HTML and CSS

EDAF90 Web Programming
Per Andersson

September 5, 2024

Outline

Terminologi

- client-server (backend, frontend)
- static web page
- dynamic web page
- Content Management Systems (CMS), for example WordPress, Drupal, and Joomla
- singel page web application
- progressive web application
- Web Application Frameworks
- responsive design
- universal design

Standardisering

- Internet Engineering Task Force (IETF) - RFC and "rough consensus and running code"
- World Wide Web Consortium (W3C)
- European Computer Manufacturers Association (ECMA) and ECMAScript

Locales and Word Order

Text communicate infomation to the user. To handle text in a program you need:

- encoding — A mapping (value ↔ symbol)
- locale — How to render dates, digits and time depends on where you are:
  - Digits: 3.142 or 3,142?
  - Date: 01/02/03
    - 3 februari 2001?
    - January 2, 2003?
    - 1 February 2003?
- collation — character order. Is Andersson before or after Åkesson?

Character encoding

There exists many different ways to encode characters

- fixed width
- variable width (compare to Hoffman coding)

Some common standards:

- Unicode and utf8, no just encoding, also collation (sorting)
- ISO-8859-1/latin 1
- UTF8 is conquering the world, it is standard for Java och JavaScript.

Unicode

A standard including:

- visual reference
- set of standard character encodings
- an encoding method
- character properties (lower/upper case)
- rules for normalization, decomposition, collation
- rules for rendering, and bidirectional text display order (right-to-left, left-to-right scripts)

Unicode Blocks (Simplified)

Code	Name	Code	Name
U+0000	Basic Latin	U+1400	Unified Canadian Aboriginal Syllabic
U+0080	Latin-1 Supplement	U+1680	Ogham, Runic
U+0100	Latin Extended-A	U+1780	Khmer
U+0180	Latin Extended-B	U+1800	Mongolian
U+0250	IPA Extensions	U+1E00	Latin Extended Additional
U+02B0	Spacing Modifier Letters	U+1F00	Extended Greek
U+0300	Combining Diacritical Marks	U+2000	Symbols
U+0370	Greek	U+2800	Braille Patterns
U+0400	Cyrillic	U+2E80	CJK Radicals Supplement
U+0530	Armenian	U+2F80	KangXi Radicals
U+0590	Hebrew	U+3000	CJK Symbols and Punctuation
U+0600	Arabic	U+3040	Hiragana, Katakana
U+0700	Syriac	U+3100	Bopomofo
U+0780	Thaana	U+3130	Hangul Compatibility Jamo

Unicode Blocks (Simplified) (II)

Code	Name	Code	Name
U+0900	Devanagari, Bengali	U+3190	Kanbun
U+0A00	Gurmukhi, Gujarati	U+31A0	Bopomofo Extended
U+0B00	Oriya, Tamil	U+3200	Enclosed CJK Letters and Months
U+0C00	Telugu, Kannada	U+3300	CJK Compatibility
U+0D00	Malayalam, Sinhala	U+3400	CJK Unified Ideographs Extension A
U+0E00	Thai, Lao	U+4E00	CJK Unified Ideographs
U+0F00	Tibetan	U+A000	Yi Syllables
U+1000	Myanmar	U+A490	Yi Radicals
U+10A0	Georgian	U+AC00	Hangul Syllables
U+1100	Hangul Jamo	U+D800	Surrogates
U+1200	Ethiopic	U+E000	Private Use
U+13A0	Cherokee	U+F900	Others

The Unicode Encoding Schemes

- each character have a unique unicode
- different ways to store the unique unicodes in a file:
  - UTF-8, UTF-16, and UTF-32.
- UTF-16 used to be standard
- uses 16 bits per character – 2 bytes –
- *FÊTE* `0046 00CA 0054 0045`
- UTF-8 has variable length for each character

UTF-8

Range	Encoding
U-0000 – U-007F	0xxxxxxx
U-0080 – U-07FF	110xxxxx 10xxxxxx
U-0800 – U-FFFF	1110xxxx 10xxxxxx 10xxxxxx
U-010000 – U-10FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

