Cours de programmation web

PAUL-ANTOINE BISGAMBIGLIA

2021-2022 V1 2015-2016

HTTP://PAUL-ANTOINE-BISGAMBIGLIA.UNIV-CORSE.FR/

@PABISGAMBIGLIA

Licence

- Cours en licence libre
- Certaines images ont été récupérées sur google image avec le filtre « réutilisation utilisée sans but commercial »
- Certains exemples de code sont issus de sites web référencés en fin de chapitre via leurs urls

Objectifs

- Approfondir les notions de programmation web
- Apprendre à créer un site internet riche

- Technologies utilisées
  - HTML5
  - CSS3
  - JavaScript
  - PHP
  - MySQL
  - Des frameworks

Prérequis

Master 1

- Cours de L3
  - Algorithmique et programmation
  - Notion de réseau
    - Communication client serveur
    - Protocole HTTP
  - Techo web : HTML/CSS/JS/PHP

Licence 3

- Notion d'algorithmique et de programmation
- Notion de réseau
  - Communication client serveur
  - Protocole HTTP

Programme

- Introduction et révisions
  - Le web, et ses techno
- HTML 5
  - Langage de balise <*body*> </*body*>
- CSS 3 avec bootstrap
  - Mise en forme
  - Ergonomie du web
- JavaScript avec jQuery
  - Dynamisme coté client
  - Angular.js
  - Node.js
- PHP
  - Dynamisme coté serveur
  - Les frameworks (Zend, Symfony, cakephp)
- Le +
  - les Design Patterns pour le web
  - Sécuriser son code
  - Django le framework web en python
  - Framework html 5 pour mobile (IONIC, Mobile Angular UI, Intel XDK, Appcelerator Titanium, Sencha Touch)
  - Apache Cordova, Meteor

JSON

JSON

- JSON (JavaScript Object Notation) est un format de stockage et d'échange de données.
- Il est souvent utilisé lorsque des données sont envoyées entre un serveur et une page Web.
  - Format d'échange de données
  - Indépendant du langage
  - Auto décrit

```
{
"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]
}
```

JSON

JSON

Le format JSON est syntaxiquement identique au code JavaScript pour créer des objets. Grâce à cette similitude, un programme JavaScript peut facilement convertir les données JSON en objets JavaScript natifs.

```
var text = '{ "employees" : [' +
'{ "firstName":"John" , "lastName":"Doe" },' +
'{ "firstName":"Anna" , "lastName":"Smith" },' +
'{ "firstName":"Peter" , "lastName":"Jones" }
]}';

var obj = JSON.parse(text);
```

JSON

JSON

- Les **données** sont composées d'une paire clé:valeur ou nom: valeur ou identifiant:valeur
- Les **objets** par des accolades
- Les **tableaux** par des crochets

Donnée

"firstName":"John"

Objet

{"firstName":"John", "lastName":"Doe"}

Tableau

"employees":[…]

JSON

Exemple

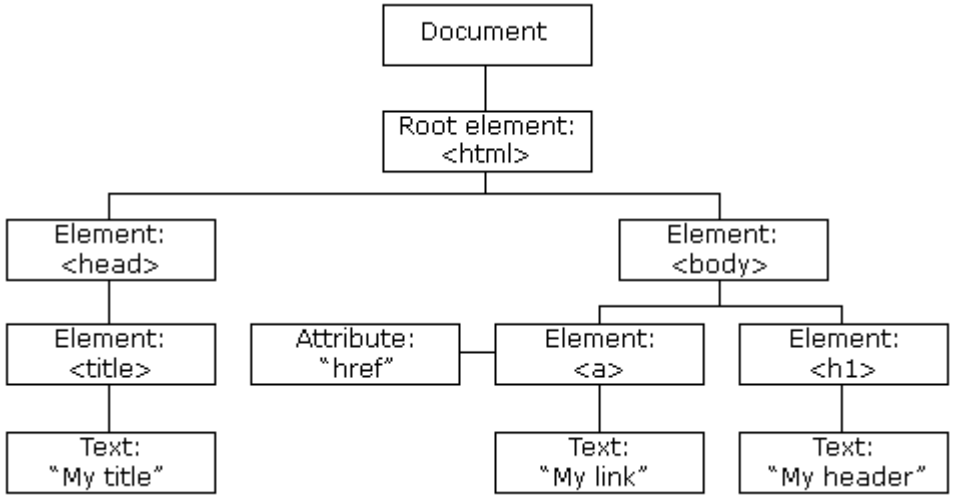
Create Object from JSON String

Anna Smith

```
<!DOCTYPE html>
<html>
<body>
<h2>Create Object from JSON String</h2>
<p id="demo"></p>
<script>
var text = '{"employees":[' +
'{"firstName":"John","lastName":"Doe" },' +
'{"firstName":"Anna","lastName":"Smith" },' +
'{"firstName":"Peter","lastName":"Jones" }]}';
obj = JSON.parse(text);
document.getElementById("demo").innerHTML =
obj.employees[1].firstName + " " +
obj.employees[1].lastName;
</script>
</body>
</html>
```

