

3I017 - TECHNOLOGIES DU WEB Les bases en Javascript

Thursday 14th March, 2019

Laure Soulier





Le web dynamique



- HTML / CSS permettent de produire des pages Web avec dispositions graphiques évoluées mais
 - Pages peu dynamiques
 - Interactions utilisateur limitées
- Langages de script pour le Web
 - Permettent de combler ce manque en donnant la possibilité de définir des fonctions

De modification des éléments affichés (modification éléments / attributs / propriétés de l'arbre DOM)

De réaction à des actions utilisateur

De communication client / serveur

- Exécution de traitements côté client
- Fortement dépendants du navigateur appelant la page web

L3 Info - 31017 1/75

</html> L3 Info - 31017

```
    Statique

<html>
<head>
<title > Page statique </title >
</head>
<body>
 La date du jour est le 25 fevrier 2019 
</body>
</html>

    Dynamique

<html>
<head>
<title > Page dynamique </title >
</head>
<body>
<script type = "text/javascript">
    date = new Date();
    document.writeln("La date du jour est le ", date);
</script>
</body>
```



- Différents langages
 - Javascript (Netscape & Sun)
 - VBscript (Microsoft)
 - XUL (Mozilla)
 - XSLT (W3C)
- European Computer Manufactures Association (ECMA)
 - ⇒ Standard ECMA 262

L3 Info - 31017 3/75

Javascript



- Javascript ≠ Java
 - Java : Langage objet compilé, qui peut être exécuté côté client par le biais d'une applet
 - Javascript : Langage de scripts moins évolué, intégré aux pages Web, qui s'inspire de différents langages (dont Java)

Javascript	Applet Java
Langage interprété	Langage compilé
Interprétation par le navigateur	Chargement d'une machine virtuelle
Code intégré au HTML	Code appelé à partir de la page
Langage peu typé	Langage fortement typé
Accessibilité du code	Confidentialité du code

L3 Info - 3I017 4/75

Javascript



- Un langage interprété
- Sans typage (mais librairies typescript / flow)
- ... qui évolue vite (ECMA-262 9th Edition en 2018)
- · ... plus vite que les navigateurs !
 - Babel: "transpileur" Javascript (en CSS, post-css)
 - Polyfill.io
- Liens avec la page Web:
 - Événements (DOM et autre)
 - Accès aux informations via des variables globales (ex. document)

L3 Info - 3I017 5/75



Insérer du code Javascript

Javascript dans balises

```
<!-- Pour differentes balises
   (avec eventHandler le nom d'un gestionnaire d'evenement) -->
<balise eventHandler="javascript:(function(){...})()" >
<!-- ou en tant qu'action d'un formulaire -->
<form action="javascript:(function(){...})()" >
```

Javascript dans <script> ... </script>

<script type="text/javascript"> Code javascript </script>

Javascript dans un ou plusieurs fichiers séparés

<script type="text/javascript" src="scripts/code.js"></script>

L3 Info - 31017 6/75

Javascript



```
Exemple premier code Javascript

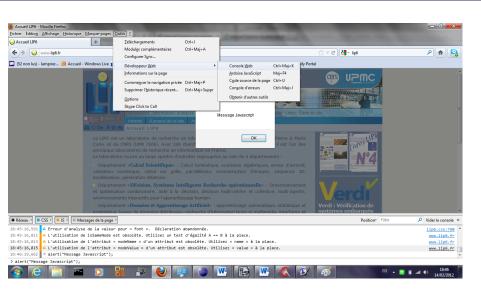
<html>
<html>
<htead>
<titte>Exemple Javascript</titte>
</html>
<br/>
<BODY>
<SCRIPT type="text/javascript">
<!--
alert("Message Javascript");
// -->
</sCRIPT>
</br/>
</br/>
// BODY>
</html>
```



L3 Info - 31017 7/75

Debuguer du Javascript : Console Web





L3 Info - 31017 8/75

Javascript



- Le langage Javascript
- Manipulation d'objets DOM
- La librairie JQuery ⇒ ReactJS

L3 Info - 31017 9/75

Langage Javascript

Javascript : Les variables



- · Variables :
 - Conteneurs pouvant accueillir toutes sortes de choses
 - En Javascript, pas de type défini lors de la déclaration
- Deux types de variables
 - Variables à portée locale
 - Variable déclarée à l'intérieur d'une fonction et avec le mot-clé var
 - ⇒ Visible uniquement dans la fonction où elle est déclarée
 - Variables à portée globale
 - Variable déclarée à l'extérieur d'une fonction ou sans le mot-clé var
 - ⇒ Visible partout

L3 Info - 31017 10/75



Variables locales / Variables globales

```
i = 0; // Variable globale
var j = 'Technos Web'; // Variable globale

function f(){
  var k = 'Javascript'; // Variable locale
  l = 20; // Varibale globale (car sans le mot-cle var)
  return i*l; // x est accessible ici car il s'agit d'une variable
}
// k n'est plus accessible ici (car locale)
```

L3 Info - 31017 11/75

Javascript: Les Fonctions



- Une fonction est un bloc d'instructions acceptant une liste de paramètres
 - Définie par le mot-clé function
 - Peut posséder un nom
 - Retourne éventuellement une valeur (grâce à return)

