

Development of Web Applications

Principles and Practice

Vincent Simonet, 2013-2014

Université Pierre et Marie Curie, Master Informatique, Spécialité STL



4

Client Technologies

Vincent Simonet, 2013-2014

Université Pierre et Marie Curie, Master Informatique, Spécialité STL

Today's agenda

- HTML, CSS and DOM,
 - JavaScript,
 - AJAX,
 - JSONP,
 - HTML5,
 - WebSockets.
-

HTML, CSS and DOM

HTML HyperText Markup Language

Main markup language for creating web pages and other information that can be displayed in a web browser.

Terminology:

- **Element:**

- `<a href="http://example/">example</a>`

- **Tag:** `a`

- **Attribute:** `href="http://example/"`

- **Character entity reference:** `%` `&`

History:

First version in 1991 (Berners-Lee), HTML 4 in 1997, HTML 5 planned for 2014.

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type"
          content="text/html; charset=utf-8">
    <link rel="stylesheet" type="text/css"
          href="resources/main.css">
    <link rel="shortcut icon"
          href="resources/icon.png">
    <script type="text/javascript"
            src="resources/main.js"></script>
    <title>Title of the Page</title>
  </head>
  <body onload="vtst.onload();" lang="fr">
    ...
  </body>
</html>
```

DOM Document Object Model

A language-independent API for manipulating HTML and XML documents.

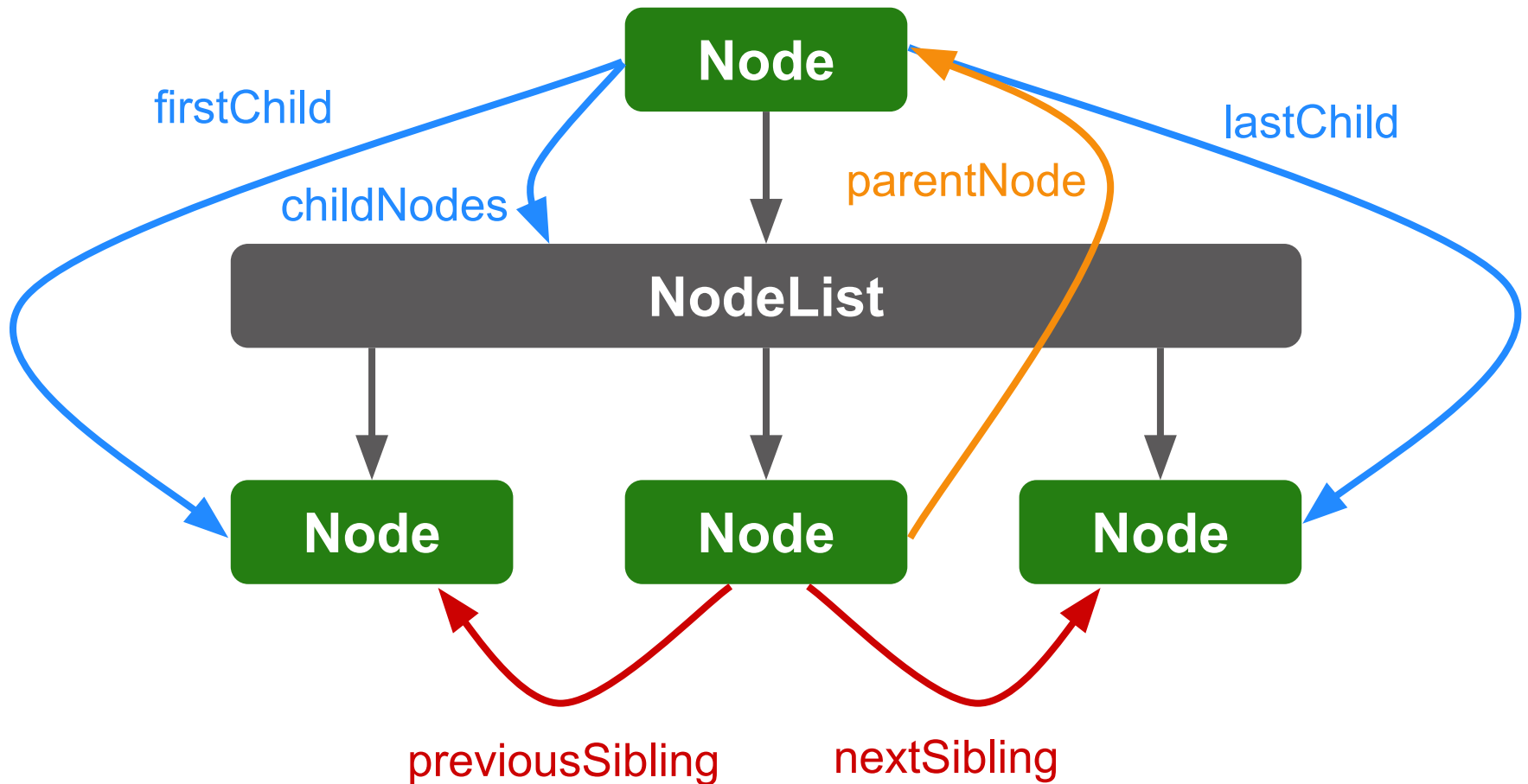
The HTML/XML document is manipulated as a tree, whose nodes are elements, attributes and text.

DOM also supports an event model.

4 levels (0 to 3).

Level 1: 1998, Level 2: 2000, Level 3: 2004.

DOM Tree



DOM Example

HTML Document:

```
<body>  
  <a href="hello.html">Hello!</p>  
</body>
```

Equivalent in DOM:

```
a = document.createElement("a");  
t = document.createTextNode("Hello!");  
a.setAttribute("href", "hello.html");  
document.body.appendChild(a);
```

DOM Events (level 0)

Inline model (in HTML):

```
<a href="..."  
  onclick="alert('Hello!');">Hello</a>
```

Traditional model (in JavaScript):

```
element.onclick = function() {  
  alert('Hello!');  
  return false; // prevent the default action  
}
```

DOM Events (level 2)

```
document.addEventListener(  
    "click",  
    function(event) {  
        alert('Hello!');  
        event.preventDefault();  
    }, false);
```

DOM Events (level 2): propagation

Consider 2 elements nested together:

```
<a><b> . . . </b></a>
```

When the user clicks on the inner element, there are two possible ways to handle it:

- Trigger the elements from outer to inner (a, then b):
capture (pass `true` to `addEventListener`)
- Trigger the elements from inner to outer (b, then a):
bubbling phase (pass `false` to `addEventListener`)

You may call `event.stopPropagation()` in an event handler to stop capture or bubbling.

CSS Cascading Style Sheets

A style sheet language, used for describing the presentation semantics (the look and formatting) of a document written in a markup language (e.g. HTML, but also SVG, XUL, etc.)

5 levels (1, 2, 2.1, 3 and 4):

- Level 1 was published in 1996,
 - Level 2.1 is the current standard (1998-2011),
 - Level 3 is under implementation by most major browsers (2012-...),
 - Level 4 is under development.
-

CSS Example

```
#name .class {  
    font-family: sans-serif;  
    color: red;  
    background-color: #0f0;  
}
```

```
a { text-decoration: none; }  
a:hover { text-decoration: underline; }
```

The power of CSS selectors in DOM

```
subElement =  
    element.querySelector("#id .class");  
subElements =  
    element.querySelectorAll("a.class")
```

Available in all modern browsers. Emulated implementations exist for older browsers.

More about CSS

- CSS Zen Garden: a single HTML page styled with plenty of different style sheets.
 - Syntactic extensions of CSS (that compile to CSS):
 - [Sass](#)
 - [LESS](#)
-

JavaScript

JavaScript

A **prototype-based** scripting language, with **dynamic typing** and **first-class functions**.

In practice, three layers:

- The JavaScript programming language (ECMAScript),
 - Standardized JavaScript APIs (DOM, AJAX, Canvas, etc.),
 - JavaScript frameworks/libraries (jQuery, Prototype, Dojo, YUI, Closure, etc.).
-

Prototypes

JavaScript uses **prototypes** where many other OO languages use **classes** for inheritance.

A prototype is a function with a `prototype` property. A method is a function.

```
MyClass = function() {  
    // constructor  
};  
  
MyClass.prototype.myMethod = function() {  
    // method  
};  
  
var myObject = new MyClass();  
myObject.myMethod();
```

this

In the scope of a function, the keyword **this** refer to the object to which the function belongs to.

