JAVASCRIPT

Amir DJOUAMA

Plan

- Introduction
- Syntax
- Types and Objects
- DOM (Document Object Model)
- Events
- Forms
- Javascript and HTML5
- advanced mode

Required knowledge

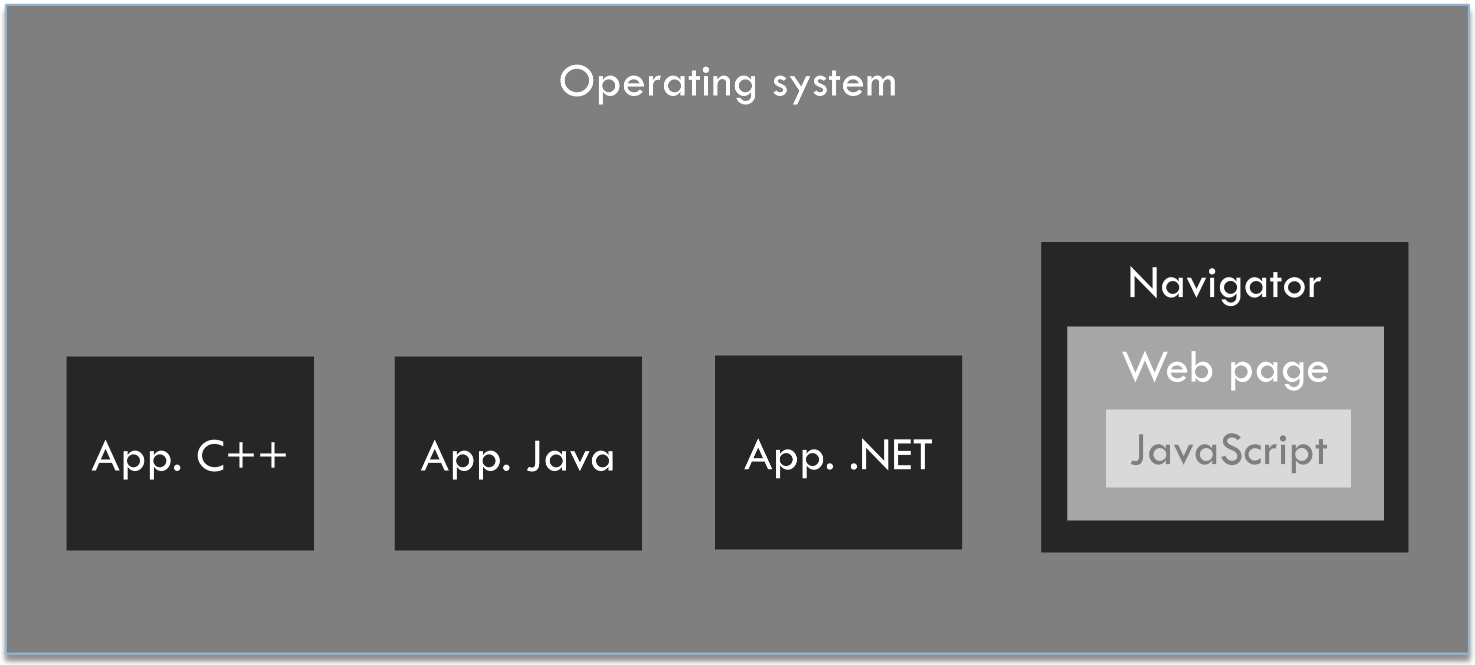
- No need to be a programmer
- Useful to know the fundamentals
- Based on C, C++, C#
- Usable with HTML and CSS

Introduction

JavaScript

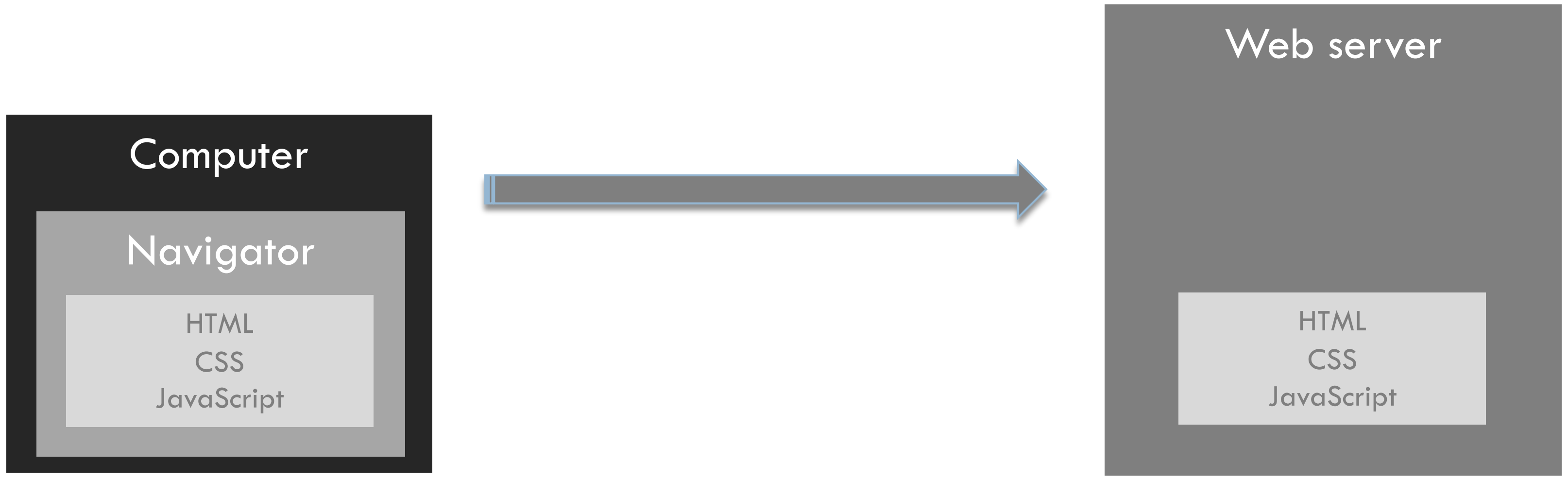
- HTML (HyperText Markup Language)
  - Content
- CSS (StyleSheet Language)
  - Presentation
- JavaScript (programming language)
  - Behavior

What is a scripting language?



Does not directly access the DB
Cannot access hardware

JavaScript client side language



JavaScript can be disabled

PHP, ASP.NET, Ruby on Rails

Historical

- 1995 → LiveScript
- 1996 → JavaScript (NetScape 2)
  - IE 3 JScript
- 1997 → ECMAScript
- 1999 → ECMAScript 3
  - IE, Firefox, Chrome, Opera , Safari…etc.
- 2009 → ECMAScript 5 released
- 2010 → Multiplication of JS frameworks
- 2013 → Facebook releases ReactJS
- 2015 → Release of ECMAScript 6
- Last → ECMA-262 (13th edition, June 2022)

What do we need?

- Mac, PC, Linux, UNIX
- ASP.NET, Ruby on Rails, ColdFusion , PHP
- Xcode , Visual Studio, Eclipse, NetBeans , TextMate , Coda

A simple text editor.

Example

```
<html>
        <head>
                <title>Single Page</title>
        </head>
        <body>
                <h1>Simple HTML page</h1>
                <p>
                        This is a very simple HTML page.
                </p>
                <p>It contains:<p>
                <ul>
                        <li>An H1 tag</li>
                        <li>Two paragraphs</li>
                        <li>An unordered list </li>
                </ul>
        </body>
</html>
```

Example

```
<html>
          <head>
                    <title>Single Page</title>
          </head>
          <body>
                    <h1>Simple HTML page</h1>
                    <p>
                              very simple HTML page .
                    </p>
                    <p>It contains:<p>
                    <ul>
                              <li>An H1 tag</li>
                              <li>Two paragraphs</li>
                              <li>An unordered list </li>
                    </ul>
                    <script>
                              alert ("Welcome to JavaScript training !");
                    </script>
          </body>
</html>
```

Javascript

App. iPhone	App. .NET	JavaScript
Objective C Mac OS X xcode	C# or VB.NET Windows VisualStudio	..?