JS & DOM

Document Object Model



"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

Le JS peut changer / déplacer / ajouter :

- Des éléments HTML à/de la page
- Des attributs HTML
- Des styles CSS

JS & DOM

Accéder aux éléments HTML

Method	Description
document.getElementById(id)	Find an element by element id
document.getElementsByTagName(name)	Find elements by tag name
document.getElementsByClassName(name)	Find elements by class name

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Page</h1>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
</body>
</html>
```

JS & DOM

Modifier des éléments HTML

Method	**Description**
element.innerHTML = *new html content*	Change the inner HTML of an element
element.attribute = new value	Change the attribute value of an HTML element
element.setAttribute*(attribute , value)*	Change the attribute value of an HTML element
element.style.*property = new style*	Change the style of an HTML element

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Page</h1>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
</body>
</html>
```

JS & DOM

Ajout et suppression d'éléments

Method	Description
document.createElement(element)	Create an HTML element
document.removeChild(element)	Remove an HTML element
document.appendChild(element)	Add an HTML element
document.replaceChild(element)	Replace an HTML element
document.write(text)	Write into the HTML output stream

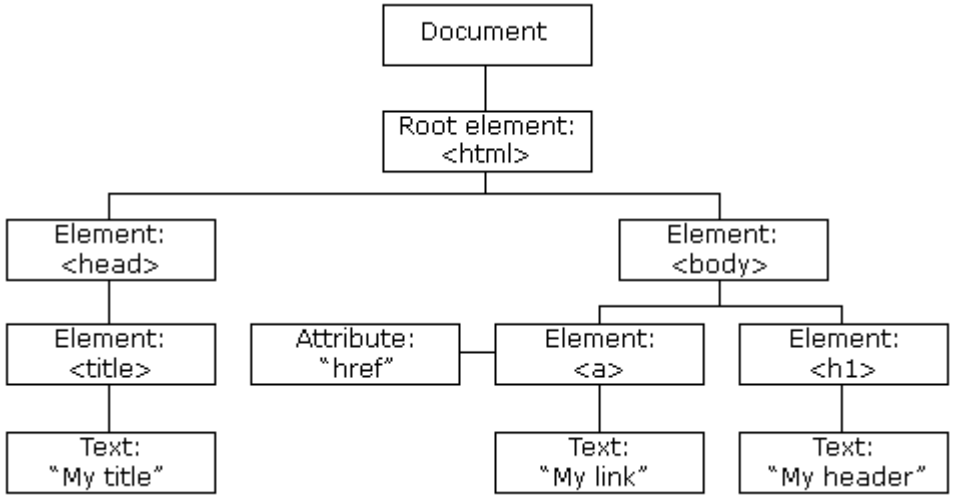
JS & DOM

Ajout et suppression d'évenement

Method	Description
document.getElementById(*id*).onclick = function(){*code*}	Adding event handler code to an onclick event

JS & DOM

Document Object Model



- document.anchors Returns all <a> elements that have a name attribute
- document.body Returns the <body> element
- document.cookie Returns the document's cookie
- document.forms Returns all <form> elements
- document.URL Returns the complete URL of the document

JS & DOM

Document Object Model

```
<!DOCTYPE html> <html> <body>

<p>Hello World!</p>

<p class="intro">The DOM is very useful.</p>

<p class="intro">This example demonstrates the <b>getElementsByClassName</b> method.</p>

<p id="demo"></p>
```

```
<script>

var x = document.getElementsByClassName("intro");

document.getElementById("demo").innerHTML =

'The first paragraph (index 0) with class="intro": ' + x[0].innerHTML;

</script>

</body>

</html>
```