Fonction Javascript

```
function nom_fonction(parametre_1, ..., parametre_n) {
   instructions;
   return expression;
}
```

• On peut stocker une fonction dans une variable

Fonction anonyme

```
var f = function(parametre_1, ..., parametre_n) {
   instructions;
   return expression;
};
```

L3 Info - 31017 12/75

Javascript: Les Fonctions



- Paramètres de fonctions
 - Pas obligatoirement même nombre de paramètres à l'appel et dans la définition

Affectation des paramètres dans l'ordre

Les paramètres peuvent être accédés par l'objet arguments ⇒ <u>arguments[n-1]</u> correspond à la n-ième valeur passée en paramètre

Si nombre de paramètres à l'appel < nombre de paramètres attendus \Rightarrow paramètres supplémentaires = $\underline{\text{undefined}}$

Si nombre de paramètres à l'appel > nombre de paramètres attendus ⇒ paramètres additionnels peuvent être récupérés par l'objet arguments

 Les paramètres sont passés par référence sauf les types de bases (nombres, caractères,...) qui sont passés par valeur (copie)

Paramètres

L3 Info - 31017

- Une fonction peut être définie à l'intérieur d'une autre
 - La fonction interne peut accéder aux variables locales de la fonction externe...
 - ... Et s'en souvient même après en être sortie
 - ⇒ On parle de fermeture

Fermeture

L3 Info - 31017 14/75



- Un tableau est un ensemble d'éléments repérés par leur indice (entier commençant à 0)
 - Création

Création littérale

Exemple : var tab = [0,1,2,3,4,5];

Création grâce au constructeur Array

Exemple : var tab = new Array(10);

Accès aux éléments par la notation tableau[indice]

Exemple: tab[0]=10; var x=tab[0];

- Propriété length qui représente la longueur du tableau

L'ajout d'un élément à un indice supérieur à cette taille, augmente automatiquement length

Décrémenter length revient à supprimer le dernier élément

- Méthodes spécifiques comme join, slice, et push

L3 Info - 31017 15/75

Javascript: La notion d'objets



- Contrairement aux langages objets classiques (Java, C++, ...)
 - La notion de classe n'existe pas
 - Le langage n'est pas typé
- En JavaScript, un objet correspond à une sorte de tableau associatif
 - Chaque élément d'un objet correspond à une "entrée" (identifiée par un nom)
 - Un attribut correspond à une entrée avec un type quelconque Une méthode correspond à une entrée dont le type attendu est fonction
 - Manipulation dynamique
 - Possibilité d'ajouter, modifier ou supprimer les entrées de l'objet tout au long de sa vie
 - Mais pas de length, ni fonctions join, splice, push, etc...

L3 Info - 31017 16/75

Javascript : Les objets



- Création d'objets
 - Deux modes de création
 - Création à l'aide de constructeur Construction littérale
- Création par constructeur
 - Un constructeur est une fonction qui associe des valeurs (pouvant être des fonctions) à des attributs
 - Un constructeur est appelé par le mot-clé new
 - Constructeur prédéfini <u>Object</u>
 Exemple : var obj=new <u>Object()</u>;
- Construction littérale
 - Déclaration entre {...} d'associations attributs-valeurs
 Exemple : var obj={x : 1, y : 'chaine'};
 - Les valeurs associées peuvent elles-mêmes être des objets

⇒ Base du format ISON

L3 Info - 31017 17/75

Javascript : Les objets



- Les propriétés des objets peuvent être créées, lues et écrites
 - Soit avec la notation "point" objet.propriété :
 Exemple : var x=obj.couleur;
 - Soit avec la syntaxe utilisée pour les éléments de tableau :
 Exemple : var x=obj['couleur'];

L3 Info - 31017 18/75

Objets Javascript

```
var obj = new Object();
obi["attribut"] = "valeur1";
// ou
obj.attribut = "valeur1";
var fonc=function(parametre1, parametre2) {
    alert("parametres: " + parametre1 + ", " + parametre2);
};
obi["methode"]=fonc;
// ou
obi.methode=fonc;
// Affichage de la valeur "attribut" de obj
alert("Valeur de attribut: " + obj.attribut);
// Execution de la m thode methode de obj
obj.methode("valeur1", "valeur2");
```

L3 Info - 31017



- Constructeurs
 - Simples fonctions d'association propriété-valeur
 - Pas de notions de classes comme dans autres langages
 - Par convention, noms de constructeurs commencent par une majuscule
- Possibilité d'écrire ses propres constructeurs d'objets
 - Utilisation du mot-clé this

this se rapporte à l'objet sur lequel a été appelé la fonction Exemple : l'objet x dans x.a()

Dans le cas d'un constructeur, appel avec new

- ⇒ this se rapporte alors à l'objet nouvellement créé
- Affectation de valeurs à différentes propriétés de l'objet Exemple : this.couleur=2;

