HLIN102
Site web dynamique

Introduction

- HTML : pages destinées à être publiées sur Internet
  - Texte à afficher + instructions de mise en page
  - Pas d'instructions de calcul ou de traitements conditionnels

  => HTML + CSS = page statique

- Des sites de plus en plus riches en informations
  - Nécessité croissante d'améliorer le contenu des sites
  - Mises à jour manuelles trop complexes
    - Exemple : modifier l'entête sur plusieurs pages !
  - Besoin de réponses spécifiques liées à un BD par exemple

- Passage de sites statiques à des sites dynamiques

Web dynamique – coté serveur

- L'interprétation est réalisée par le serveur :
  - Indépendant de la machine et du navigateur
  - "Compatible" avec tous les navigateurs
    - Les échanges ne concernent que du HTML (ou Json ou autre)
  - Les sources sont sur le serveur donc invisibles

- Besoin d'échanges entre le navigateur et le serveur
  - Rechargement de la page à chaque modification
  - Ou Ajax…

Web dynamique – coté client

- Traité par le navigateur :
  - Résultats variables en fonction du navigateur
    - Nécessité de tests importants
  - Indépendant du serveur
    - Pas de rechargement de la page, tout est fait en local

- Confiance :
  - Sources du programme disponibles
  - Données envoyées au serveur pas fiables
  - Base de données stockée chez le client ?

Web dynamique – client ou serveur

- Il faut les deux !

- Script côté client (Javascript) :
  - Calculs et traitement simples
  - Mises à jour de la page web sans rechargement (ajax)

- Script côté serveur (Php ou autre) :
  - Calculs, traitements plus conséquents
  - Requêtes vers une base de données
  - Opérations sécurisées

AJAX

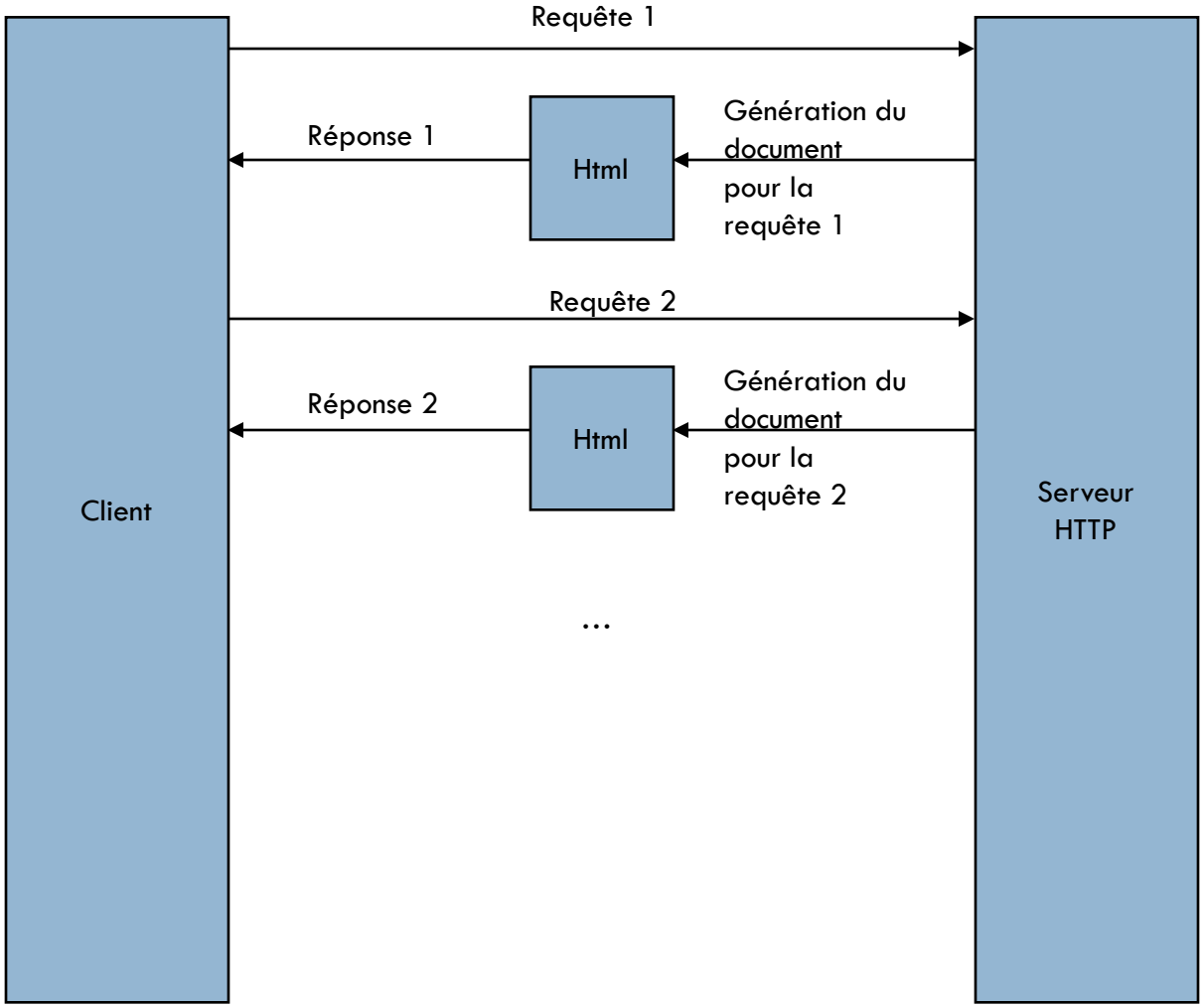
LI288 – web et développement web

Application traditionnelle

- Application WEB traditionnelle :
  - Le client envoie une requete HTTP
  - Le serveur renvoie une page