It doesn't matter what you use!

Dreamweaver. Xcode . Visual Studio. TextMate . NetBeans . vi. Emacs.

BBEdit . SlickEdit . TextWrangler . Aptana . Komodo . NotePad ++...

Syntax

Structure of JavaScript

- Is an interpreted language
- And not compiled, unlike other languages
- Just write the code and run it on the browser
- It is case sensitive (upper and lower case) unlike HTML

  `alert("text")` not `Alert("text")`

Semicolon

- A statement ends with a ";"
  - Used to separate two statements on the same line
- Example

```
alert("hello"); alert("and welcome");
```

And the space?

- Insensitive to space
- Example

```
alert("Hello");

alert
( "Good morning"
);

alert("Hello and welcome");
```

Comments

```
// This is a comment
alert ("Hello"); // Can go here

/* A comment
on several
lines
*/
```

Execution order

- Sequentially, but when?
- Example `scriptPosition.html`
  - In `head`
  - In `bodysuit`

Where to put javascript ?

- You can put it in a separate file
- Example:

```
<script>
 alert ("Hello");
</script>
<script src =" myscript
</script>
```

myscript.js

```
alert ("Hello");
```

- Concretely:
  - Preferably at the end of the page

script tag attribute

```
<script src ="myscript.js" type=" text/javascript ">
</script>
```

- But also
  - `type="text/ecmascript"`
  - `type="application/javascript"`
  - `type="application/ecmascript"`
- **Default:** `type="text/javascript"`

Variables

- To create a variable:
  - `var year;`
  - `var emailClient;`
  - `var dateDay;`
  - `var x;`
  - `var 09pbm, $pbm;` X

- `var` is not required

Attention

```
var x = 200;
X = 210
```

- Two variables will be created `x` and `X` and no error will be reported

Data type

- There are no variable types
- `var myVariable`
- `myVariable = 200;`
- `myVariable = "Hello";`
- `myVariable = 'Hello'`
- `myVariable = true;`
- `myVariable = false;`

Condition

```
if(condition(s)) {
// Some code here
// …
} else {
// Other code
if() {
// Another process
}
}
```

- Or

```
conditions ? right : wrong
```

Operators and Expressions

- Arithmetic operators (+ - * / %)
- Assignment (=)
- Example
  - `score = score + 10; score += 10;`
- Priority
  - `result = 5 + 5 * 10`
  - `result = (5 + 5) * 10`

Operators and Expressions

- Equality (`a == b`), strict equality (`a === b`)
- Example

```
var a = 5, b = "5"
if(a == b) // (a === b)
 alert ("equals");
else
 alert ("different");
```

Operators and Expressions

- Comparison:

```
if (a == b) { …
if (a != b) { …
if (a === b) { …
if (a !== b) { …
if (a > b) { …
if (a < b) { …
if (a >= b) { …
if (a <= b) { …
```

Operators and Expressions

- Logic (AND / OR)

```
if (a === b && c === d) { …
if (a === b || c === d) { …
if ( (a > b) && (c < d) ) { …
if (
(a > b)
&&
(c < d) ) { …
```

Operators and Expressions

- Modulo (%)
- Example

```
var year = 2023
var rest = year % 3; // remainder is 1
```

Operators and Expressions

- Increment / decrement
- Example :

```
a = a + 1; a = a - 1;
a += 1; a-=1;
a++;
++a;
```

Console

```
console.log ("message" or variables);
console.info ("message" or variables);
console.warn ("message" or variables);
console.error ("message" or variables);
```

while loop

- Example

```
let a = 1;
while (a < 10) {
 console.log(a);
a++;
}
```

do… while loop

- Example

```
let a = 1;
do {
 console.log (a);
 a++;
} while(a<10);
```

The block will be executed at least once

for loop

- Example

```
for(var i = 1; i < 10; i++) {
// …
}
```

Break

- Example

```
for (var i = 1; i < 5000; i++) {
// …
    if (i == 101) {
        break;
    }
// …
}
```

Continue

- Example

```
for (var i = 1; i < 5000; i++) {
// …
    if (i % 5 == 0) {
        continue;
    }
// …
}
```

`continue` returns to the beginning of the loop

Functions

```
function myFunction () {
   // …
}
```

- The call:

```
myFunction ();
```

- Declare them before using them

Functions with parameters

```
function myFunction (param1[,param2 …]) {
    // …
    // possibly return;
}
```

Functions – special cases

```
function calculation( code, month,
interest, name ) {
    // lots of code
}


calculation(145,8,4,"Omar Kamel"); // correct
calculation (145,8,4,"Omar Kamel", " something
more")
// Extras will be ignored
calculation (145.8) // missing ones will be "
undefined "
```

Types and Objects

The tables

- A single variable with multiple values

```
var valMultiple = [];
   valMultiple [0] = 50;
   valMultiple [1] = 60;
   valMultiple [2] = "character";

 var valMultiple = [50,60,"character"];
```

The tables

```
var valMultiple = [];
var valMultiple = new Array();
var valMultiple = Array();
var valMultiple = Array(5);
```

Arrays – properties and methods

- Table length

```
var valMultiple = [10,20,30,40,50];
console.log(valMultiple.length);
```

- Reverse an array

```
var valMultipleInv = valMultiple.reverse();
```

- Create a character string

```
var string = valMultiple.join(); //"10,20,30,40,50"
```

Paintings are everywhere

- **Example**

```
let myLinkArray =
document.getElementsByTagName("a");
// get a list of links from the page
```

Numbers

- Using numbers is very easy
- javascript numbers are 64 bit float numbers

Addition and concatenation

```
let a = 5;
var b = 5;
console.log(a+b); //10

var a = "5";
var b = "5";
console.log(a+b); //55
```

Addition and concatenation

```
let a = 5;
var b = "5";
console.log ( a+b ); // 55 one is

let a = 5;
var b = "c";
console.log (a*b); // NaN
```

Addition and concatenation

- To remedy

```
var a = "55"; // maybe "abc"
var myNumber = Number(a); //create a number
```

- If it does not work

```
if(isNaN ( myNumber )) {
 console.log ("not a number");
}
```

The MATH object

- Very useful for maths, conversions, calculations…
- Examples

```
var x = 200.6;
var y = Math.round(x); // 201

let a = 200, b = 1000, c = 2;
var big = Math.max ( a,b,c ); //
Math.min (a,b,c)
Math.PI, Math.random().sqrt(), .log()
```

Strings - Strings

- Works with quotes, quotes but not a mix of both
- Be careful in the middle

```
Exp : var phr = 'friend of javascript ';
```

- To remedy:

```
var phr = " javascript's friend ";
var phr = 'friend of javascript ';
```

String – properties and methods

- Length

```
console.log ( phr.length );
```

- All upper/lower case

```
phr.toUpperCase() / toLowerCase ();
```

- Split a string

```
var word = phr.split(" "); // an array of
words
```

- Position

```
var position = phr.indexOf("of"); //6
```

- There is also `.lastIndexOf`

String – properties and methods

- Cut a chain

```
var phr = "Another sentence.";
var segment = phr. slice(2,5);
```

  - Same as `.substring()` or `.substr()`

String – Comparison

```
var str1 = "Hello";
let str2 = "hello"; // str1 != str2


if(str1.toLowerCase() ==
str2.toLowerCase()) {
 console.log ("ok");
}
```

- You can also use < > == ...