L3 Info - 31017 20/75



Constructeur d'objets

```
function MonObjet(param1, param2) {
  this.attribut1 = param1;
  this.attribut2 = param2;
objet = new MonObjet(5, 'bleu');
alert(objet.attribut1); // affiche 5 alert(objet["attribut2"]); // affiche "bleu"
objet.attribut3 = new Date(); // ajoute une nouvelle propriete a l'objet
alert (objet.attribut3); // affiche la date d'ajout de attribut3
MonObjet.statique = "serif"; // ajoute une propriete statique
alert (MonObjet.statique);  // affiche "serif"
alert (objet.statique);  // affiche undefined
delete objet.attribut2;  // enleve une prop
alert(objet.attribut2);  // affiche undefined
                                   // enleve une propriete a l'objet
delete objet;
                                   // supprime l'objet entier (rarement utilise)
alert (objet.attribut1); // declenche une exception
```

L3 Info - 31017 21/75

Javascript : Les objets



- this permet d'affecter des attributs / méthodes à un objet
 - On peut y accéder de l'extérieur du constructeur
 - ⇒ Ils sont donc publics
- Grâce au mécanisme de fermeture, on peut définir des attributs / méthodes privés :

Constructeur d'objets

```
function MonObjet(val) {
    var a = val:
    this.getA = function() {
      return(a);
    this.setA = function(newVal) {
      a = newVal;
var obj = new MonObjet(1);
console.log("A: " + obj.a); // Affiche undefined (a non d fini)
console.log("A: " + obj.getA()); // Affiche "A: 1" (getA est un accesseur)
obi.a=2:
console.log("A: " + obj.getA()); // Affiche "A: 1" (la valeur n'a pas change)
obj.setA(2);
console.log("A: " + obj.getA()); // Affiche "A: 2" (setA est un modificateur)
```

L3 Info - 31017 22/75



• On peut faire la même chose avec les méthodes :

```
Constructeur d'objets
```

```
function MonObjet(parametre1, parametre2) {
    var attribut1 = parametre1;
    var attribut2 = parametre2;
    var methode privee = function() {
        console.log("Attributs: " + attribut1 + ", " + attribut2);
    this.methode publique = function() {
        methode privee();
var obj = new MonObjet(1, 2);
alert("Attribut1: " + obj. attribut1); // Affiche undefined
                                    //(attribut1 non defini)
obj.methode_privee(); // "TypeError: obj.methode_privee not a function"
obj.methode publique(); // Affiche "Attributs: 1, 2"
```

L3 Info - 31017 23/75

- Il n'y pas de types en javascript
- mais

```
class Rectangle {
  constructor(hauteur: number, largeur: number) {
    this.hauteur = hauteur:
    this.largeur = largeur;
  area() {
    return this.hauteur * this.largeur;
// Erreur!
let r = new Rectangle("1", "2");
// OK
let r = new Rectangle(1, 2);
```

L3 Info - 31017 24/75

Javascript : Construction littérale



- Comme on l'a vu précédemment :
 - Un objet peut être instancié par un constructeur...
 - ... ou formé par une déclaration littérale
- · Construction littérale :
 - Un objet se construit par { }
 - Un tableau se construit par []
 - Possibilité de former des imbrications d'objets / tableaux

Exemple de construction littérale

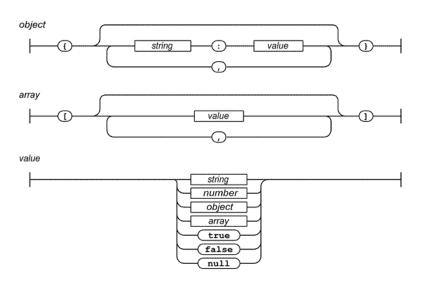
L3 Info - 31017 25/75



- JSON (JavaScript Object Notation)
 - Format de données textuel
 - Dérivée de la construction littérale d'objets
- Format JSON est composé :
 - d'ensembles de paires nom / valeur ⇒ les objets
 - de listes ordonnées de valeurs ⇒ les tableaux

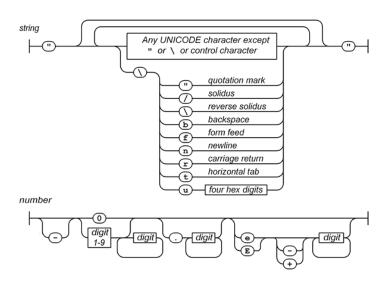
L3 Info - 31017 26/75





L3 Info - 31017 27/75





L3 Info - 31017 28/75



 Le format JSON = chaine de caractères correspondant à la formation littérale d'un objet

⇒ Nécessite de disposer :

 $D'un parser : texte JSON \Rightarrow objet$

D'un serializer : objet \Rightarrow texte JSON

L3 Info - 31017 29/75



- Parser : la fonction eval(string)
 - Permet d'interpréter une chaîne de caractères
 - Puisque le format JSON = chaine de construction littérale, eval('('+json_text+')') construit l'objet correspondant au texte contenu dans json text
- Mais:
 - eval est une fonction générique permettant d'évaluer n'importe quelle chaîne représentant du code Javascript
 - ⇒ Problèmes de sécurité car du code nuisible peut être exécuté lors de la transformation du texte JSON

Faille de sécurité

```
// JSON transmis par le serveur :
json_texte="{\"g\":1,\"f\":\"json\"}";
var obj=eval("("+json_texte+")"); // construction de l'objet
// JSON tronqu lors du transfert :
json_texte="function(){alert('Hack!')})(";
var obj=eval("("+json_texte+")"); // Affiche "Hack!"
```