- Consommation inutile de la bande passante :
  - Une grande partie du code HTML est commun aux différentes pages de l'application.
- Le chargement d'une nouvelle page à chaque requête n'est pas ergonomique

Application traditionnelle



AJAX

- Qu'est-ce qu'AJAX ?
  - Asynchronous Javascript and XML

- Pourquoi AJAX:
  - Javascript est très utilisé au niveau du client :
    - validation de formulaire, modifications de la page, …
  - Tout ne peut pas être confié au client :
    - Manque de sécurité/confiance
    - Limitations

AJAX

- Principe de base :
  - Le client et le serveur dialoguent.

- Autant faire en sorte que les messages soient le plus petits possibles.
  - Le client n'a pas besoin de toute la base de données, juste de suffisamment de données pour le client.
- Le serveur et le client ont chacun un travail
  - L'application ne doit donc pas être prise en charge entièrement d'un coté ou de l'autre.

AJAX = un acronyme de plus

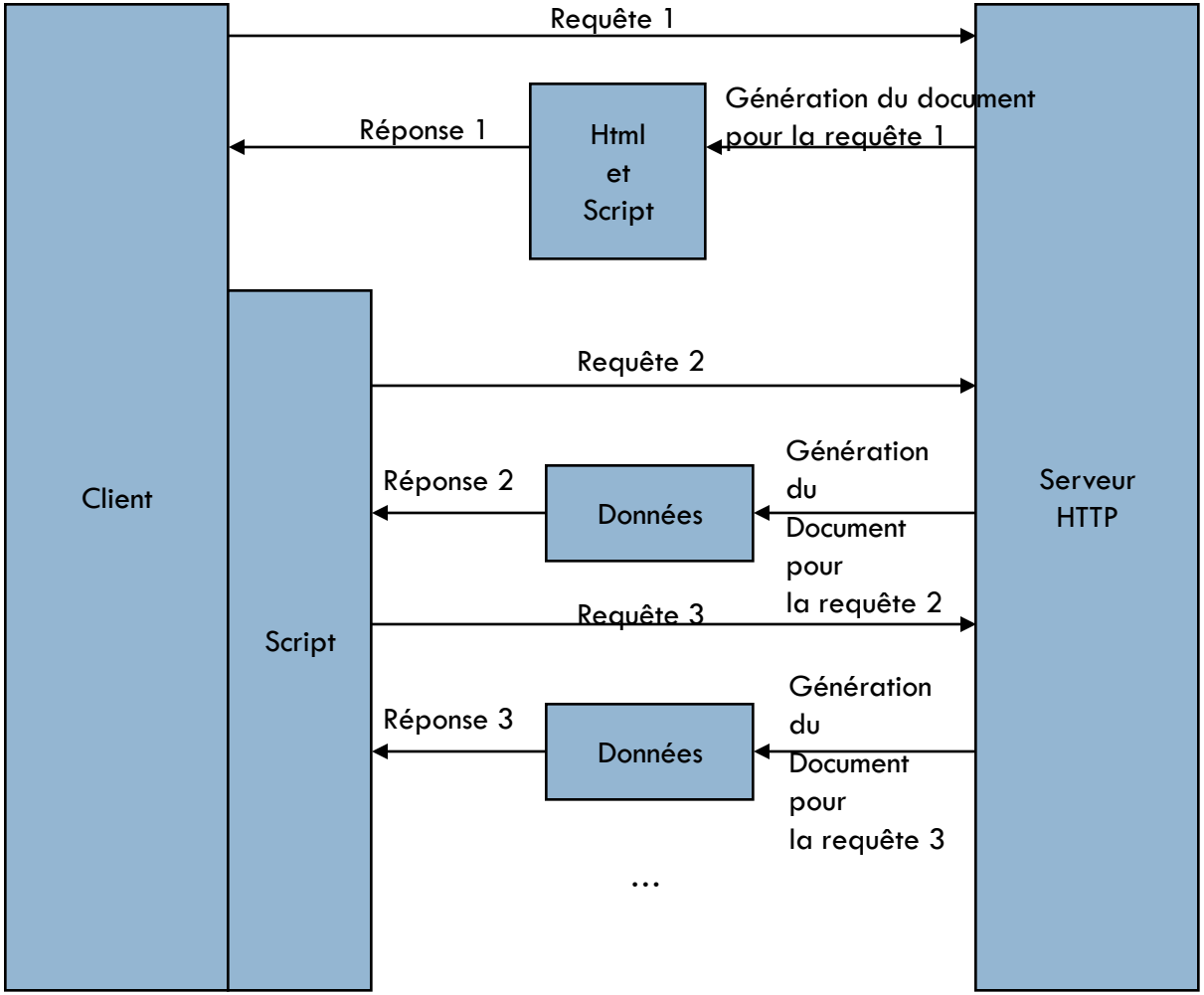
- C'est juste du Javascript classique.