String – reference

- developer.mozilla.org/en/JavaScript/Reference _ _ _

The dates

- Dates can be manipulated easily
- EXP :

```
var ajd = new Date(); // current date and time
var d = new Date(2015,0,1);
// year , month, day, hours, minutes, seconds
//january starts at 0
```

Date methods

```
var ajd = new Date();
ajd.getMonth (); // returns 0 - 11
ajd.getFullYear (); // YYYY
ajd.getYear (); // depreciate
ajd.getDate (); // 1 - 31 day of the month
ajd.getDay (); // 0 - 6 weekday, 0 = Sun
ajd.getHours (); // 0 - 23
ajd.getTime (); // milliseconds since 1/1/1970
```

The dates – comparison

```
var date1 = newDate(2015,0,1);
var date2 = newDate(2015,0,1);


if (date1 == date2) { … // false!
```

- These are two objects, complex value!

```
if (date1.getTime() == date2.getTime() ) {
…
// true !
```

objects in javascript

```
   var playerName = "Omar";
   var playerScore = 10000;
   var playerRank = 1;
```

- You can create a container

```
var player = new Object();
 player.name = "Omar";
 player.score = 10000;
 player.rank = 1;
```

- We no longer speak of variables but of properties

objects in javascript

- You can also use another syntax:

```
var player1 = {name:"Omar",score:10000,rank:1};
var player2 = {name:"Samy",score:1000,rank:13};
```

objects in javascript

- To create the methods

```
function ShowDetails () {
 console.log ( this.name +" is at the
rank of "+ this.rank );
}

player1.logDetails = ShowDetails ;
player2.logDetails = ShowDetails ;
```

The call is made with: `player1.logDetails();`

DOM (Document Object Model)

What is DOM?

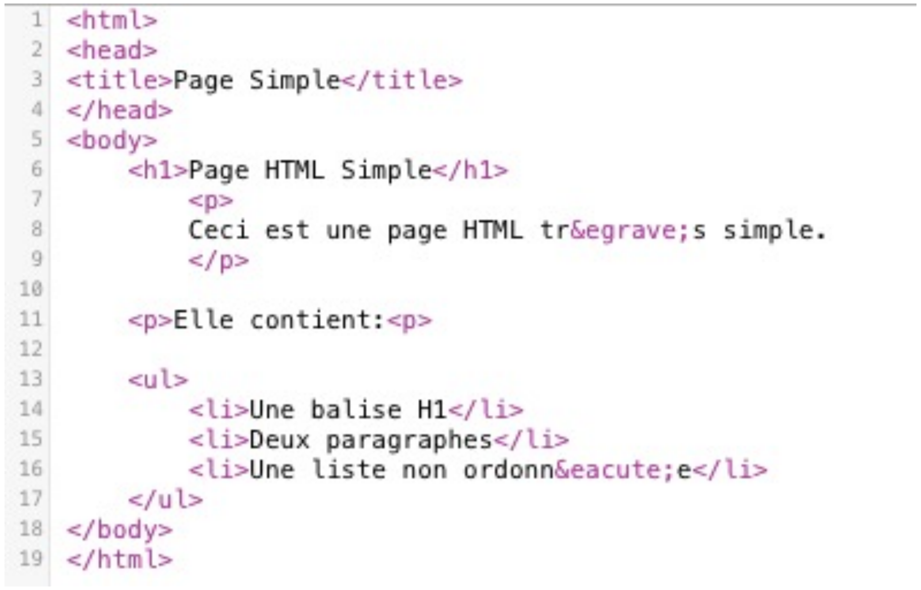
- Is a W3C standard that describes an interface independent of any programming language and any platform,
- Allowing computer programs and scripts to access or update the content, structure or style of XML and HTML1 documents.
- The document can then be processed and the results of this processing can be reincorporated into the document as it will be presented.

Source: Wikipedia

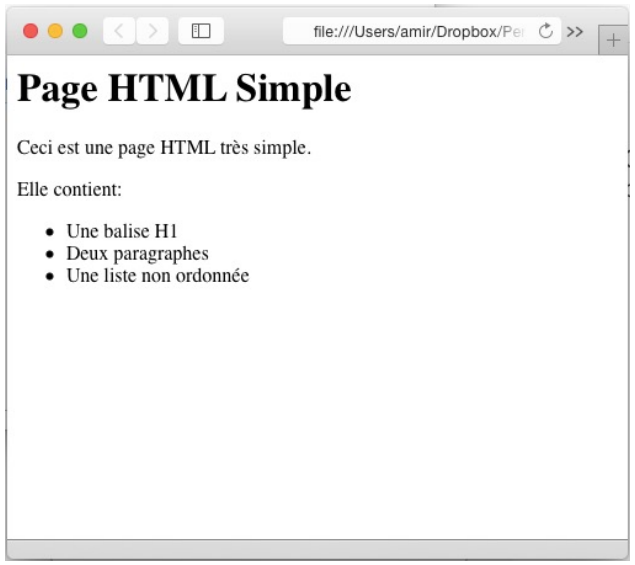
What is DOM?

Document

- Sets the page (not the site)
- A single document but several representations



```
<html>
<head>
<title>Page Simple</title>
</head>
<body>
    <h1>Page HTML Simple</h1>
        <p>
        Ceci est une page HTML tr&egrave;s simple.
        </p>

    <p>Elle contient:<p>

    <ul>
        <li>Une balise H1</li>
        <li>Deux paragraphes</li>
        <li>Une liste non ordonn&eacute;e</li>
    </ul>
</body>
</html>
```



Page HTML Simple

Ceci est une page HTML très simple.

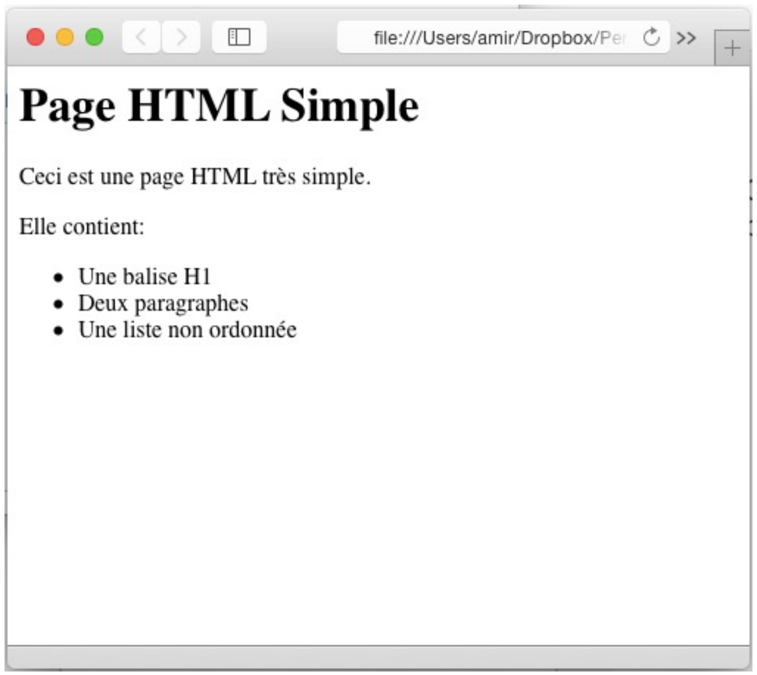
Elle contient:

- Une balise H1
- Deux paragraphes
- Une liste non ordonnée

What is DOM?

Object

- The elements that make up a page (a document)
- Tables, Dates, character strings …



Page HTML Simple

Ceci est une page HTML très simple.