HTML

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
    <link rel="stylesheet" href="css/styles.css">
    <script src="my-awsome-code.js"></script>
    <base  href="https://www.cs.lth.se/eda095/">
  </head>

  <body>
    <h1>Hello World</h1>
    <p id="my-blue-box">My awsome page.
  </body>
</html>
```

HTML - element

Semantic tags

<h1>, <h2>, <p>, <abbr>, <code>, <samp>, <kbd>, <var>, <footer>, <header>, <details>, <nav>. . .

Structure

<table>, <ul>, <ol>, <div>, <span>. . .

Functionality included

<form>, <input>, <select>, <button>, <a>. . .

Learn more about HTML tags

https://developer.mozilla.org/en-US/docs/Web/HTML/Element

https://www.w3schools.com/tags/default.asp

HTML - elements

Data:

- between the tags: `<h1>My Headline</h1>`
  - is rendered
  - text
  - may include other html-elements
- attributes: `<a href="http://cs.lth.se">my link</a>`
  - text is not shown on screen
  - only text
- `id` - optional attribute, unique for each element
  can be used to find/refer to an element
- `class` - used for styling (do not relate to JavaScript classes)
- `name` - reference in some context, for example in `<form>`
- `aria-label` - aid for screen readers, when no other textual representation exists

tag + content $\approx$ element

HTML

```
<form>
  <div>
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" id="exampleInputEmail1" placeholder="name@example.com"
    aria-describedby="emailHelp">
    <div id="emailHelp">We’ll never share your email with anyone else.</div>
  </div>
  <div>
    <label for="exampleInputPassword1">Password</label>
    <input type="password" id="exampleInputPassword1">
  </div>
  <div>
    <input type="checkbox" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit">Submit</button>
</form>
```

Layout

Rendering

- the rendering is controlled by
  - element tag: `<p>`, `<select>`
  - properties: `font-family`, `background-color`
- some properties are dynamic, updated by the browser rendering engine
- most properties are inherited from the surrounding
- properties can not be deleted, only shadowed
- give an property a value:
  - the `style` attributet in the HTML element
  - Cascading Style Sheets (CSS)

example of what you can do with css: https://www.w3schools.com/css/css_intro.asp

CSS

Cascading Style Sheets

- separate the content from the layout
- a set of rules:
  - selection
  - declaration (attribut = value)
- the declaration is applied to all elements matching the selection

syntax:

```
urval : { property1: value1; property2: value2;}
```

CSS - selection

CSS selection is based on pattern matching:

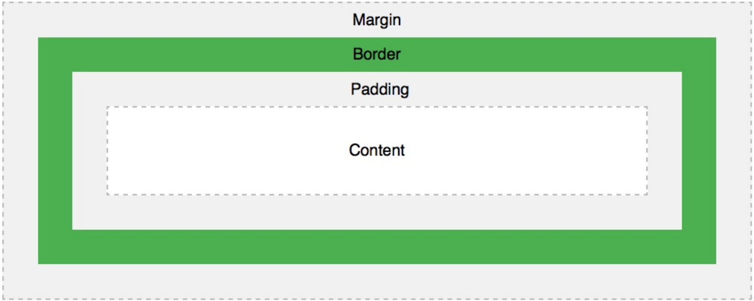
- instances of an element: `<h1>`
- all elements with a class: `<div class="my-style">`
- the element with a given `id`: `<div id="my-tag">`
- `pseudo classes` `focus`, `hover`, `visited`, `valid`, ...
- `pseudo element` `nth-child(2)`, `only-child`, ...
- `attribute value:` `[title~="flower"]`, ...

Match patterns can be combined

CSS - exempel