- Principe de base :
  - L'application Javascript émet des requêtes vers le serveur avec un protocole donné.
  - Le serveur répond avec les informations demandées.
  - Tout se passe sans rechargement de la page.
    - Mode synchrone ou asynchrone pour le client.
  - Javascript traite les données reçues et modifie la page en conséquence.

Qui utilise Ajax

- Clients de messagerie : Gmail, Yahoo Mail, HotMail
- Google Maps
- FlickR, Picasa
- Deezer
- Youtube, Dailymotion
- Myspace, Facebook

AJAX


Client
Script
Serveur HTTP
Requête 1
Réponse 1
Html et Script
Génération du document pour la requête 1
Requête 2
Réponse 2
Données
Génération du Document pour la requête 2
Requête 3
Réponse 3
Données
Génération du Document pour la requête 3
...

Attention !

- Les requêtes AJAX asynchrones passent par Internet
  - Aucune garantie que les paquets arrivent dans l'ordre.
  - Aucune garantie qu'une requête soit terminée avant qu'une autre ne soit lancée :
    - Les délais peuvent varier énormément à cause de la charge du serveur et du réseau.

Inconvénients

- JavaScript doit être activé.
- Les données chargées de façon dynamique ne font pas partie de la page. Prise en compte par les moteurs de recherche pas claire.
- Asynchrone => affichage avec délai, peut poser problème à l'utilisateur.
- Le bouton « Page précédente » ne marche pas en général.

Conclusions sur Ajax

- Combinaison des langages standards du WEB (Javascript, DOM HTML, XML / JSON)
- Grâce à l'objet XMLHttpRequest / à JQuery
- WEB dynamique « coté client »
- Utilisé par tous les sites « WEB 2.0 »

JAVASCRIPT

LI288 – web et développement web

Introduction

- Le Javascript est un langage "de script" simplifié "orienté objet" :
  - Initialement élaboré par Netscape en association avec Sun Microsystem.
  - Standardisé par un comité spécialisé, l'ECMA (European Computer Manufactures Association).

- Javascript permet :
  - De rendre dynamique un site internet développé en HTML :
    - Validation de formulaires, calculs, messages,
    - Modification de la page web,
    - Communication avec un serveur directement (AJAX)
  - De développer de véritables applications fonctionnant exclusivement dans le cadre d'Internet.

Caractéristiques principales

- Le Javascript est :
  - Ecrit directement dans le document HTML
  - Un script encadré par des balises HTML
  - Exécuté chez le client (pas d'appel réseau)
  - Interprété (pas compilé)
- Supporté par la plupart des navigateurs web
- Syntaxe proche du C

JAVASCRIPT SYNTAXE

LI288 – web et développement web

HTML et JavaScript

- Deux types d'insertion (comme CSS)
  - Directement dans le fichier HTML
  - Dans un fichier externe et inclus en HTML

- Utilisation de balises spécifiques :
  - <script type="text/javascript">…</script>

Insertion dans une page HTML

- Dans le corps de la page HTML
  - Le code s'exécute lors du chargement de la page

```
<html>
<body>
<script type="text/javascript">
  alert('bonjour');
</script>
</body>
</html>
```

Insertion dans une page HTML

- Dans l'entête de la page
  - Le code s'exécute lors d'un événement venant de l'utilisateur
  - Le code correspondant à cet événement se trouve dans le corps du document.

- En pratique les deux sont similaires sur la plupart des navigateurs

```
<html>
<head>
<script type="text/javascript">
   function f () { alert('Au revoir'); }
</script>
</head>
<body onUnload="f()">
</body>
</html>
```

Insertion dans un fichier externe et inclus en HTML

- A placer dans le <head> ou le <body>
  - Fichier en format texte
  - Permet de réutiliser les scripts dans plusieurs pages
  - Inconvénient : requête supplémentaire vers le serveur

```
<html>
<head>
  <script type="text/javascript" src="fichier.js"></script>
</head>

<body>
  <input type="button" onclick="popup()" value="Clic">
</ body>
</html>
```

Structure d'un script

- Similaire à Java ou C

- Règles générales
  - On peut mettre des espaces n'importe où
  - On sépare les commandes par des point-virgule ";"
  - Les réels sont notés avec un "." et pas une virgule ","
  - Commentaires : //… ou /*…*/
    - // ceci est un commentaire
    - /* ceci est aussi un commentaire */

Les variables

- Déclaration et affectation
  - Déclaration avec le mot clé "var"
  - Affectation avec le signe d'égalité (=)
- Remarques :
  - La déclaration est faite par défaut (si affectation sans déclaration préalable)
  - La lecture d'une variable non déclarée provoque une erreur
  - Une variable déclarée non affectée est de type undefined (indéfinie)

```
//Déclaration de i, de j et de k.
var i, j, k;
 //Affectation de i.
i = 1;
//Déclaration et affectation de prix.
var prix = 0;

//Déclaration et affectation d'un tableau
var car = ["a", "b", "c"];
```

Les variables

- Contraintes concernant les noms de variables :
  - Les noms de variables ne peuvent contenir que des lettres, chiffres, ou le caractère "_" (underscore)
    - var Mon_Prenom; // correct
  - Les caractères spéciaux et accentués sont interdits :
    - var Mon_Prénom; // incorrect
  - Attention aux majuscules et minuscules :
    - MonPrenom est différent de Monprenom.
  - Un nom de variable ne peut contenir d'espaces.
    - var Mon Prenom; // incorrect
  - Les mots réservés JavaScript ne peuvent être utilisés comme noms de variable.