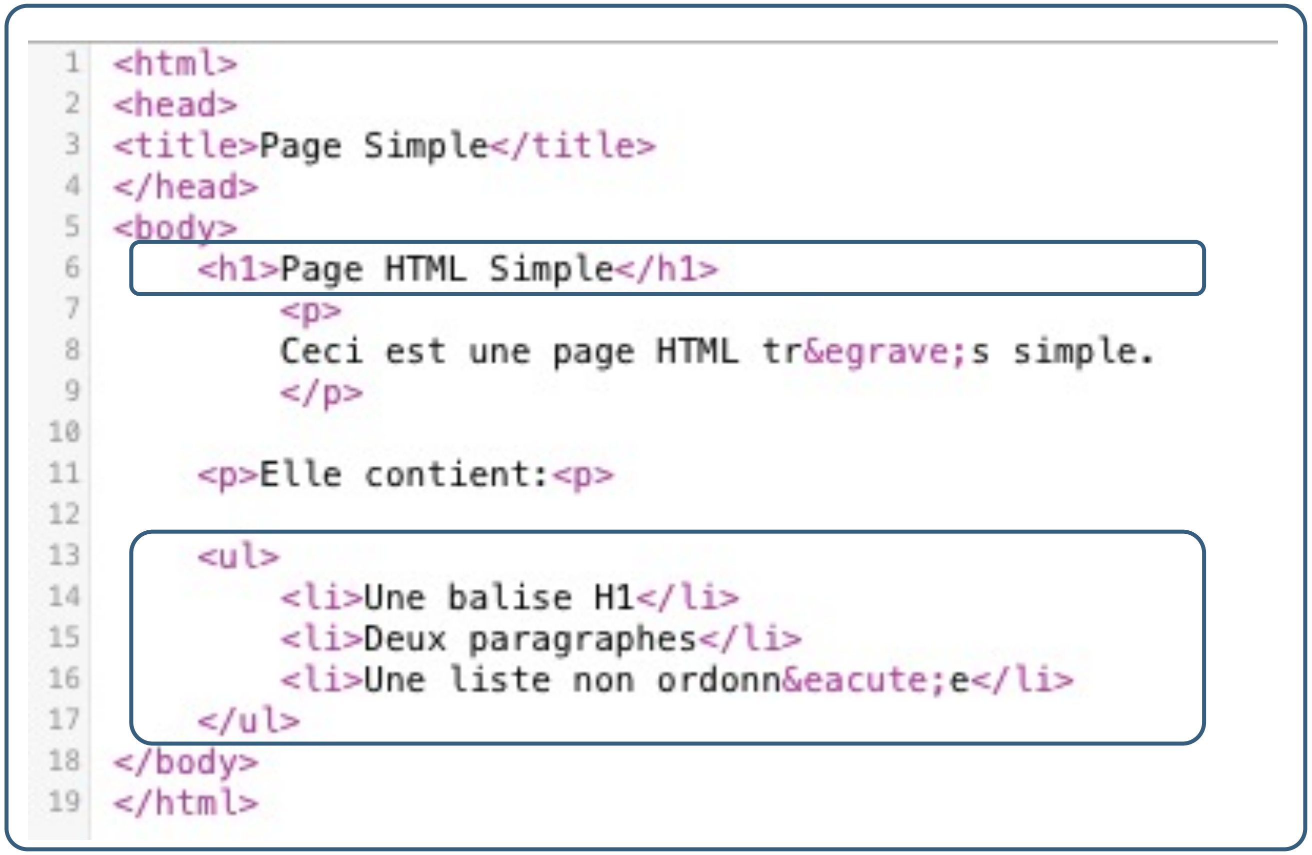
Elle contient:

- Une balise H1
- Deux paragraphes
- Une liste non ordonnée

What is DOM?

Object

- Vision according to a developer

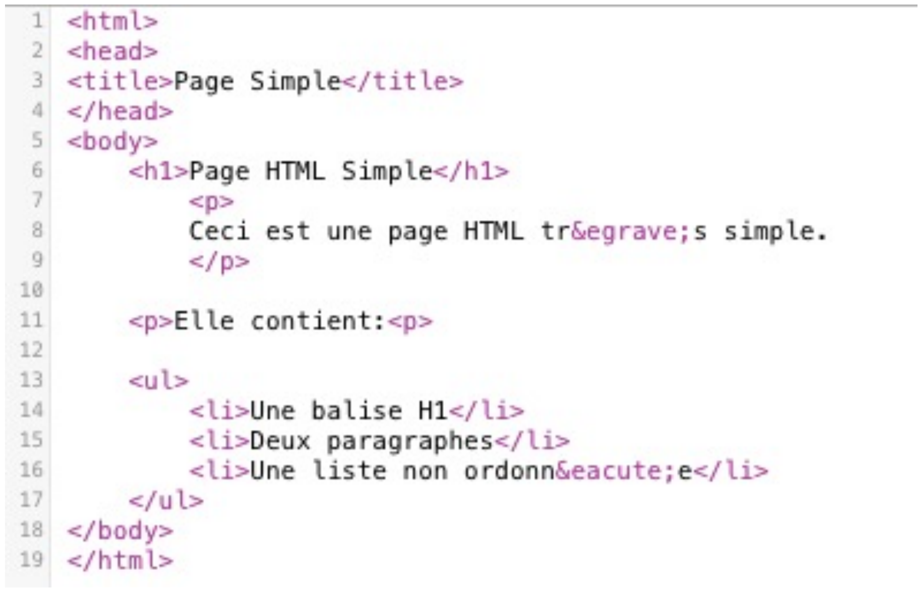


```
<html>
<head>
<title>Page Simple</title>
</head>
<body>
    <h1>Page HTML Simple</h1>
        <p>
        Ceci est une page HTML tr&egrave;s simple.
        </p>

    <p>Elle contient:<p>

    <ul>
        <li>Une balise H1</li>
        <li>Deux paragraphes</li>
        <li>Une liste non ordonn&eacute;e</li>
    </ul>
</body>
</html>
```

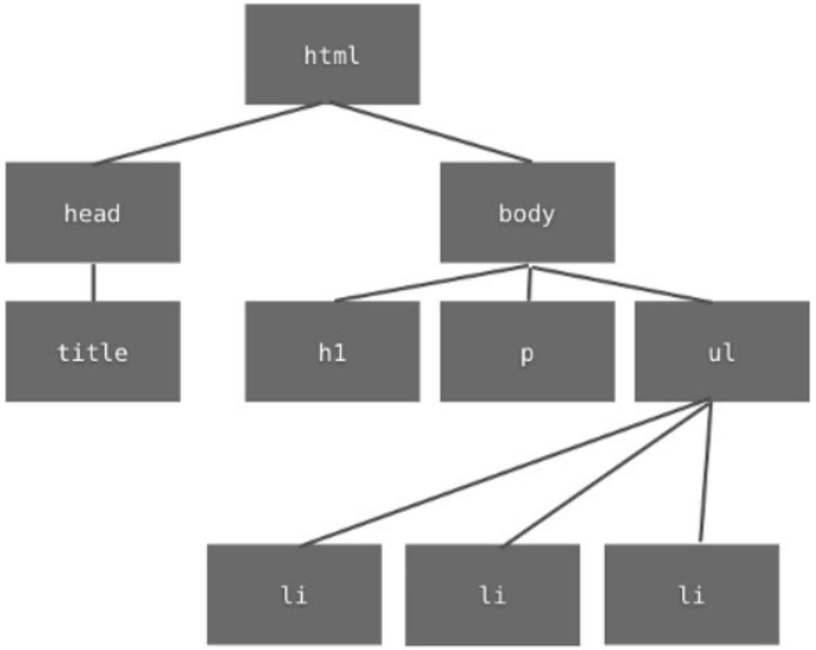
What is DOM?

Model

- Make complexity simple and abstract
- Represent an HTML page in a structured tree



```
<html>
<head>
<title>Page Simple</title>
</head>
<body>
    <h1>Page HTML Simple</h1>
        <p>
        Ceci est une page HTML tr&egrave;s simple.
        </p>

    <p>Elle contient:<p>

    <ul>
        <li>Une balise H1</li>
        <li>Deux paragraphes</li>
        <li>Une liste non ordonn&eacute;e</li>
    </ul>
</body>
</html>
```



What is DOM?

Model

- It allows us to find the desired object, its parent, its children...

- A conventional set of terms that may be used
- Which describe the interaction with elements
- From the webpage

What is DOM?