L3 Info - 3I017 30/75



- Depuis 2009, les navigateurs intègrent un support JSON comportant un fonction parse(json text, reviver)
 - json text : chaîne JSON à transformer
 - reviver (facultatif): méthode appelée sur chaque couple (clé,valeur) à chaque niveau de la construction de l'objet
 - ⇒ Méthode spécifique n'interprétant pas d'autre code qu'une chaîne de construction JSON

JSON.parse

L3 Info - 31017 31/75



- Le support JSON comporte égalemment une fonction stringify(objet, replacer);
 - objet : objet à transformer en chaîne JSON
 - replacer (facultatif) : méthode appelée sur chaque couple (clé,valeur)
 à chaque niveau de la structure de l'objet pour spécifier un traitement spécial

JSON.stringify

```
//Objet serializer en JSON
var obj=new Object();
obj.g="1";
obj.f="json";

var json_text=JSON.stringify(obj);
// json_text contient "{\"g\":\"1\",\"f\":\"json\"}"
```

⇒ Attention : JSON.stringify ne serialise pas ce qui est dans le prototype

L3 Info - 31017 32/75

Exemples utilisation replacer et revival : objet Date()

```
var obj={g:1, r:new Date()};
obj; // Affiche "({g:1, r:(new Date(1329482734849))})" sur la console Web
json text = JSON. stringify (obj, function (key, value) {
                return this [key] instanceof Date ? 'Date(' + this [key] + ')' :
});
obj2=JSON.parse(json text, function (key, value) {
                var d:
                if (typeof value === 'string' &&
                   value.slice(0.5) == 'Date(' &&
                   value.slice(-1) === ')') {
                        d = new Date(value.slice(5, -1));
                        if (d) {
                                return d:
                return value:
});
obj2; Affiche "({g:1, r:(new Date(1329482734000))})" sur la console Web
```

L3 Info - 31017 33/75



Pour s'assurer que JSON est bien pris en charge par le navigateur :

```
<!-- Au cas ou pas JSON sur browser -->
<script type="text/javascript" src=
    "https://github.com/douglascrockford/JSON-js/blob/master/json2.js">
</script>
```

L3 Info - 31017 34/75

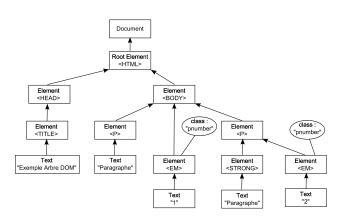
Arbre DOM



- Arbre DOM (Document Object Model)
 - Standard du W3C
 - Décrit une interface indépendante de tout langage de programmation
 - ⇒ Accéder ou de mettre à jour le contenu, la structure ou le style de documents XML et HTML
- Arbre BOM (Browser Object Model)
 - Pas de standard sur la manière de gérer les éléments du navigateur
 - Nécessité de produire du code javascript adapté à tout navigateur / système d'exploitation
 - ⇒ BOM sert d'interface standard entre le navigateur et javascript

L3 Info - 31017 35/75

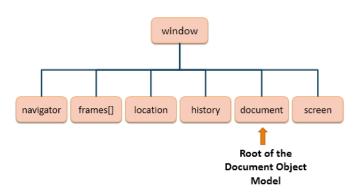




L3 Info - 31017 36/75

Browser Object Model (BOM)





L3 Info - 3I017 37/75

Javascript : Objets du Navigateur



- Lors de l'ouverture d'une page Web, le navigateur crée différents objets.
 Les principaux sont :
 - window : fenêtre d'affichage de la page (contient des propriétés sur la fenêtre mais aussi les objets de la page chargée)
 - navigator : qui contient des informations sur le navigateur
 - location : contient des informations relatives à l'adresse de la page à l'écran
 - history: historique de navigation (liste de liens visités précédemment)
 - document : contient les propriétés sur le contenu de la page Web chargée

L3 Info - 31017 38/75



- Objet Window : Fenêtre active du navigateur
- · Parent de tous les autres objets
 - window.navigator
 - window.location
 - window.document
 - window.history
 - window.frames
 - …
- · Méthodes relatives à window
 - méthodes pour l'ouverture de boîtes de dialogue alert(), confirm() et prompt()
 - méthodes pour l'ouverture ou la fermeture de fenêtres open() et close()
 - ⇒ open(x,[y]) permet de charger dans une fenêtre une page d'url x. Si y absent, on charge dans une nouvelle fenêtre, sinon on charge dans la fenêtre / onglet y (une nouvelle si pas encore de fenêtre nommée y)

L3 Info - 31017 39/75



- Objet Navigator : Permet d'avoir des informations sur le navigateur utilisé
 - Infos générales sur le navigateur : navigator.userAgent;
 - Nom du navigateur : <u>navigator.appName</u>;
 - Version du navigateur : navigator.appVersion;
 - Méthodes de test sur le nom du navigateur : <u>isIE()</u>, isFirefox, isSafari(), ...
 - Système d'exploitation du poste client : navigator.platform;
 - Langue utilisée par le navigateur : naviigator.language;
 - Types de données supportées par le navigateur : navigator.mimeTypes;
 - Liste de plugins installés : navigator.plugins;
 - Méthode pour savoir si le Java est autorisé : navigator.javaEnabled();
 - Pour savoir si les cookies sont autorisés : navigator.cookiesEnabled;