Les types de variables

- Principaux types :
  - Chaînes
  - Nombre (entier ou décimaux) : 10^-308 >nombre < 10^308
  - 3 valeurs spéciales :
    - Positive Infinity ou +Infinity (valeur infini positive)
    - Negative Infinity ou –Infinity (valeur infinie négative)
    - Nan (Not a Number) en général le résultat d'une opération incorrecte
  - Boolean
    - true (vrai) et false (faux)

- Le type d'une variable dépend de son contenu
  - var maVariable = "Philippe"; // type chaîne
    maVariable =10; // type nombre (entier)

Portée des variables

- Globale :
  - Variable déclarée en début de script
  - Accessible à n'importe quel endroit du programme

- Locale :
  - Variable déclarée à l'intérieur d'une fonction
  - Accessible uniquement dans la fonction

Syntaxe, les boucles

- Boucle for :
  - for (i=0; i<5; i++) {…}
- Boucle while :
  - while (test) {…}
  - do {…} while (test)

Syntaxe, les conditions

- if (test) {} else {}

- Tests possibles :
  - Egalité : ==, !=
  - Inférieur, supérieur : =<, >=, >, <
  - Opérations bit à bit : &, |
  - Identique à : ===, !== (teste valeur et type)
    - ('1' == 1) // true
    - ('1' === 1) // false
  - Opérations logiques : &&, | |

Les fonctions

- Définition :
  - function maFonction(arg1,arg2) {instr;}
- Appel :
  - maFonction("12",13);

- Exemple : calcul de la fonction factoriel
  - Calcul récursif
  - Choix (if)
- Mais aussi (à venir) :
  - Appelé sur un événement
  - Utilisation de document.getElementById
  - Utilisation de this.value

Arguments

- Pas de type dans la signature de la fonction
- La déclaration d'arguments est optionnelle
  - On peut déclarer une fonction sans arguments
  - On peut ensuite lui passer des arguments
  - Et y accéder dans la fonction (via la variable arguments)

```
function test(argument1) {
  alert("argument : " + argument1);
  alert("nombre d'args : " + arguments.length);
  for (var i=0 ; i<arguments.length ; i++) {
    alert("arg " + i + " : " + arguments[i]);
  }
}

test(); // undefined - 0
test("1"); // 1 – 1 - 1
test("1","2"); // 1 – 2 – 1 - 2
```

Opérations sur les chaînes

- La concaténation : utilisation de +
  - var chaine = "bonjour" + "FI3/FCD1";
- Déterminer la longueur d'une chaîne : attribut length
  - var ch1 = 'bonjour';
  - var longueur = ch1.length;
- Extraction d'une partie de la chaîne :
  - var dateDuJour = "04/04/03"
  - var carac = dateDuJour.charAt(2);
    - Extrait le caractère en position 2 de la chaîne
  - var mois = datteDuJour.substring(3, 5);
    - 3: est l'indice du premier caractère de la sous-chaîne à extraire
    - 5 : indice du dernier caractère à prendre en considération ; ce caractère ne fera pas partie de la sous-chaîne à extraire

JAVASCRIPT ÉVÉNEMENTS

LI288 – web et développement web

Gestionnaire d'événements

- Les événements servent à interagir avec l'utilisateur
  - On peut détecter les clics, les modifications de formulaires, …
- Chaque événement a un identifiant
  - De la forme onQuelqueChose
  - Par exemple : onLoad, onClick, onMouseOver, etc.
- Un événement peut exécuter du code javascript
  - Une ou plusieurs instructions, en général un appel de fonction
- Activation :
  - <balise … onQuelqueChose="code javascript;">

Les événements de base

- Evénement onLoad
  - Se produit lorsque une page web est chargée dans la fenêtre du navigateur
  - Toute la page (y compris les images qu'elle contient si leur chargement est prévu) doit avoir été chargée pour qu'il ait lieu
  - Cet événement peut être associé à une image seulement (il se produit alors une fois le chargement terminé)

Les événements de base

- Evénement onClick
  - Se produit lorsque l'utilisateur clique sur un élément spécifique dans une page, comme un lien hypertexte, une image, un bouton, du texte, etc.
  - Ces éléments sont capables de répondre séparément à cet événement
  - Il peut également être déclenché lorsque l'utilisateur clique n'importe où sur la page s'il a été associé non pas à un élément spécifique, mais à l'élément body tout entier

Une liste plus longue

- Globales :
  - onAbort : chargement d'une image interrompu
  - onError : une erreur durant le chargement de la page
  - onLoad : chargement de la page
  - onUnload : l'utilisateur quitte la page

- Souris :
  - onBlur : un élément perd le focus
  - onClick : clic sur l'élément
  - onDblclick: double clic sur l'élément
  - onDragdrop : drag and drop sur la fenêtre du navigateur
  - onFocus : le focus est donné à un élément
  - onMouseOver : la souris passe sur un élément
  - onMouseOut : la souris quitte un élément
  - onResize : la fenêtre est redimensionnée

Une liste plus longue

- Formulaires :
  - onChange : modification d'un champ de données
  - onReset : effacement d'un formulaire à l'aide du bouton Reset.
  - onSelect : sélection d'un texte dans un champ "text" ou "textarea"
  - onSubmit : clic sur le bouton de soumission d'un formulaire

- Clavier :
  - onKeyDown : appui sur une touche du clavier
  - onKeyPress : appui et maintient sur une touche
  - onKeyUp : relâchement d'une touche

- Attention, selon les versions de javascript, les événements peuvent ne pas exister.