- It's not a language
- It's an idea, a convention
- Javascript adheres to this idea

What we can do with DOM

- Get text title
- Get the second paragraph
- Get the third menu link and change its CSS attribute: `display:none`
- `<li>` elements from last unordered list
- Find the image that has logo `id` and move it 40 pixels to the `right`

Nodes and Elements

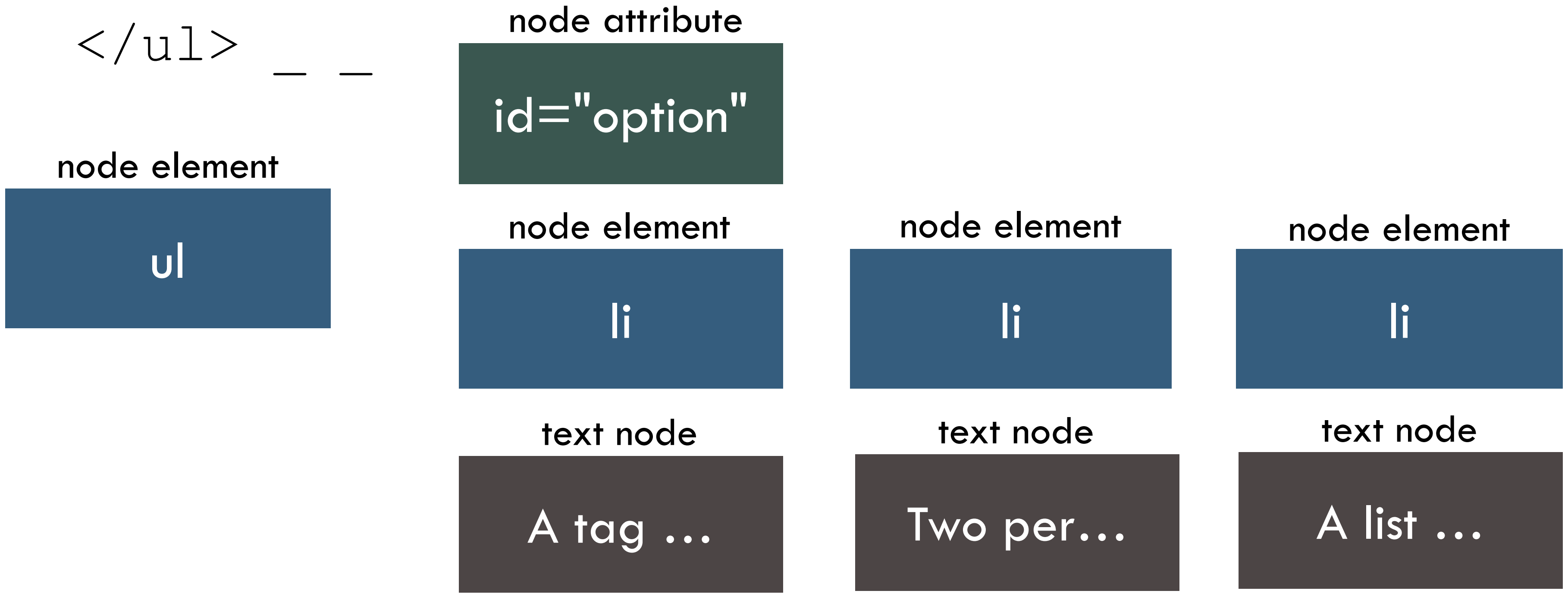
- Nodes represent elements, attributes, text, comments

Types of nodes

- Node.ELEMENT_NODE
- Node.ATTRIBUTE_NODE
- Node.TEXT_NODE

ELEMENTS, ATTRIBUTES AND TEXT

```
< ul id="option">
<li>An H1 tag</li>
<li>Two paragraphs</li>
<li>An unordered list </li>
</ul> _ _
```



How to access an element?

- It is necessary to ensure the uniqueness of the element
- It must contain an `id`

```
document. getElementById (" someID
");
```

- Case sensitive

- It allows to use/modify the properties/methods of the selected element

How to access an element?

- We also have

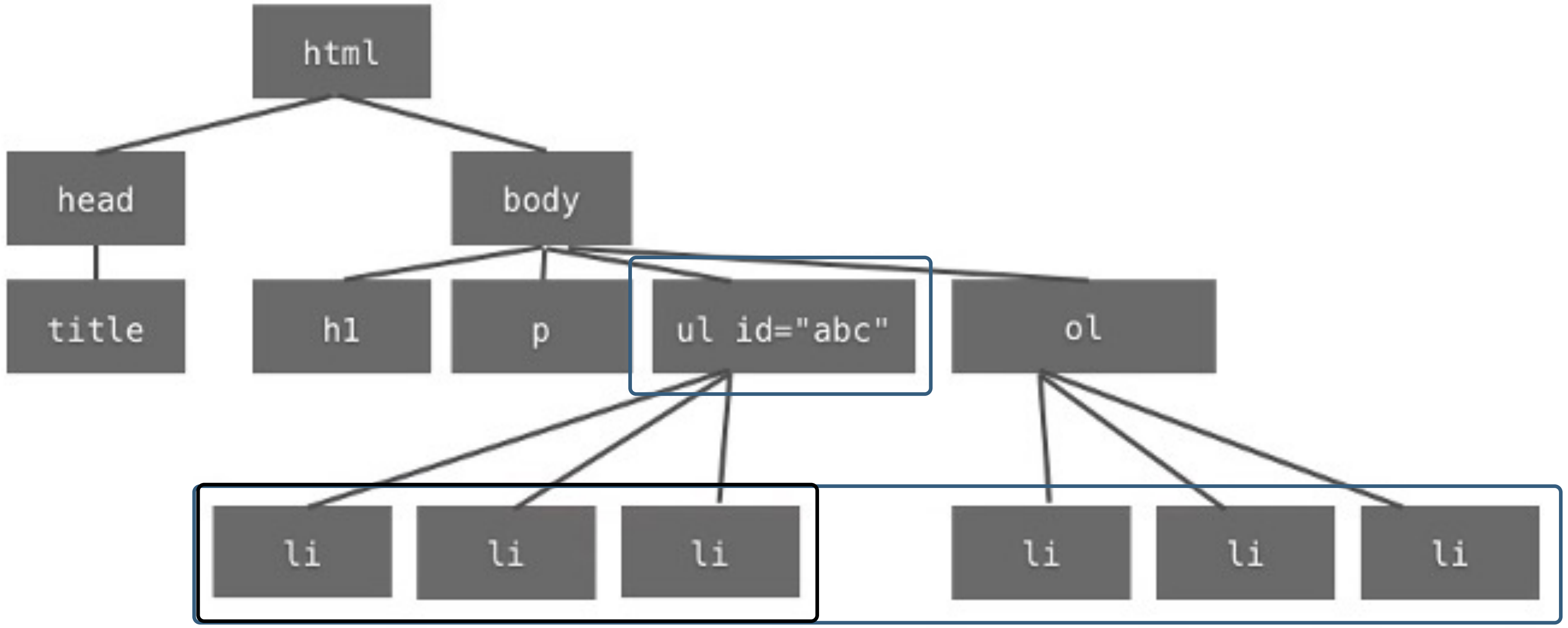
```
document. getElementsByTagName
("a");
```

- gives access to all the link elements
- EXP :

```
var myList = document.getElementsByTagName ("li");
//give an array of li
```