L3 Info - 31017 40/75

Javascript : Objet History



- Objet History: Contient des infos sur l'historique
 - La propriété <u>history.length</u> permet de connaître le nombre d'objets dans l'historique
 - La méthode <u>history.back()</u> permet d'aller à l'URL précédent dans l'historique
 - La méthode <u>history.forward()</u> permet d'aller à l'URL suivant dans l'historique
 - La méthode <u>history.go(variable)</u> permet d'aller à un des URL de l'historique (variable peut être un index de page ou une chaîne proche de la page désirée).

L3 Info - 31017 41/75

Javascript : Objet Location



- Objet Location : Contient les éléments de l'URL du document
 - location.protocol : contient la partie protocole de l'url (le plus souvent "http:")
 - location.port : Numéro du port utilisé (serveur web : port 80)
 - location.hostname : Domaine de la page
 - location.host : "domaine:port"
 - location.pathname : Répertoire et nom de fichier de la page
 - location.hash: Ancre de l'url ("presentation" dans "www.lip6.fr#presentation")
 - <u>location.search</u>: Liste des variables et leurs valeurs (par exemple "?nom=valeur&nom2=valeur2")
 - <u>location.href</u>: url complète (on peut changer cette propriété pour changer de page)
 - location.reload(): pour recharger la page courante

L3 Info - 31017 42/75

Javascript : Objet Document



- Objet Document : Contenu de la page
- Propriétés
 - alinkColor : couleur des liens lorsqu'ils sont cliqués
 - bgColor : couleur d'arrière plan
 - charset : jeu de caractères utilisés
 - cookie : chaîne de caractères pouvant être sauvegardée chez l'utilisateur
 - fgColor : couleur du texte
 - lastModified : date de dernière modification
 - linkColor : couleur des liensreferrer : pages déjà visitées
 - title : titre de la page

L3 Info - 3I017 43/75

Javascript: Cookies



- Un cookie est une paire attribut/valeur
 - Stocké dans le navigateur du client
 - Possède éventuellement une date péremption
 - Contient des informations envoyées par un serveur à un client HTTP
 - → Envoyé par le client en entête de chaque communication HTTP avec le serveur qui l'a créé
- Sert à enregistrer chez le client
 - Des données d'authentification
 - L'identifiant de la session en cours sur le serveur
 - Des préférences de l'utilisateur
 - Le contenu d'un panier d'achat électronique

L3 Info - 31017 44/75

Javascript: Objet Document



Gestion de cookies

```
function createCookie(name, value, days) {
        if (days) {
           var date = new Date();
           date.setTime(date.getTime()+(days*24*60*60*1000));
           var expires = "; expires="+date.toGMTString();
        else var expires = "";
        document.cookie = name+"="+value+expires+": path=/":
function readCookie(name) {
        var nameEQ = name + "=";
        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,c.length);
           if (c.indexOf(nameEQ) == 0)
                return c.substring(nameEQ.length,c.length);
        return null:
function eraseCookie(name) {
        createCookie (name, "", -1);
createCookie("a",1,10); // cr e un cookie a=1 expirant dans 10 jours
var val=readCookie("a"); // lit un cookie nomm a
```

L3 Info - 31017

Javascript : Objet Document



Méthodes

- createAttribute(nom attribut) : Crée un noeud attribut
- createElement(nom balise) : Crée un noeud élément
- createTextNode(texte) : Crée un noeud de texte
- getElementById(id): Retourne le noeud correspondant à l'identifiant passé en paramètre
- getElementsByName(nom) : Retourne un tableau contenant les éléments possédant le nom passé en paramètre
- getElementsByTagName(nom_balise): Retourne un tableau contenant les éléments du type passé en paramètre
- ⇒ Attention : manipuler arbre DOM suppose d'attendre la fin du chargement de la page. Les traitements devant être faits lors de de l'affichage doivent être appelés par le gestionnaire d'évènement document.onload.

L3 Info - 31017 46/75

- Node = Noeud de l'arbre DOM
 - Différents types de noeud

Numéro	Type de nœuds
1	Nœud élément
2	Nœud attribut
3	Nœud texte
4	Nœud pour passage CDATA
5	Nœud pour référence d'entité
6	Nœud pour entité
7	Nœud pour instruction de traitement
8	Nœud pour commentaire
9	Nœud document
10	Nœud type de document
11	Nœud de fragment de document
12	Nœud pour notation

L3 Info - 31017 47/75

Javascript : Node



- Node = Noeud de l'arbre DOM
- Propriétés
 - nodeType : numéro du type du noeud
 - nodeValue : contenu du noeud
 - nodeName : nom du noeud
 - attributes : tableau de noeuds attribut associé
 - firstChild : premier noeud enfantlastChild : dernier noeud enfant
 - nextSibling : noeud suivant dans l'arborescence
 - parentNode : noeud parent
 - previousSibling : noeud précédent dans l'arborescence