JAVASCRIPT MANIPULATION DE PAGE

LI288 – web et développement web

Entrées/sorties

- 3 types de boites de messages peuvent être affichés en utilisant Javascript
  - Méthode alert()
    - sert à afficher à l'utilisateur des informations simples de type texte. Une fois que ce dernier a lu le message, il doit cliquer sur OK pour faire disparaître la boîte
  - Méthode confirm()
    - permet à l'utilisateur de choisir entre les boutons OK et Annuler.
  - Méthode prompt()
    - La méthode prompt() permet à l'utilisateur de taper son propre message en réponse à la question posée

- La méthode document.write permet d 'écrire du code HTML dans la page WEB

Entrées/sorties

```
<html>
<head>
<title> une page simple </title>
</head>
<body>
  <script type="text/javascript">
    alert('bonjour');
    document.write (
      prompt('quel est votre nom ?','Indiquer votre nom ici')
    );
    confirm('quel bouton allez-vous choisir ?');
  </script>
</body>
</html>
```

Les objets du navigateur

- L'objet le plus haut dans la hiérarchie est "window" qui correspond à la fenêtre même du navigateur.
- L'objet "document" fait référence au contenu de la fenêtre :
  - "document" = ensemble des éléments HTML de la page.

- On peut accéder ces éléments avec :
  - méthodes propres à l'objet document :
    - getElementById( ) trouve l'élément avec son identifiant (ID)
    - getElementsByName
  - soit des collections d'objets qui regroupent sous forme de tableaux Javascript tous les éléments de type déterminé.

L'objet window

- Propriétés : (accessibles avec IE et N)
  - closed : indique que la fenêtre a été fermée
  - defaultStatus : indique le message par défaut dans la barre de status
  - document : retourne l'objet document de la fenêtre
  - frames : retourne la collection de cadres dans la fenêtre
  - history : retourne l'historique de la session de navigation
  - location : retourne l'adresse actuellement visitée
  - name : indique le nom de la fenêtre
  - navigator : retourne le navigateur utilisé
  - opener : retourne l'objet window qui a créé la fenêtre en cours
  - parent : retourne l'objet window immédiatemment supérieur dans la hiérarchie
  - self : retourne l'objet window correspondant à la fenêtre en cours
  - status : indique le message affiché dans la barre de status
  - top : retourne l'objet window le plus haut dans la hiérarchie

L'objet window

- Méthodes :
  - blur( ) : enlève le focus de la fenêtre
  - close( ) : ferme la fenêtre
  - focus( ) : place le focus sur la fenêtre
  - moveBy( ) : déplace d'une distance
  - moveTo( ) : déplace la fenêtre vers un point spécifié
  - open( ) : ouvre une nouvelle fenêtre
  - print( ) : imprime le contenu de la fenêtre
  - resizeBy( ) : redimensionne d'un certain rapport
  - resizeTo( ) : redimensionne la fenêtre
  - setTimeout( ) : évalue une chaîne de caractère après un certain laps de temps

L'objet document

- Propriétés :
  - applets, forms, images, links : retourne les collection d'applets java, formulaires… présents dans le document
  - cookie : permet de stocker un cookie
  - domain : indique le nom de domaine du serveur ayant apporté le document
  - referrer : indique l'adresse de la page précédente
  - title : indique le titre du document

```
<html><head><title>Test</title></head>
<body>
<a href="http://www.yahoo.fr/">Yahoo</a>
<a href="http://www.google.fr/">Google</a>
<script type="text/javascript">
  alert(document.title);
  for(var i=0; i < document.links.length; ++i)
    alert(document.links[i]);
</script>
</body></html>
```

L'objet document

- Méthodes :
  - close( ) : ferme le document en écriture;
  - open( ) : ouvre le document en écriture;
  - write( ) : écrit dans le document;
  - writeln( ) : écrit dans le document et effectue un retour à la ligne

L'objet navigator

- Propriétés
  - appName : application (Netscape, Internet Explorer)
  - appVersion : numero de version.
  - platform : système d'exploitation (Win32)
  - plugins
  - language
  - mimeTypes
  - JavaEnabled()

Manipulation des objets

- Pour adresser un objet, il faut préciser son « chemin d'accès » dans l'arborescence de la structure (DOM).
- Si le nom de la fenêtre est omis, le navigateur utilisera par défaut la fenêtre courante (attention aux frames)
- On peut omettre window.document (un seul objet "document")

```
<html>
<body onLoad="window.document.f1.zone.value='Bonjour';">
<form name="f1">
  <input name="zone" type="text">
</form>
</body>
</html>
```

