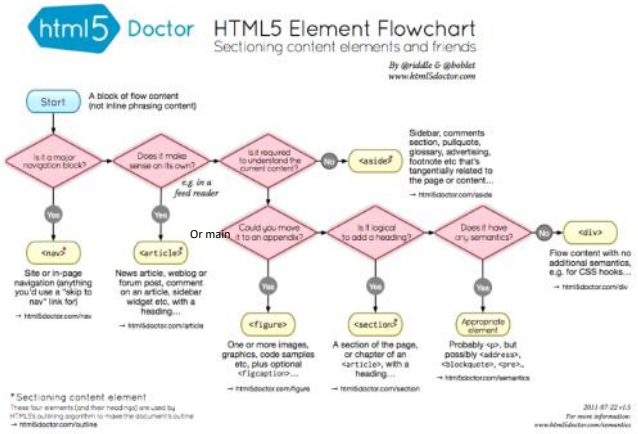
- Inline-level container element
- No predefined semantics
- Invisible layout
- Used to mark-up parts of a paragraph, with specific classes

Opaque vs. Semantic elements

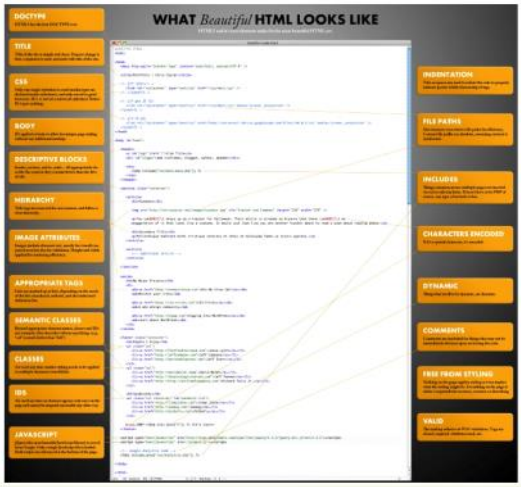


<section>Defined by the language! Do not do anything but clarify!

Just Identifier



Or main



https://css-tricks.com/what-beautiful-html-code-looks-like/

https://css-tricks.com/what-beautiful-html-code-looks-like/

HTML Validator

https://validator.w3.org/

References

- HTML: The Living Standard, Developer's Edition — Last Updated 20 March 2020 - https://html.spec.whatwg.org/dev/
- MDN: HTML: Hypertext Markup Language - https://developer.mozilla.org/en-US/docs/Web/HTML
- A Modern HTML Introductory Tutorial - http://fastwebstart.com/modern-html-tutorial/ (and linked resources)
- https://css-tricks.com/what-beautiful-html-code-looks-like/

License