L3 Info - 31017 48/75

Javascript : Node



Méthodes

- appendData(texte) : concatène du texte en fin de la valeur d'un noeud texte ou attribut
- insertData(pos,texte): insère du texte à la position pos dans la valeur d'un noeud texte ou attribut
- replaceData(pos,nb,texte) : remplace du texte d'un noeud texte ou d'un noeud attribut (pos correspond à la position dans la chaîne et nb le nombre de caractères à supprimer)
- deleteData(pos,nb): efface du texte d'un noeud texte ou d'un noeud attribut

L3 Info - 31017 49/75

Javascript : Node



Méthodes

- getAttribute(nom) : retourne la valeur d'un noeud attribut
- getAttributeNode(nom) : retourne un noeud attribut
- setAttribute(nom,valeur) : fixe la valeur d'un noeud attribut
- setAttributeNode(node) : ajoute un noeud attribut
- removeAttribute(nom) : efface la valeur d'un noeud attribut
- removeAttributeNode(node) : supprime un noeud attribut

L3 Info - 3I017 50/75



Méthodes

- appendChild(node): ajoute un noeud enfant (en tant que dernier enfant)
- removeChild(node) : retire un noeud passé en paramètre de la liste des fils du noeud
- replaceChild(new,old): remplace un noeud enfant old par un noeud new dans la liste des fils du noeud
- hasChildNodes() : vérifie l'existence de noeuds enfants
- insertBefore(new,avant): insère un noeud enfant avant un autre noeud enfant dans la liste de fils du noeud (new = nouveau noeud, avant = noeud enfant avant lequel insérer)
- cloneNode(booleen) : copie un noeud (si le booléen passé en paramètre est true, alors on copie aussi les fils du noeud)

L3 Info - 31017 51/75



Manipulation arbre DOM

L3 Info - 31017 52/75



Gestionnaire d'évènement	Description
onClick	Lors d'un clic sur
ONCIICK	l'élément associé à l'événement
onLoad	Lorsque le navigateur charge
onLoad	la page en cours
onUnload	Lorsque le navigateur
OnOmoad	quitte la page en cours
onMouseOver	Lorsque le curseur de la souris
OnviouseOver	passe au-dessus de l'élément
onMouseOut	Lorsque le curseur de la souris
oniviouseout	quitte l'élément
onFocus	Lors de l'obtention du focus
on ocus	(élément sélectionné comme étant l'élément actif)
onBlur	Lors de la perte du focus
Olibiui	(l'utilisateur clique hors de l'élément)
onChange	Lors de la modification du contenu
Offichange	d'un champ de données
onSelect	Lors de la sélection d'un texte (ou une partie d'un texte)
OliSelect	dans un champ de type "text" ou "textarea"
onSubmit	Lors d'un clic sur un bouton de soumission
onsabilit	d'un formulaire (le bouton qui permet d'envoyer le formulaire)

L3 Info - 31017 53/75

JQuery

Javascript



La librairie JQuery

L3 Info - 3I017 54/75



- jQuery
 - Bibliothèque JavaScript libre qui porte sur l'interaction entre JavaScript et HTML
 - ⇒ Simplifier les commandes communes de JavaScript.
- La librairie JQuery inclus
 - Parcours et modification du DOM
 - Gestion d'évènements
 - Effets et animations
 - Manipulations des feuilles de style

- ...

L3 Info - 31017 55/75



Chargement librairie JQuery

```
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
</html>
```

L3 Info - 31017 56/75



• La bibliothèque ¡Query peut être appelée de trois manières différentes :

Via la fonction \$(expression) \$("div.test").add("p.quote").addClass("blue").slideDown("slow");

Via le préfixe jQuery(expression)

```
jQuery("#div a effacer").slideUp("fast");
```

Via le préfixe de fonction \$.

```
$.each([1,2,3], function() {
  document.write(this + 1);
});
```

L3 Info - 31017 57/75



- Fonction \$() permet différentes choses selon les paramètres
 - Une fonction
 - ⇒ Lancer cette fonction lorsque la page est chargée par le navigateur (comme window.onload mais n'attend pas que tout soit complètement téléchargé)

Processus de chargement

- 1. Le fichier html est t l charg .
- 2. Il est lu par le navigateur.
- 3. Le navigateur commence t l'charger les lments auxquels le fichier html fait r f rence (images, css...).
- 4. Le code situ dans \$(function(){}) est ex cut d s que possible.
- 5. Lorsque tout est t I charg , I code situ dans window.onload = function(){} est ex cut .