Exemples

```
<html><head>
<script type="text/javascript">
function changeCouleur(color) {
  var ml = document.getElementById("maListe");
  ml.setAttribute("style","color:"+color);
}
</script>
</head><body>

<ul id="maListe">
  <li><a href="javascript: changeCouleur ('red');">Rouge</a></li>
  <li><a href="javascript: changeCouleur ('blue');">Bleu</a></li>
  <li><a href="javascript: changeCouleur ('black');">Noir</a></li>
</ul>
</body></html>
```

Pour aller plus loin

- Toutes les commandes sur :
  - https://developer.mozilla.org/fr/DOM/element#Propriétés

- Pour tester/débugger :
  - Firebug
  - Console d'erreur.
  - Utiliser des alertes.

JSON
JAVASCRIPT OBJECT NOTATION

LI288 – web et développement web

JSON ?

- Format d'échange de données.

- Objectifs :
  - Simple.
  - Extensible.
  - Ouvert.
  - Lisible par un humain.

- Similaire à la définition des objets Javascript.

JSON : JavaScript Object Notation

- Les types de base
  - Nombres entiers, réels ou à virgule flottante
  - Chaînes de caractères
  - Booléen true et false
  - Tableaux [..., ...] ou tableaux associatifs (objets) "clé":valeur : {..., ...}
  - null

```
{
  "Nom":"Guillaume",
  "Adresse": {"rue":"4 place Jussieu", "cp":"75004",
   "ville":"Paris"},
  "notes": [1, 2, 4, 8, 16, 32]
}
```

JSON et Javascript

- JSON peut être utilisé directement :
  - Inclusion dans du HTML
    - `<script> var data = JSONdata; </script>`
  - Peut être converti en un objet Javascript
    - `responseData = eval('(' + responseText + ')');`
- JSON peut être inclus à partir d'un fichier externe

JSON ou XML ?

- JSON est très utilisé avec AJAX (le X de AJAX est pour XML)

- JSON :
  - {"nom": "Guillaume", "prenom": "Jean-Loup"}

- XML :
  - <?xml version='1.0‘ encoding='UTF-8'?>
  - <element>
  - <nom>Guillaume</nom>
  - <prenom>Jean-Loup</prenom>
  - </element>

JSON ou XML

- Taille des données :
  - plus petite en JSON (pas de fermeture de tag)
  - XML se compresse mieux
- Vitesse :
  - XML se parse mieux
  - JSON s'évalue avec eval : peu efficace (pour l'instant)
- Choix :
  - JSON : structures de données
  - XML : structuration de documents
- A vous de faire votre choix !

JQUERY

LI288 – web et développement web

jQuery

- Librairie Javascript qui permet de :
  - Simplifier les taches de base en Javascript.
  - Accéder à des partie d'une page HTML.
  - Modifier l'apparence et le contenu de la page.
  - Créer et modifier des événements.
  - Animer une page.
  - Faire de l'AJAX.
  - Eviter les problèmes de compatibilité entre navigateurs.

jQuery

- Projet open-source, bien supporté et documenté.
  - http://jQuery.com/
- Ne se substitue pas à la connaissance de Javascript
  - Tout ne se fait pas naturellement en jQuery.
  - Eviter de tout "Jqueriser".
- Beaucoup d'autres librairies similaires :
  - Mootools, Prototype, Dojo, script.aculo.us, …
  - Jquery est efficace, légère, bien documentée

Philosophie jQuery

- Sélectionner une partie du document.
- Agir dessus

- Objet jQuery = ensemble de nœuds du DOM
  - C'est-à-dire un ensemble de balises du document.
- Les objets jQuery se créent avec la fonction $().
  - $("div") retourne tous les "div" du document.
  - $(" ") crée une nouvelle balise div.

Philosophie jQuery

- Sélectionner une partie du document.
- Agir dessus

```
// récupérer tous les div du DOM puis les cacher
$("div").hide();

// Simuler un clic sur tous les boutons
$(":button").click();

// Créer un <span> et l'ajouter à la fin du body
$("<span/>").appendTo("body");
```

Inclure du jQuery

- Télécharger le fichier jQuery.js
  - Il existe un version jQuery-min.js plus légère
  - Gain de bande passante mais fichier non lisible.

- Inclusion du fichier (comme tout javascript) :
  - En tant que script externe :
    - <script type="text/javascript" src="jQuery.js"></script>
  - Depuis le site de jQuery :
    - src="http://code.jquery.com/jquery-1.8.2.min.js"

Besoin d'aide

- http://jQuery.com/
  - Tout y est :
    - Le code sources
    - La doc sur tous les paramètres et toutes les fonctions.
    - Des exemples.
    - Plein de plugins.
    - …

Dans la suite

- Des exemples pour :
  - Sélectionner des objets.
  - Ajouter des effets d'animations.
  - Utiliser les événements.
  - Faire des requêtes "à la AJAX".
  - La librairie jQuery user interface.

JQUERY
COMMENT SÉLECTIONNER

LI288 – web et développement web

Sélecteur

- Principe de base :
  - Un sélecteur retourne un tableau d'objets jQuery :
    - $("*") retourne toutes les balises
    - $("div") est un tableau contenant tous les de la page.
    - $("div").length permet de connaître le nombre de div dans la page.