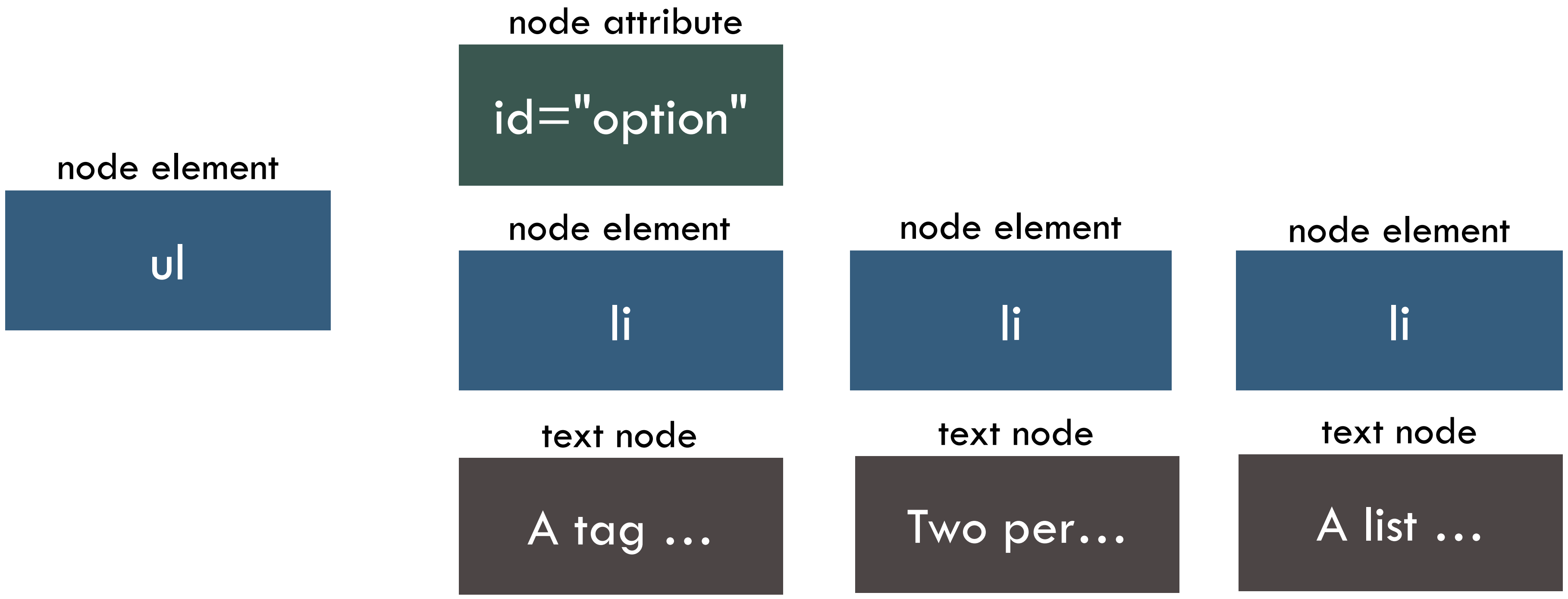
How to access an element?

```
var myLists = document.getElementsByTagName ("li");
var firstList = document.getElementById ("abc");
var listLimit = premList.getElementsByTagName
("li");
```



Change DOM content

- Item need



- What should we change (attributes, text, etc.)

Change DOM content - attributes

- `myElement.getAttribute (" align ");`
- `myElement.setAttribute (" align ", " left ");`

Create DOM content

- Create item

```
var elNew = document.createElement (" element ");
// element = li, p, a …
```

- Add it to the document

```
elementExistant.appendChild ( elNew );
```

- Provide content

```
elNew.innerHTML = "HTML content";
```

Create a text node

```
var myText =
document.createTextNode ("Text");
elNew.appendChild ( myText );
```

Events

What is an event?

- Events are everywhere in the web page,
  - When the page reloads → event
  - When we click → event
  - When typing on the keyboard → event
  - When we move the mouse → event
  - …
- He knows how to hang them and handle them

Event name

- on< eventName >
  - onload
  - onclick
  - onfocus
  - onblur
- We are talking about an event listener (event listener)

Handle event – method 1

- Directly on the HTML of the element

```
< button onclick =" alert ( ' hello ' );"> codeJS
</button>
```

Handle event – method 2

- Element event
- Often used

```
element.event = function () {
} ;
```

```
window.onload , fieldPreci.onblur ,
myElement.onclick
```

Handle event – method 3

- A function linked to an event
  - Event without on
  - Linked function
  - False optional

```
document.addEventListener (' event ', function,false
);
document.removeEventListener (' event ',
function,false );
//For IE8 and earlier
document. attachEvent (' onclick ', function);
```

The forms

Improving forms with JS

- Element values
- Events used to change their values, to move us from one element to another
- The form event
  - In particular → validation (submit)

Access forms and form elements

- Either by the ID (seen in the previous chapter)
- Either by name (name)
- Example:

```
< form name =" frmContact " method …>
<input type=" text " name ="email" … />

document.forms.frmContact // form
document.forms [" frmContact "] // form
document.forms.frmContact.email // input
document.forms.frmContact.elements [i]
document.forms.frmContact.elements [email]
```

Input – text

- Main property
  - `inputText.value` (input or output property)
- Main events
  - `onfocus`
  - `onblur`
  - `onchange`
  - `onkeypress`
  - `onkeydown`
  - `onkeyup`

Radio buttons and checkboxes

- Main property
  - `checkRadio.checked` (returns true or false)
- Main events
  - `onclick`
  - `onchange`

List

- Main properties
  - `mySelect.type` (single or multiple choice)
  - `mySelect.selectedIndex` (single choice)
  - `mySelect.options [i]. selected` (multiple choice)
  - `mySelect.options [ mySelect.selectedIndex ]. text` (for text)
  - Difference between value and text !
- Main events
  - `onchange`

Form events

- More complex (multiple events come into action)
- Main event
  - `onsubmit`
- Disable Validation
  - `event. preventDefault ();`

CSS and JavaScript

Manage Styles

- We use the `style property` of JS
  - `myElement.style`
- Then we use the style attributes
  - `myElement.style.color`
  - `myElement.style.font`
  - `myElement.style.left`
  - `myElement.style.backgroundRepeat`

CSS property naming

- Pay attention to properties

```
#example {
 width : 230px;
 color : #FFF;
font- weight : bold ;
background- color : #193742;
}


myElement.style.width ="230px";
myElement.style.color ="#FFF";
myElement.style.fontWeight = " bold "; //no font-
weight
myElement.style.backgroundColor ="#193742";// not
background- color
```

Manage classes

- We use the `className property` of JS
  - `myElement.className`
- `class` cannot be used
  - This is a reserved keyword ➔

  - myElement.className = " someClass ";
  - myElement.className = "";

Frameworks

Usefulness

- Many developers use/code javascript
- Years of codes have been entered
- A reusable code
- What we want to develop has probably been done
- → library

Where to find them? (Before)

- General
  - mootools
  - YUI – Yahoo User Interface
  - Dojo
  - jQuery
  - Closure Library on GitHub
  - Prototype
- specific
  - LightBox2
  - Aculo script (script.aculo.us)
  - Moofx (moofx.mad4milk.net)
  - CurvyCorners
  - SweetAlert
  - …

Where to find them? (NOW)