L3 Info - 31017 58/75



- Fonction \$() permet différentes choses selon les paramètres
 - Un élément, ou un tableau d'éléments DOM
 - ⇒ Pour créer un objet JQuery auquel on peut appliquer diverses fonctions

Création d'objet JQuery

```
var mon_element = document.getElementById("id_de_mon_element_html");
var mon_objet_jquery = $(mon_element);
```

L3 Info - 31017 59/75

- Fonction \$() permet différentes choses selon les paramètres
 - Le mot-clé this
 - ⇒ Pour créer un objet JQuery contenant l'objet sur lequel la fonction dans laquelle on se trouve a été appelée

Création d'objet JQuery

```
var mon_element = document.getElementById("main");
var jquery1=$(mon_element);
mon_element.getJQuery=function(){return($(this));}
var jquery2 = mon_element.getJQuery();
// => jquery1 et jquery2 quivalents
```

L3 Info - 31017 60/75



- Fonction \$() permet différentes choses selon les paramètres
 - Une expression en CSS
 - ⇒ Permet de sélectionner facilement des éléments
 - ⇒ Forme un objet JQuery regroupant les noeuds correspondant à l'expressions CSS

Sélection d'objets avec sélecteurs CSS

```
// S lection des Iments ayant pour classe "ique",
// dans l' Iment ayant pour id "fixe"
var mon_objet_jquery = $("#fixe .ique");

// S lection du premier Iment  de l' Iment <div >.
var mon_objet_jquery = $("div p:first-child");

// S lection des Iments , enfants directs d'un lement <div>
// et ayant pour classe "ique".
var mon_objet_jquery = $("div > p.ique");
```

L3 Info - 31017 61/75



- Fonction \$() permet différentes choses selon les paramètres
 - Un chemin XPath
 - ⇒ Permet de sélectionner facilement des éléments
 - ⇒ Forme un objet JQuery regroupant les noeuds correspondant aux chemins XPath

Sélection d'objets avec chemin XPath

```
// S lection des lments p enfants de body, lui-m me enfant de html  ("/html/body/p");
```

Documentation XPath en JQuery :

http://docs.jquery.com/DOM/Traversing/Selectors#XPath Selectors

L3 Info - 31017 62/75



- Fonction \$() permet différentes choses selon les paramètres
 - Une expression avec sélecteurs spécifiques à JQuery
 - ⇒ Permet de sélectionner facilement des éléments
 - ⇒ Forme un objet JQuery regroupant les noeuds correspondant à l'expression JQuery

Sélection d'objets avec sélecteurs JQuery

```
// S lection du deuxi me lment p de la page
$("p:nth(1)");

// S lection des div cach s
$("div:hidden");

// S lection des div contenant la chaîne de caract res "test"
$("div:contains('test')");
```

Documentation sélecteurs JQuery :

http://docs.jquery.com/DOM/Traversing/Selectors#Custom Selectors

L3 Info - 31017 63/75



- Fonction \$() permet de regrouper des noeuds correspondant à un sélecteur passé en paramètre
 - ⇒ Objet JQuery contenant des noeuds DOM
 - ⇒ Possibilités d'y appliquer des méthodes JQuery
- Attention :
 - Fonctions Node pas applicables directement
 - ⇒ Accès aux éléments de l'objet par la méthode JQuery get

get() retourne un tableau contenant les noeuds sélectionnés get(index) retourne le noeud à l'index index

Exemple utilisation JQuery

```
// Mauvaise utilisation JQuery (g n re une erreur)
$("#comment > p").firstChild.appendData("Post par Joe le 14/02/2012");

// Acc s au premier | lment de | 'objet et application m thode au noeud
$("#comment > p").get(0).firstChild.appendData("Post par Joe le 14/02/2012");

// quivalent :
$("#comment > p").get()[0].firstChild.appendData("Post par Joe le 14/02/2012");
```

L3 Info - 31017 64/75



- Objet JQuery
 - ⇒ Propriété length : nombre d'éléments contenus
 - ⇒ Éléments rangés entre index 0 et length-1
 - ⇒ Méthode slice(debut,fin) : retourne l'objet jquery contenant uniquement les objets d'un jquery initial situés entre l'index debut et fin-1

Exemple utilisation JQuery

```
// Application m thode tous les lments s lectionn s
var jquery=$("p");
for (i=0;i=jquery.length;i++){
          jquery.get(i).firstChild.appendData("Post par Joe le 14/02/2012");
}
```

L3 Info - 31017 65/75



- Objet JQuery
 - ⇒ Méthode each(func) : applique la function func à tous les éléments de l'objet

Exemple utilisation JQuery

```
<span> Premier paragraphe </span>
<span> Deuxi me paragraphe </span>
<span> Troisi me paragraphe </span>
// Remplacement du texte des span par "Paragraphe 1", "Paragraphe 2", ...
$("span").each(function(index) {
  if (this.firstChild!=undefined){this.firstChild.nodeValue="Paragraphe "+index;}
}).get().join(',');
                                              " div"
// Modification de la couleur des
                                      Iments
$ ( "div " ).each(function( i ) {
    if ( this.style.color !== "blue" ) {
      this . style . color = "blue":
    } else {
      this.style.color = "";
});
```

L3 Info - 31017 66/75



- Méthodes d'accès / modification de contenu
 - html() : retourne le code html du le premier élément de l'objet JQuery
 - html(code) : donne un code html à tous les éléments de l'objet
 - text() : retourne le texte contenu par le premier élément de l'objet (et ses descendants)
 - text(texte) : donne un texte à tous les éléments de l'objet
 - val() : retourne la valeur contenue par le premier élément de l'objet (et ses descendants)
 - val(valeur) : donne une valeur à tous les éléments de l'objet