```
MyClass = function() { ... };
```

```
MyClass.prototype.myMethod = function() { ... };
```

```
var myObject = new MyClass();
```

```
myObject.myMethod(); // this is myObject
```

```
var fn = myObject.myMethod;
```

```
fn(); // this is not myObject
```

```
var fn2 = myObject.myMethod.bind(myObject);
```

```
fn2(); // this is myObject
```

JavaScript minification and compilation

Why? Reducing the size of the JavaScript source code, for speeding up download, parsing and evaluation.

How?

- Removing whitespace and comments,
- Renaming variables,
- Removing unused code

Tools: Closure Compiler, YUI Compressor, minify, etc.

AJAX

AJAX Asynchronous JavaScript and XML

With AJAX, a JavaScript script can send data to, and retrieve data from, a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

Despite the name, the use of XML is not required. In fact, JSON is much more popular.

AJAX call example

```
var xhr = new XMLHttpRequest();
xhr.open('get', 'http://example/method');

xhr.onreadystatechange = function() {
    // Ready state 4 means the request is done
    if (xhr.readyState === 4) {
        if(xhr.status === 200){
            alert('Success: ' + xhr.responseText);
        } else {
            alert('Error: ' + xhr.status);
        }
    }
}

xhr.send(null);
```

AJAX: XML or JSON response

XML response:

```
xhr.responseText = "document";  
xhr.responseXML.documentElement
```

JSON response:

```
xhr.responseText = "json";  
eval(xhr.responseText)  
(if you trust the response source!).
```

Same origin policy

AJAX requests can be made only to URLs of the same domain (host and port) as the page.

AJAX is hence useful for communication with the server of a web application, but not for doing calls to a third-party API.

For remote API calls, several workarounds are used:

- JSONP (by far the most popular),
 - Using the application server as a proxy (costly),
 - iframes / using the URL to communicate (tricky),
 - Messages (the clean way, in HTML5).
-

JSONP

JSONP JSON with Padding

An alternative to AJAX, for requesting data from a server in a different domain.

How it works?

- The client script generates the request by adding a `<script>` tag to the page:

```
<script type="application/javascript"
      src="http://directory/?id=42">
```
 - The server returns a JavaScript containing a JSON value, wrapped into a function call (the padding):

```
callback({"id": 42,
          "name": "Vincent Simonet"});
```
-

JSONP in practice

- The name of the padding is usually passed as an argument in the request:

```
<script type="application/javascript"
        src="...?id=42&jsonp=mycallback">
mycallback({"id": 42,
            "name": "Vincent Simonet"});
```

- JavaScript frameworks provide helper functions for making this transparent. E.g. in jQuery:

```
$.ajax({url : 'http://.../?id=42',
        dataType : 'jsonp',
        jsonp : 'jsonp',
        success : function(data) {}
    });
```

JSONP limitations

- Only GET (POST is doable, but tricky),
 - No access to HTTP headers (in request and response), including cookies.
-

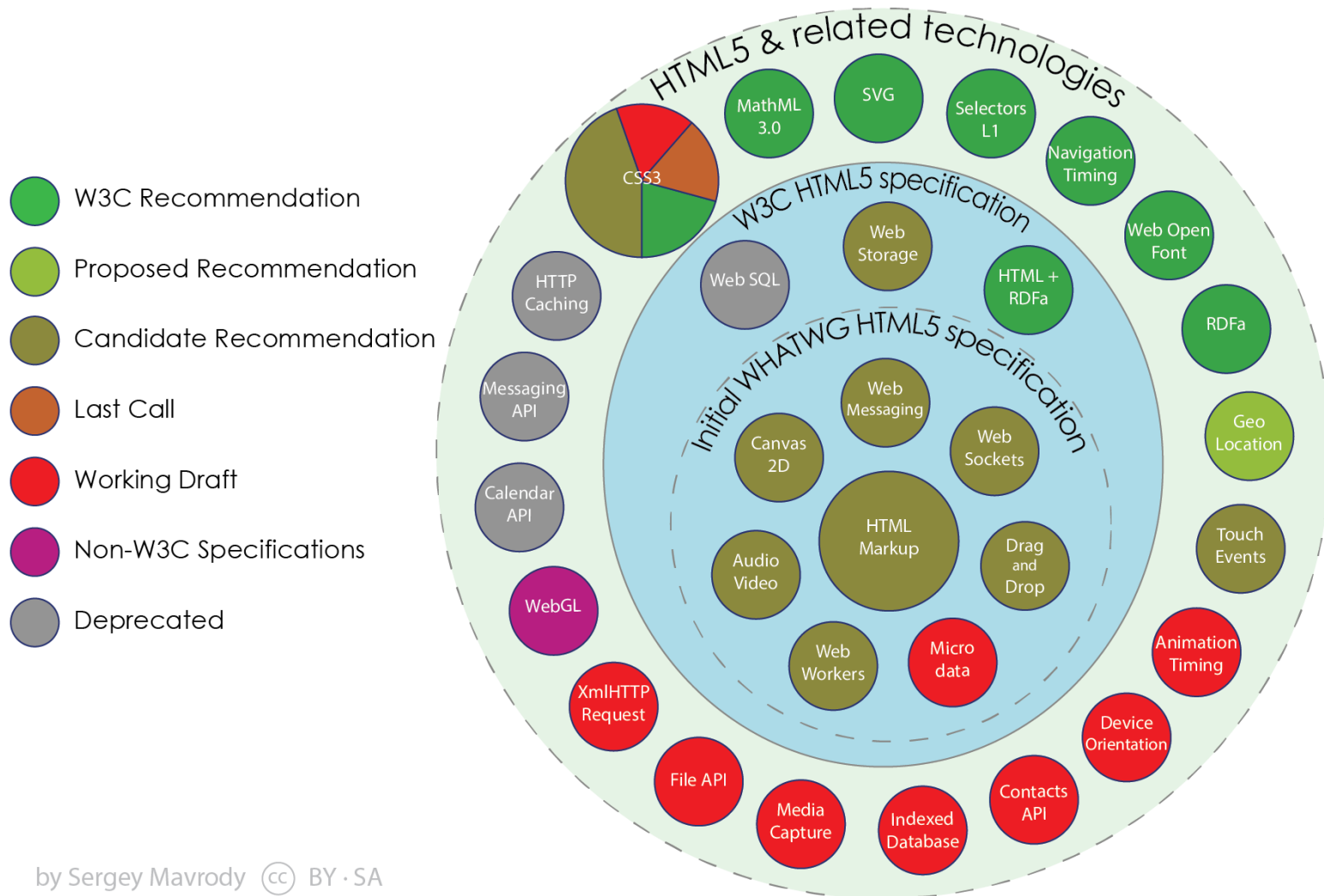
HTML5

What is HTML5?

The 5th version of the HTML language, subsuming HTML 4.01 and XHTML 1.1

New/extended markup, and a galaxy of APIs.

Specification status at the beginning of 2013



Implementation status

Source: html5test.com

Chrome 29	463/500
Opera 16	442/500
Firefox 24	414/500
Safari 6.0	378/500
Internet Explorer 10	320/500

Main HTML5 features

- **Semantic tags,**
 - Canvas,
 - Video,
 - Geo-localisation,
 - Local storage,
 - Offline,
 - **Improved forms,**
 - **Microdata,**
 - History manipulation.
-

Tags in HTML5

- **Semantic replacements of <div> or :**
<nav> <header> <footer> <section>
<hgroup> <article> <aside> <time>
<mark>
- **Replacements of <object>:**
<audio> <video>
- **Removal of some style tags:**
 <center> <strike> <tt>

(non-exhaustive list)

Improved forms

- **New input types:** color, date, datetime, datetime-local, email, month, number, range, search, tel, time, url, week