JS & DOM

Document Object Model

Cet exemple renvoie la liste des élément <p> avec comme class intro

```
<!DOCTYPE html>
<html>
<body>
<p>Hello World!</p>
<p class="intro">The DOM is very useful.</p>
<p class="intro">This example demonstrates the <b>querySelectorAll</b> method.</p>
<p id="demo"></p>
<script>
var x = document.querySelectorAll("p.intro");
document.getElementById("demo").innerHTML =
'The first paragraph (index 0) with class="intro": ' + x[0].innerHTML;
</script>
</body>
</html>
```

JS & DOM

Document Object Model

Changer un attribut

```
<!DOCTYPE html>
<html>
<body>
<img id="image" src="smiley.gif" width="160" height="120">
<script>
document.getElementById("image").src = "landscape.jpg";
</script>
<p>The original image was smiley.gif, but the script changed it to landscape.jpg</p>
</body>
</html>
```

JS & DOM

Document Object Model

Changer un style css

```
<!DOCTYPE html>
<html>
<body>
<p id="p1">Hello World!</p>
<p id="p2">Hello World!</p>
<script>
document.getElementById("p2").style.color = "blue";
document.getElementById("p2").style.fontFamily = "Arial";
document.getElementById("p2").style.fontSize = "larger";
</script>
<p>The paragraph above was changed by a script.</p>
</body>
</html>
```

JS & BOM

Modèle objet du navigateur

- Il n'y a pas de modèle standard
- Le BOM permet d'accéder :
  - L'écran
  - L'historique de navigation
  - La localisation
  - Etc.

Some methods:

- window.open() - open a new window
- window.close() - close the current window
- window.moveTo() -move the current window
- window.resizeTo() -resize the current window

JS & BOM

window.screen

- L'objet **window.screen** permet de récupérer des informations sur l'écran :
  - screen.width
  - screen.height
  - screen.availWidth
  - screen.availHeight
  - screen.colorDepth
  - screen.pixelDepth

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
"Screen width is " + screen.width +" x "+ screen.height;
</script>
</body>
</html>
```

JS & BOM

window.location

- L'objet **window.location** permet de récupérer des informations sur la localisation
  - window.location.href returns the href (URL) of the current page
  - window.location.hostname returns the domain name of the web host
  - window.location.pathname returns the path and filename of the current page
  - window.location.protocol returns the web protocol used (http:// or https://)
  - window.location.assign loads a new document

```
<!DOCTYPE html>
<html>
<head>
<script>
document.getElementById("demo").innerHTML = "Page hostname is " + window.location.hostname;

function newDoc() {
    window.location.assign("http://www.univ-corse.fr")
}
</script>
</head>
<body>
<p id="demo"></p>
</br>
<input type="button" value="Load new document" onclick="newDoc()">
</body>
</html>
```

JS & BOM

window.history

- L'objet **window.history** permet d'accéder à l'historique de navigation, pour des raisons de respect de la vie privée, il n'y a que deux méthodes :
  - history.back()
  - history.forward()

```
 <html>
<head>
<script>
function goBack() {
    window.history.back()
}
</script>
</head>
<body>

<input type="button" value="Back"
onclick="goBack()">
</body>
</html>
```

JS & BOM

window.navigator

- L'objet **window.navigator** permet d'accéder aux propriétés du navigateur :
  - navigator.appName
  - navigator.appCodeName
  - navigator.platform
  - navigator.cookieEnabled

```
<!DOCTYPE html>
<html>
<body>
<p>What is the name(s) of your browser?</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML =
    "Name is " + navigator.appName +
    "<br>Code name is " + navigator.appCodeName +
    "<br>Browser engine is " + navigator.product;
}
</script>
</body>
</html>
```

JS & BOM

Alert

- Les ***alert box***
  - ***window.alert(***"text"***)***
  - ***window.confirm(***"text"***)***
  - ***window.prompt(***"text", "valeur par défaut"***)***

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to display a confirm box.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x;
    if (confirm("Press a button!") == true) {
        x = "You pressed OK!";
    } else {
        x = "You pressed Cancel!";
    }
    document.getElementById("demo").innerHTML = x;
}
</script>
</body>
</html>
```

JS & BOM

Alert

- Les ***alert box***
  - ***window.alert(***"text"***)***
  - ***window.confirm(***"text"***)***
  - ***window.prompt(***"text", "valeur par défaut"***)***