```
// element
div { color: blue; border: 1px; }
// id, <p id="my-blue-box">
#my-blue-box { background-color: lightblue; }
// class, <div class="center">
.center { text-align: center; color: red; }
// element och and, <p class="center">
p.center { text-align: center; color: green; }
// inside, p is a decendant of div
div p { text-align: center; color: green; }
// p witch is a direct child of a div
div > p { text-align: center; color: green; }
// directly after
div + p { text-align: center; color: green; }
// pseudo-class
a:hover { background-color: lightblue; }
```

Box Modellen



Layout

CSS Properties for layout

- `display`: `block`, `inline`, `none`, `flex`, `...`
- `visibility`: `visible`, `hidden`, `...`
- `position`: `static`, `relative`, `absolute`, `fixed`, `...`
- `overflow`: `visible`, `hidden`, `scroll`, `auto`, `...`
- `z-index`: `auto`, *number*

Frameworks

Creating a good layout is costly.

- needs a lot of testing on different browsers
- you can use or extend use existing:
  - bootstrap
  - material design

My own standard

Each browser have their own implementation of the

- rendering engine
- JavaScript engine

With their own

- interpretation of the standard
- selection of standard features to support
- bugs
- extensions

Webkit Mozilla

The same feature apperas with different names in different browsers:

- `box-shadow`
- `-webkit-box-shadow`
- `-moz-box-shadow`

DOM

Document Object Model

- a web page/html is a tree
- the nodes are the HTML elements
- HTML attriburtes are attributes in the nodes
- `<html>` is the root of the tree

DOM

Document Object Model

- `Document` is a class for representing the DOM
- the nodes inherits from the `Element.prototype` object
- the globala variable `document` refers to the DOM
- API for
  - navigate in the tree: `document.body.getElementsByTagName( 'H1' )`
  - serach for elements: `document.getElementById("intro")`
  - modify the DOM: `Element.innerHTML`
  - read/writer attribute, `myInputElement.value="Nisse Hult"`

jQuery

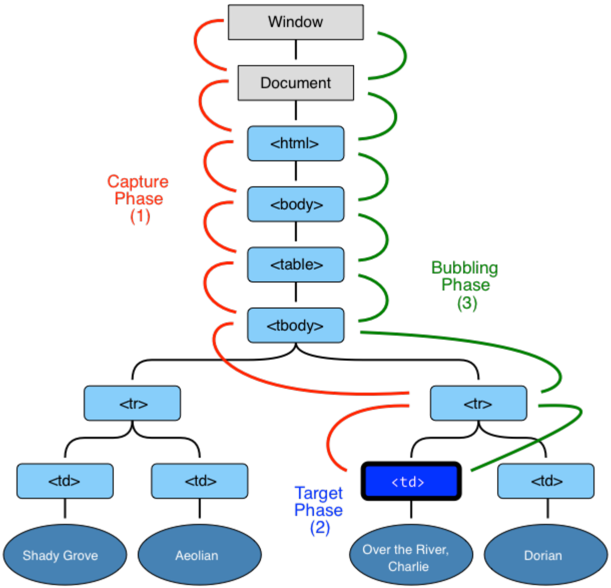
(not part of the course)

- jQuery is an old library for simple access to and modification of the DOM
- deprecated use react, vju, angular, or any modern framework
- common to find references in examples and on Stack Overflow et.c.
- all functions are place under $ in the global namespace
- now you can guess what `$(".test").hide()` does
  Hint: jQuery use the same pattern matching syntax as css

Events

- the browser creates events: `blur`, `submit`, `resize`, `keydown`
- call-back-methods
  - `<p id="demo" onclick="myHandler(event)">`
  - `addEventListener(eventType, handler[, options])`
- `event` is an instance of `Event`
- event propagates trough the DOMen, three phases:
  1. capturing
  2. target
  3. bubbling

event phases



Events

- not all events propagate, `focus` do not.
- `this===event.currentTarget`, the DOM element containing the handler
- `event.target` the source of the event, a DOM element
- event propagates trough the DOMen, three phases:
  1. capturing
  2. target
  3. bubbling
- `event.stopPropagation()`
- `event.preventDefault()`

Forms

```html
<form onsubmit="myFunction(event)">
  <label for="id-checkbox">Checkbox:</label>
  <input type="checkbox" id="id-checkbox"/>
  <input type="submit" value="Send Request">
</form>
```

Form submission is default behaviour for many events (click on submit button, enter in input field)

- submit the form using HTTP
- the server responds with a new html-page
- the browser renders the new page