Sélection restreinte

- Possibilité de sélectionner (comme en CSS) :
  - Par type de bloc.
  - Par identifiant (attribut id d'une balise).
  - Par classe (attribut class d'une balise).

```
// <div>Test</div>
$("div")

// <span id="test">JL</span>
$("#test")

// <ul class="test">JL</ul>
$(".test")
```

Sélection restreinte

- Possibilité de combiner les sélecteurs.

```
// tous les divs de classe main
$("div.main")

// le tableau d'identifiant data
$("table#data")

// objets d'identifiant "content" ou de classe "menu"
// attention à la position des guillemets
$("#content, .menu")
```

Sélecteurs d'ordre

```
// premier paragraphe
p:first

// dernier élément de liste
li:last

// quatrième lien
a:eq(3)

// paragraphes pairs ou impairs
p:even ou p:odd

// Tous les liens à partir (greater than) du quatrième ou avant
   (lower than)
a:gt(3) ou a:lt(4)
```

Sélecteurs d'attributs

```
// les éléments <div /> ayant un identifiant
$("div[id]")

// les éléments <div /> avec id "test"
$("div[id='test']")

// les éléments <div /> dont l'id commence par test
$("div[id^='test']")

// les éléments <div /> dont l'id termine par test
$("div[id$='test']")

// les éléments <div /> dont l'id contient test
$("a[href*='test']")
```

Sélecteurs de formulaires

```
// sélectionner les checkbox
$("input:checkbox")

// sélectionner les boutons radio
$("input:radio")

// sélectionner les bouton
$(":button")

// sélectionner les champs texte
$(":text")
```

La fonction each

- appelle une fonction pour chaque élément :
  - $(this) = élément courant
  - i - index de l'élément courant
  - Possibilité de récupérer l'élément sous forme DOM

```
$("table tr")
   .each(function(i){
        if (i % 2==0)
               $(this).addClass("odd");
   });
```

JQUERY COMMENT MODIFIER

LI288 – web et développement web

Modifier le contenu HTML

- `html()` : renvoie le code HTML
- `html("...")` : remplace le code HTML

```
// modifier le contenu de tous les p du document
$("p").html("<div>Bonjour</div>");

// modifier le contenu de div.a en celui de div.c
$("div.a").html($("div.c").html());

// idem pour div.b avec échappement du contenu de div.c
$("div.b").text($("div.c").html());
```

```
<p></p>


<div id="a">Bonjour</div>

<div id="b"><a href="#">Au
   revoir</a></div>

<div id="c"><a href="#">Au
   revoir</a></div>
```

```
<p><div>Bonjour</div></p>


<div id="a"><a href="#">Au
   revoir</a></div>

<div id="b">&lt;a href="#"&gt;Au
   revoir&lt;/a&gt;</div>

<div id="c"><a href="#">Au
   revoir</a></div>
```

Modifier des valeurs

- val() : permet d'obtenir la valeur des objets
- val(valeur) permet de modifier la valeur des objets

```
// obtenir la valeur de la première checkbox cochée
$("input:checkbox:checked").val();

// modifier la valeur d'un champ text de nom txt
$(":text[name='txt']").val("Hello");

// sélectionne une valeur d'un select d'identifiant lst
$("#lst").val("NS");
```

Manipulation de CSS

- Il est possible de modifier les classes des objets :
  - addClass, removeClass, toggleClass
  - hasClass

```
// ajouter et supprimer une classe
$("p").removeClass("blue").addClass("red");

// ajouter si absent ou l'inverse
$("div").toggleClass("main");

// vérifier l'existence d'une classe
if ($("div").hasClass("main")) { //… }
```

Manipulation de CSS

```
// récupérer le style (un argument)
$("div").css("background-color");

// modifier le style (deux arguments)
$("div").css("float", "left");

// modifier le style, version rapide
$("div")
   .css({
        "color":"blue",
        "padding": "1em",
        "margin-right": "0",
        "margin-left": "10px"
   });
```

Insérer des éléments

```
// Ajouter à la fin de l'objet
// sélection des ul et ajout à la fin
$("ul").append("<li>Test</li>");

// création d'un objet <li>
// modification du contenu
// ajout à la fin des ul
$("<li />").html("Test").appendTo("ul");

// Ajouter au début
$("ul").prepend("<li>Test</li>");
$("<li />").html("Test").prependTo("ul");
```

Remplacer des éléments

```
// remplace tous les <h1> par des <div>Test</div>
$("h1").replaceWith("<div>Test</div>");

// replaceAll fonctionne à l'envers
$("<div>Test</div>").replaceAll("h1");

// Sans modifier le contenu :
$("h1").each(function(){
   $(this)
        .replaceWith("<div>"+ $(this).html()+"</div>");
   });
```

Supprimer des éléments

```
// supprimer tous les span de classe names
$("span.names").remove();

// vider tout le contenu de l'objet d'identifiant mainContent
$("#mainContent").empty();

// déplacer un objet par suppression + création
// supprime tous les p et les ajoute à la fin du document
$("p")
   .remove()
   .appendTo("body");
```