```
<input type="color" name="favcolor">
```

```
<input type="number" name="quantity"  
min="1" max="5">
```

- **New input attributes:** autocomplete, autofocus, multiple, min, max, pattern, required, etc.
- **New elements:** <datalist> <keygen>
<output>

Use them!

WebSockets

The problem

In HTTP and AJAX, all exchanges are initiated by **client** requests.

In some applications, it is useful to have the server pushing information to the client. E.g.:

- Notifications in a news website,
 - Messages in a chat system,
 - etc.
-

Workaround solutions

- The client can make periodic requests to the server,
- The client can make a request to the server, which will answer with an "infinite" response. Known as Comet.

Several implementations:

- Streaming (using iframe or special XmlHttpRequest),
 - Long polling (using XmlHttpRequest or script tags).
-

The HTML5 solution: WebSockets

WebSocket is is a protocol providing full-duplex communications channels over a single TCP connection.

Enables a stream of *messages*.

Its only relationship to HTTP are:

- its handshake is interpreted by HTTP servers as an Upgrade request,
 - it is using port 80 as default port.
-

WebSocket protocol handshake

Request:

```
GET /mychat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: x3JJHMBDL1EzLkh9GBhXDw==
Sec-WebSocket-Protocol: chat
Sec-WebSocket-Version: 13
Origin: http://example.com
```

Response:

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: HSmrc0sMlYUkAGmm5OPpG2HaGWk=
Sec-WebSocket-Protocol: chat
```

Client-side JavaScript API

```
var connection = new WebSocket('ws://.../echo',  
    ['soap', 'xmpp']);
```

```
connection.onopen = function () {  
    connection.send('Ping');  
};
```

```
connection.onerror = function (error) {  
    console.log('WebSocket Error ' + error);  
};
```

```
connection.onmessage = function (e) {  
    console.log('Server: ' + e.data);  
};
```

Server-side implementations

- Java: [Jetty](#)
- Node.js: [ws](#), [WebSocket-Node](#)
- Python: [pywebsocket](#)