- ReactJS
- EmberJS
- AngularJS
- JS View
- Slender JS

- Backbone.js _
- Mithril.js
- Polymer.js
- Node.js
- Meteor.js

Links to external files

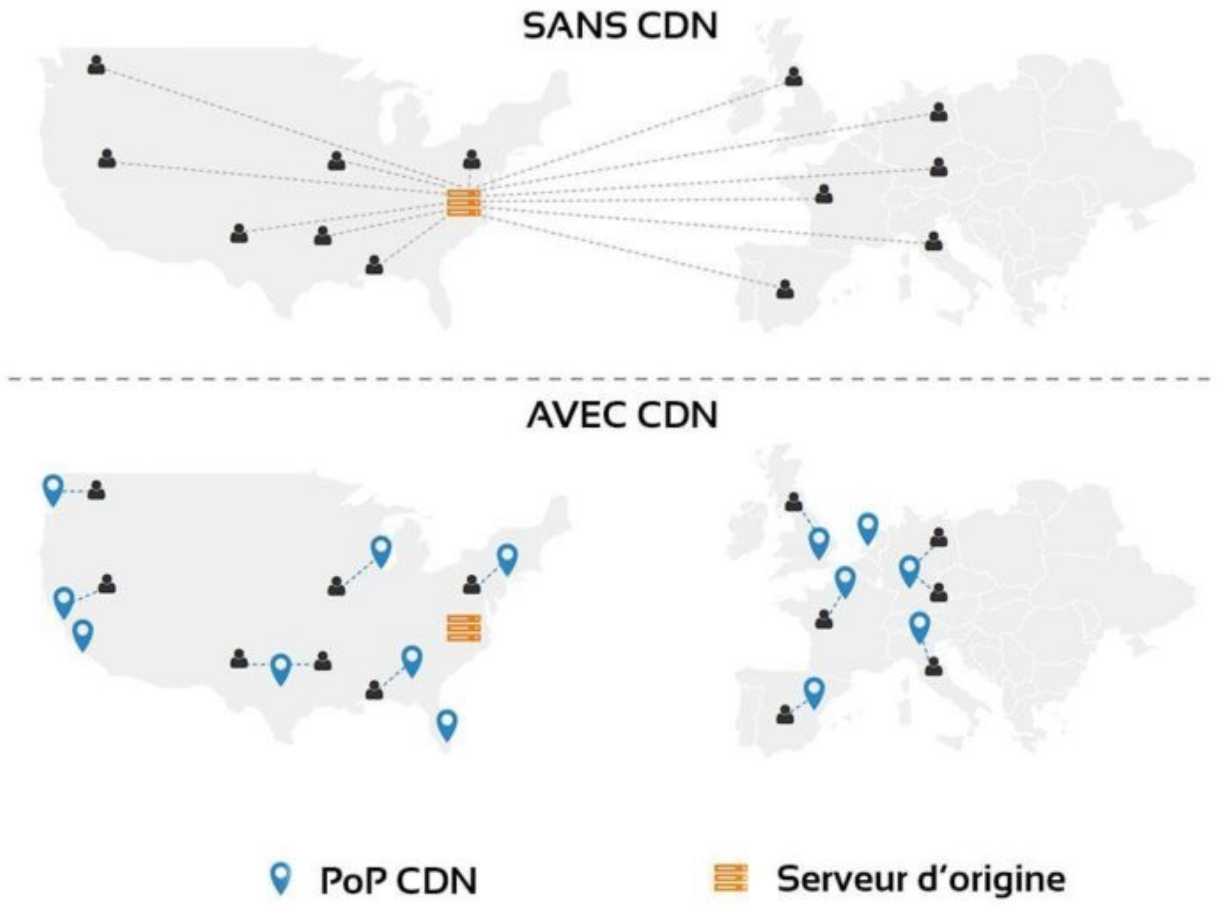
- Watch out for slow
  - Before we . js not getting downloaded,
  - The previous one will have to be fully downloaded
  - Consider CDNs (Content Delivery Network)
    - Improves speed, downloading, redundancy…

```
…
<script src =" script.js "></script>
<script src =" another.js "></script>
<script src =" athird.js "></script>
```

A few statistics

- **79%** of online shoppers say they won't return to a website if they've had problems with the site's loading speed.
- **47%** of people expect your site to load in less than 2 seconds.
- **40%** will completely abandon it if it takes more than 3 seconds to load.
- **85%** of internet users expect a mobile site to load as fast or faster than on their desktop.

CDN



Which ones?

- MaxCDN
- CloudFlare
- Incapsula
- Amazon CloudFront
- Google App Engine
- Microsoft Azure CDN

How to choose them?

- Its performance (query speed)
- Network availability (RUM uptime)
- The number of servers distributed in the world
- Its features
- His reputation
- A good price-performance ratio

Introduction to jQuery

- The most popular JS library
- Makes javascript easier
  - To navigate and manipulate DOMs
  - To manipulate events
  - To work with animations

How to use it?

- Download source from www.jquery.com

JavaScript vs. jQuery

- `document.getElementById (" myElem "). className = " someClass ";`
- `$("# myElem "). addClass (" someClass ");`
- **JS: With `getElementById` , we are dependent on an ID**
- **jQuery : more flexible (we will use CSS selectors)**
  - `$(". otherClass ")`
  - `$("p")`
  - `$(" p.description ")`
- Other properties
  - `:first`
  - `:last`
  - `: contains ()`

jQuery

```
$(" whatToFind "). actionTODo ;
```

- A.k.a

```
$(" whatToFind "). actionToDo ( param );
$(" myElem "). addClass (" someClass ");
$(" myElem "). addClass (" someClass ");
```

Event management

```
$(" whatToFind "). event ( function () {
// Instructions
});

// ready , click, hover , dblclick ,
```

Event management

```
Example 1
var hiddenBox = $("#banner-message");
$( "# button -container button ").on( "click", function () {
 hiddenBox. show ();
});
Example 2
$("# other ").click( function () {
$("# target ").focus();
});
Example 3
var e = jQuery.Event ("click");
// trigger an artificial click event
jQuery("# other ").trigger(e);
```

The ready event

- Triggers only when the page is loaded
- Example

```
jQuery (document). ready ( function
($) {
});


$(document). ready …


$( function () { …
```

Animations - the effects

- show() sudden appearance
- hide () abrupt disappearance
- fadeIn () appearance in split
- fadeOut () split fadeout
- fadeToggle () appearance/disappearance on each click
- slideToggle () roll up/unroll on each click

- $(" element (s)").effect …

Advanced Mode

document.write

- You can use document.write (" Text /HTML");
- Inconvenience:
  - document.write overwrites the initial content of the HTML page

Regular expressions

- Character sequences that match a search in a character string
- To create:

```
var expReg = /hello/
var expReg = new RegExp ("hello"); //The same
var myString = "is there a hello in the string?";
if ( expReg. test ( myString )) {
 alert ("yes");
}
```

Creation of models

```
var expReg = /^hello/; //^ at the beginning of the string
/hello$/; //$ at the end of the string
/ good+day /; //+ "n" at least once
//hello, hello , hello
/Good morning/; //* "n" zero or more
// hello , hello, hello
/ good morning /; //? zero or once
// hello , hello
/ hello|bye /; // Or
/good...r/; //. any car.
/\ wonjour /; //\w alphanumeric or _
/gr[ aèio ]s/; //[…] one of the characters
```