JQUERY
LES EFFETS

LI288 – web et développement web

Apparition et disparition

```
// montrer un élément
$("div").show();

// montrer un élément lentement (slow=600ms)
$("div").show("slow");

// cacher un élément rapidement (fast=200ms)
$("div").hide("fast");

// inverser (montrer ou cacher) en une durée fixée
$("div").toggle(100);
```

Translation et fading

```
// Translation
$("div").slideUp();
$("div").slideDown("fast");
$("div").slideToggle(1000);

// Fading
$("div").fadeIn("fast");
$("div").fadeOut("normal");

// Fading avec opacité fixée
$("div").fadeTo("fast", 0.5);
```

Effet personnalisé

- .animate(options, durée, transition, complete, ...) :
  - Options : ensemble de propriétés CSS.
  - Transition : comment se déroule l'animation (linéaire ou pas).
  - Complete : callback exécuté après la fin de l'animation.
  - ...

```
// réduction de la largeur à 50% et changement d'opacité.
// Le tout en 1s.
$("div")
  .animate(
    {width: "50%", opacity: 0.5,},
    1000);

<div style="border:1px solid red">contenu du div</div>
```

JQUERY
LES ÉVÉNEMENTS

LI288 – web et développement web

Le premier événement

- Chargement de la page
  - $(document).ready(…)
  - Assez similaire à onLoad() :
    - onLoad : quand tout est chargé
    - ready : quand le DOM est prêt (avant chargement des images)

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

// similaire à
$(function() {...});
```

Gestionnaire d'événements

- Événements disponibles :
  - blur, focus, load, resize, scroll, unload, beforeunload, click, dblclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, change, select, submit, keydown, keypress, keyup, error

```
$("div").click( function() {
        //le code à exécuter
})
```

L'objet Event - attributs

- type :
  - nom de l'événement exécuté
- target :
  - objet qui a exécuté l'événement
  - cf propagation des événements
- currentTarget :
  - = this
- pageX et pageY :
  - position de la souris
- …

Un exemple : position de la souris

- Exemple avec événement lié au déplacement de la souris.
  - Via un événement lié à tout le document.

```
<div id="log"></div>

<script>
$(document)
   .bind('mousemove',function(e){
        $("#log")
                .text(e.pageX + ", " + e.pageY);
   });
</script>
```

JQUERY
FONCTIONNALITÉS "AJAX"

LI288 – web et développement web

Charger du contenu

- Pour mettre du contenu dans un objet :
  - C'est un peu plus simple qu'en "Ajax".

```
// version sans arguments :
// récupère fichier.html et met le contenu dans le div#content
$("div#content").load(
  "fichier.html"
);

// version avec arguments :
// appelle la page fichier.php?nom=guillaume
$("div#content").load(
  "fichier.php",
  {"nom":"guillaume"}
);
```

Charger du contenu

- get() : récupèrer des données sur le serveur
- post() : envoyer des données au serveur

```
// récupère le fichier fichier.html puis exécute la fonction
$.get(
  "fichier.html",
  function(data){
    alert(data);
  });


$.post(
  " fichier.html",
  function(data){
    alert(data);
  });
```

Récupérer du JSON - Javascript

```
$.getJSON(
   "fichier.json",
   function(users) {
        alert(users[0].name);
   });


$.getScript(
   "script.js",
   function(){
        ...;
   });
```

CONCLUSIONS

LI288 – web et développement web

jQuery

- Possibilité de récupérer des objets du DOM.
- Evénements et animations.
- AJAX.

- Tout ça de manière indépendante des navigateurs.

Mais encore

- En plus des fonctions, on a accès à de nombreuses fonctions d'interface utilisateur (jQuery-UI) :
  - Drag and drop
  - Tableaux triables
  - Accordéons
  - Eléments triables
  - Barres de progression
  - Onglets
  - …
- Plugins
  - plus de 4000 plugins actuellement

Sortable – éléments triables

```
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript" src="jquery-ui.js"></script>
<script type="text/javascript">
  $(function() {
    $( "#sortable" ).sortable();
  });
</script>
</head>
<body>
<ul id="sortable">
   <li>Item 1</li>
   <li>Item 2</li>
   <li>Item 3</li>
   <li>Item 4</li>
</ul>
</body>
```

Draggable / droppable

```
<script type="text/javascript">
  $(function() {
    $( "#draggable" ).draggable();
    $( "#droppable" ).droppable({drop: function( event, ui ) {
      $( this ).html( "laché!" );}});
  });
</script>
</head>

<body>
<div id="draggable" >move</div>
<div id="droppable" style="border:1px solid red">on peut
   dropper ici</div>
```

Diaporama

```
/* code jquery */
function slideShowStart() {
  $('#gallery img').hide();
  $('#gallery img:first').addClass('show').show();
  $('#gallery img').click(nextImage);
  setInterval('nextImage()',3000);
}
function nextImage() {
  var current = $('#gallery img.show');
  var next = (current.next().length)?current.next():$('#gallery img:first');
  next.addClass('show').show();
  current.removeClass('show').hide();
}
$(document).ready(function() {slideShowStart();});

/* code html */
<div id="gallery">
  <img src="image1.jpg" alt="Image 1" />
  <img src="image2.jpg" alt="Image 2" />
  <img src="image3.jpg" alt="Image 3" />
</div>
```