L3 Info - 31017 67/75



- Méthodes de manipulation de l'arbre
 - before(x), after(x): insère x avant ou après tous les éléments de l'objet (x peut être du code html, un élément DOM ou un objet JQuery)
 - insertBefore(x), insertAfter(x): insère tous les éléments de l'objet
 JQuery avant ou après l'objet x (x peut être un objet DOM ou une expression JQuery permettant de sélectionner des objets)
 - append(x), prepend(x) : insère x à la fin ou au début de tous les éléments de l'objet
 - appendTo(x), prependTo(x): insère tous les éléments de l'objet
 JQuery à la fin ou au début de l'objet x
 - replaceWith(x): remplace tous les éléments de l'objet JQuery par x
 - replaceAll(x) : remplace x par les éléments de l'objet

L3 Info - 31017 68/75



Exemple utilisation JQuery

```
<div class='a'>
  < div class = 'b' > b < / div >
</div>
$('.a').after($('.c'));
<div class='a'>
  < div class = 'b' > b < / div >
</div>
<div class='c'>c</div>
$('.a').before($('.c'));
<div class='c'>c</div>
<div class='a'>
  < div class = 'b' > b < / div >
</div>
$('.a').append($('.c'));
<div class='a'>
  < div class = 'b' > b < / div >
  <div class='c'>c</div>
</div>
```

L3 Info - 31017 69/75



- Méthodes pour le CSS
 - css(nom) : retourne la valeur de la propriété css nom pour le premier élément de l'objet JQuery
 - css(nom,valeur) : donne une valeur à une propriété css pour tous les éléments de l'objet
 - height(), width(): retourne la hauteur et largeur calculées pour le premier élément de l'objet
 - height(hauteur), width(largeur) : donnent une hauteur et une largeur à tous les éléments de l'objet
 - offset(): retourne un objet contenant les coordonnées absolues du premier élément de l'objet
 - offset(coordinates) : donne les coordonnées de l'objet coordinates à tous les éléments de l'objet

L3 Info - 31017 70/75



Events

- click(fonc) : spécifie fonc comme fonction à lancer lors du clic d'un des éléments de l'objet JQuery
- keypress(fonc) : spécifie fonc comme fonction à lancer lors de la pression d'une touche (avec le focus sur un des éléments de l'objet)
- hover(fonc) : spécifie fonc comme fonction à lancer lors du passage au dessus d'un des éléments de l'objet
- focus(fonc) : spécifie fonc comme fonction à lancer lors de l'obtention du focus par un des éléments de l'objet
- resize(fonc) : spécifie fonc comme fonction à lancer lors du redimensionnement d'un des éléments de l'objet

— …

Exemple utilisation JQuery

```
$( "#target" ). click(function() {
alert( "Handler for . click() called ." );
});
```

L3 Info - 31017 71/75

Effets

 show(duree,[fonc]), hide(duree,[fonc]): affiche / cache petit à petit un élément en jouant sur la propriété display

Le paramètre duree permet de spécifier une vitesse à l'animation Le paramètre fonc correspond à une fonction à appeler lorsque l'action est terminée

 fadeln(duree,[fonc]), fadeOut(duree,[fonc]): affiche / cache petit à petit un élément en jouant sur la propriété opacity

> Le paramètre duree permet de spécifier une vitesse à l'animation Le paramètre fonc correspond à une fonction à appeler lorsque l'action est terminée

slideDown(duree,[fonc]), slideUp(duree,[fonc]): affiche / cache petit
à petit un élément en jouant sur la propriété height

Le paramètre duree permet de spécifier une vitesse à l'animation Le paramètre fonc correspond à une fonction à appeler lorsque l'action est terminée

L3 Info - 31017 72/75



- animate(objectif, duree, [fonc]) : Produit une animation de transfert entre l'état courant et un état visé
 - Le paramètre objectif est un ensemble de {propriété:valeur}
 - ⇒ Uniquement sur les propriétés numériques
 - ⇒ Replacer tirets par majuscules (exemple margin-left ⇒ marginLeft)
 - Le paramètre duree donne la durée de l'action
 - Le paramètre fonc est une fonction à lancer une fois l'action terminée

L3 Info - 31017 73/75



Animation JQuery

```
<!DOCTYPE html>
<html>
<head>
  <style>div { margin:3px; width:40px; height:40px;
        position:absolute; left:0px; top:60px;
        background: green; border: solid; } </style>
 <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
<div id></div>
<script>
var div = $("div");
function runlt() {
  div.hide("fast"); div.show("slow");
  div.fadeOut(2000): div.fadeIn(2000):
  div.slideUp(1000); div.slideDown(1000);
  div.animate({ left: '+=200'},2000); div.animate({ marginTop: '+=200'},2000);
  div . animate({borderLeftWidth: '+=200'},2000);
  div . animate ({ height: '+=200', width: '+=200'}, 2000);
  div.animate({ left: '-=200'},2000); div.animate({ marginTop: '-=200'},2000);
  div.animate({borderLeftWidth: '-=200'},2000);
  div.animate({height: '-=200', width: '-=200'},2000, runlt);
runlt();
</script>
</body>
</html>
```



Documentation Javascript : http://fr.selfhtml.org/javascript/index.htm Documentation Javascript : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Language_Resources

Documentation JQuery : http://api.jquery.com/

L3 Info - 31017 75/75