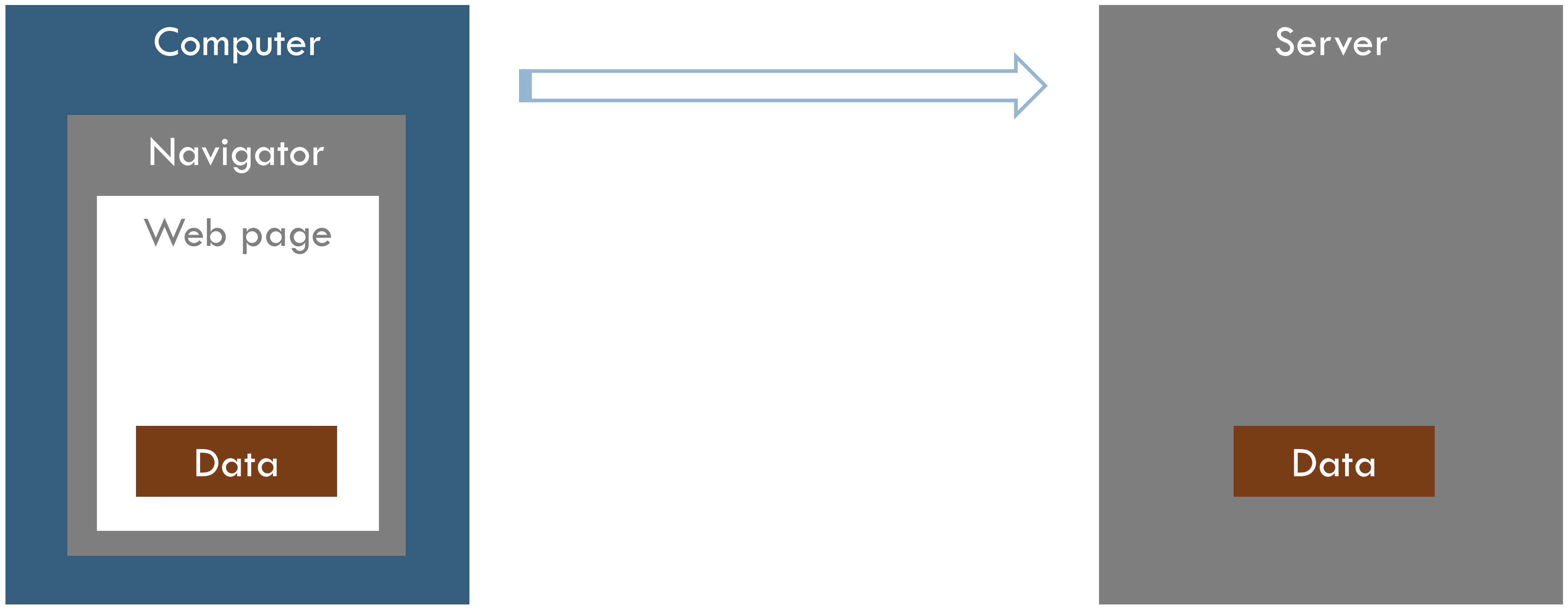
More complex models

```
/^[0-9]{5}$/; //Postal code
/^[a- zA -Z]+[a-zA-Z0-9._-]*\[at\][a-zA-Z0-9.-
]+\.[a-zA-Z ] { $2.4}/;
// E-mail
```

AJAX

- AJAX: Asynchronous JavaScript And XML
- AJAX == JavaScript

What is AJAX?



Create the query

```
var myReq ;
if( window. XMLHttpRequest ) {
 myReq = new XMLHttpRequest ();
} else if ( window.ActiveXObject ) {
 myReq = new ActiveObject (" Microsoft.XMLHTTP ");
}
```

Prepare for the answer

```
myReq.onreadystatechange = function () {
if (( myReq.readyState == 4) && ( myReq.status !="404")) {
var p = document. createElement ("p");
var t = document.createTextNode ( myReq.responseText );
   p.appendChild ( t ); document.getElementById (" mainContent
"). appendChild (p);
}
};

//After configure and send
myReq.open (" GET ","http ://link_to_page.php ", true );
myReq.send ( null );
```

The readyState property

Value _	State _	D escription
0	UNSENT	The customer has been created. open() has not been called yet.
1	OPENED	open() was called.
2	HEADERS_RECEIVED	send () has been called, and headers and status are available.
3	LOADING	Download; responseText contains partial data.
4	DONE	The operation is complete.

The status property

Value _	Status _	D escription
1xx: Information		
100	keep on going	The server has received the request headers, and the client should proceed to send the request body.
101	switching Protocols	The requester has requested the server to change protocol.
103	Early Hints	Used with the Link header to allow the browser to begin preloading resources while the server prepares a response.

The status property

Value _	Status _	D escription
2xx: Successful		
200	OK	The request is OK
201	C reated	The request has been satisfied and a new resource is created.
202	A ccepted	The request has been accepted for processing, but processing has not been completed.
203	Non- Authoritative Information	The request was processed successfully, but it returns information that may have come from another source.
204	No Content	The request was processed successfully, but does not return any content.
205	Reset Content	The request was processed successfully, but returns no content, and requires the requester to reset the document view.
206	Partial Content	The server only delivers part of the resource due to a

The status property

Value _	Status _	D escription
3xx: Redirect		
300	Multiple Choices	A list of links. The user can select a link and go to that location. Five addresses maximum
301	Moved Permanently	The requested page has been moved to a new URL
302	Found	The requested page has been temporarily moved to a new URL.
303	See other	The requested page is under another URL.
304	Not Modified	Indicates that the requested page has not been modified since the last request.
307	Temporary redirect	The requested page has been temporarily moved to a new URL.
308	Permanent Redirect	The requested page has been permanently moved to a new URL.

The status property

Value _	Status _	D escription
4xx: Customer Error		
400	bad request	The request cannot be fulfilled due to bad syntax.
403	F orbidden	The request was legal, but the server refuses to respond.
404	Not found	The requested page could not be found but may be available again in the future.
413	Request Too Wide	The server does not accept the request because the request entity is too large.

The status property

Value _	Status _	D escription
5xx: Server Error		
500	Internal Server Error	A generic error message, given when no more specific message is suitable.
502	Bad gateway	The server was acting as a gateway or proxy and received an invalid response from the upstream server.
503	Service Unavailable	The server is currently unavailable (overloaded or down).
504	Gateway Timeout	The server was acting as a gateway or proxy and was not getting a timely response from the upstream server.

Ajax with jQuery

```
<? php
// Retrieve parameters
$string = '';
if ( isset ($_GET[' string ']) ){
$ string = $_GET[' string '];
}
// Processing
$return = array (
' string ' => strtoupper ($ string ),
' date ' => date ('d/m/Y H:i:s '),
' phpversion '=> phpversion ()
);
// Send the return (we return the array $return encoded in JSON)
header('Content-type: application/ json ');
echo json_encode ($ return );
?>
```

Ajax with jQuery

```
< form id=" form ">
<input name=" string " type="text" id=" string "
value="Hello" />
<input type="submit" value=" Submit " id="handle" />
</ form >

<div id="return">
<i>empty</i>
</div>
```

Ajax with jQuery

```
$('#form').submit( function (e){
// Disable default browser behavior
// (which consists of calling the action page of the form)
        e.preventDefault ();
$. getJSON ( // We send the AJAX request
' file.php ', { string : $('# string '). val ()},
            function (data){
$('#return').hide();
$('#return').html('')
.append('<b>Uppercase parameter</b>: '+ data.string +'< br />')
. append ('<b>Date</b>: '+ data.date +'< br />')
. append ('<b>PHP Version</b>: '+ data.phpversion +'< br />');
$('#return'). fadeIn ();
}
);
});
});
```

Tips and Best Practices

Writing Style Guide

- The rules we must follow to write JS
  - How to name variables/functions
  - Where should we place the functions
  - How to make code readable
- Because javascript is readable by everyone

Naming conventions

- Variables / functions
  - Must start with letters, numbers, $ or _
  - Avoid " `var _XYZ$$_lk5sh33` "," `var a, b…` "
  - Advice:
    - var score; // full name
    - var bestScore ; // camelCase -style
    - function calculateDistance () { …

Naming conventions

- Objects
  - Uppercase first letter
    - Math, Date…
- Convention adopted (camelCase)
  - Yahoo, Google, jQuery , DOM methods

Braces

```
if (x) {
// …
} else {
// …
}
```

- To avoid:

```
if (x)
{
// …
}
```

The blocks

```
if (x) {
 alert (Message);
}
```

- To avoid

```
if (x)
 alert (Message);
```

Function calls

```
function aFunction () {
 otherFunction ();
}
function otherFunction () {
//…
}
// preferably
function otherFunction () {
//…
}
function aFunction () {
 otherFunction ();
}
```

Reminder

- Use the camelCase style
- Open braces in the same line
- Always use blocks, even with a single line
- Declare functions before calling them
- Always use semicolons
- Always use var to declare variables
- ➔ javascript style guidelines()

Minification javascript

- Reduce code size saving loading time
- Rename variables and functions to shorter variables " `var a, b` " instead of " `var lastname, firstname` "

Minification tools

- JSMin
- YUI Compressor
- JS Compress
- Google Closure Compiler