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to demonstrate the prompt box.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var person = prompt("Please enter your name", "prenom nom");
    if (person != null) {
        document.getElementById("demo").innerHTML =
        "Hello " + person + "! How are you today?";
    }
}
</script>
</body>
</html>
```

JS & BOM

Evènement temporisé

- setTimeout(function, milliseconds)
  Exécute *function* après avoir attendu un certain nombre de millisecondes spécifié *milliseconds*.
- setInterval(function, milliseconds)
  Même chose mais répète l'exécution de manière continue
- window.clearTimeout(timeoutVariable)
  Stop l'exécution de la fonction

```
<!DOCTYPE html>
<html>
<body>
<p>A script on this page starts this clock:</p>
<p id="demo"></p>
<script>
var myVar = setInterval(myTimer, 1000);
function myTimer() {
    var d = new Date();
    document.getElementById("demo").innerHTML
= d.toLocaleTimeString();
}
</script>
</body>
</html>
```

JS & BOM

Cookies

- Les cookies sont des petits fichiers de données stockés sur le client. Ils permettent d'enregistrer des informations comme le nom de l'utilisateur.

- En JS on peut les créer, lire et supprimer
- On peut spécifier une date d'expiration, par défaut ils sont détruits à la fermeture du navigateur.

JS & BOM

Cookies

- Créer un cookie :

document.cookie="username=John Doe; expires=Thu, 18 Dec 2016 12:00:00 UTC; path=/";

- Lire un cookie :

var x = document.cookie;

⇒ cookie1=value; cookie2=value; cookie3=value;

- Supprimer un cookie

document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 UTC";

JS & BOM

Cookies

```
function setCookie(cname, cvalue, exdays) {
    var d = new Date();
    d.setTime(d.getTime() + (exdays*24*60*60*1000));
    var expires = "expires="+d.toUTCString();
    document.cookie = cname + "=" + cvalue + "; " + expires;
}
```

```
function getCookie(cname) {
    var name = cname + "=";
    var ca = document.cookie.split(';');
    for(var i=0; i<ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0)==' ') c = c.substring(1);
        if (c.indexOf(name) == 0) return c.substring(name.length,c.length);
    }
    return "";
}
```

```
<!DOCTYPE html>
<html>
<head>
<script>
function setCookie(cname,cvalue,exdays) {
    var d = new Date();
    d.setTime(d.getTime() + (exdays*24*60*60*1000));
    var expires = "expires=" + d.toGMTString();
    document.cookie = cname+"="+cvalue+"; "+expires;
}
function getCookie(cname) {
    var name = cname + "=";
    var ca = document.cookie.split(';');
    for(var i=0; i<ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0)==' ') c = c.substring(1);
        if (c.indexOf(name) == 0) {
            return c.substring(name.length, c.length);
        }
    }
    return "";
}
function checkCookie() {
    var user=getCookie("username");
    if (user != "") {
        alert("Welcome again " + user);
    } else {
        user = prompt("Please enter your name:","");
        if (user != "" && user != null) {
            setCookie("username", user, 30);
        }
    }
}
</script>
</head>
<body onload="checkCookie()">
</body>
</html>
```

http://www.w3schools.com/js/js_cookies.asp

A vous > form dynamique

1. Faire un fichier json qui liste les types d'input (text, mail, number, date, submit)
2. Créer un fichier html composé de deux zones horizontales (A/B)
3. A partir du fichier json (et avec jQuery) créer en ensemble de checkbox qui listent les éléments du fichier (Zone A)
4. À la sélection d'une box, le composant doit-être ajouté dans la zone B.

- Aide :
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Utiliser_les_objets
- http://defeo.lu/aws-2014/classes/class2
- https://blog.groupe-sii.com/introduction-aux-web-components/
- https://www.w3schools.com/jquery/jquery_ajax_get_post.asp

A vous > form dynamique

```
{
"items":[
    {"type":"text", "id":"text1",
"name":"text1"},
    {"type":"submit", "id":"sub1",
"name":"text1"},
    {"type":"mail", "id":"mail1",
"name":"text1"},
    {"type":"number", "id":"num1",
"name":"text1"}]
}
```

- Aide :
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Utiliser_les_objets
- http://defeo.lu/aws-2014/classes/class2
- https://blog.groupe-sii.com/introduction-aux-web-components/
- https://www.w3schools.com/jquery/jquery_ajax_get_post.asp

Locale storage

- https://www.w3schools.com/jsref/prop_win_localstorage.asp
- https://developer.mozilla.org/fr/docs/Web/API/Window/localStorage

- monStockage = localStorage;
- localStorage.setItem('monChat', 'Tom');
- var cat = localStorage.getItem('myCat');
- localStorage.removeItem('myCat');
- localStorage.clear();