Cours 1 – M4103C

Programmation Web orientée client
Le langage JavaScript

2016-2017

Organisation du module

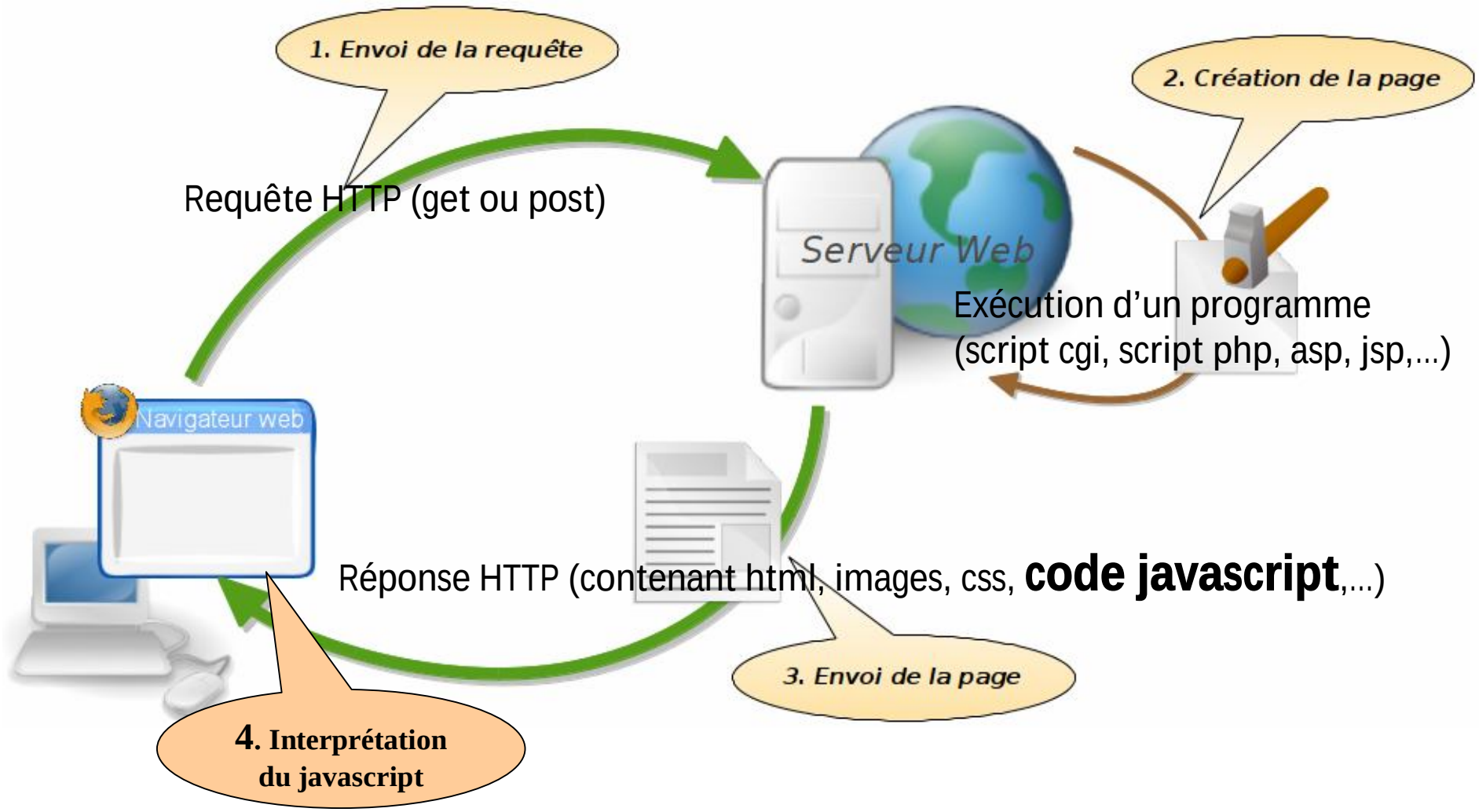
Numero semaine / [numero calendaire]		Cours	TD
semaine 1 [7]	JavaScript & DOM	CM 1.5h A	TD 2h SX
semaine 2 [9]			TD 2h SX TD 2h SX
semaine 3 [10]	Ajax & JQuery	CM 1.5h A	TD 2h SX
semaine 4 [11]			TD 2h SX TD 2h SX
semaine 5 [12]	Frameworks JavaScript	CM 1.5h A	TD 2h SX
semaine 6 [13]	projet		TD 2h SX TD 2h SX
semaine 7 [14]	projet		TD 2h SX TD 2h SX

Coefficient global : **15**

Contrôle machine : 10

Projet : 5

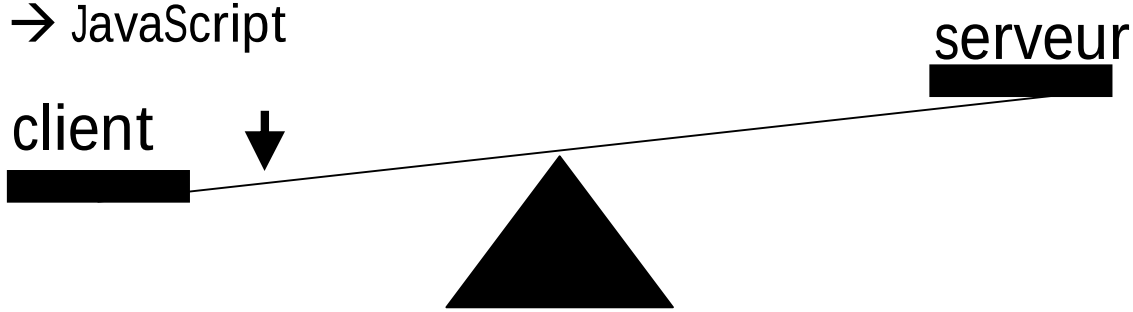
Environnement d'exécution d'une application WEB



La solution « Client riche »

• Quantité de traitement plus importante sur le client (on décharge donc le serveur d'une partie des traitements)

→ JavaScript



• Ne pas demander au serveur de produire une nouvelle page à chaque fois

→ récupérer des données auprès du serveur (au début ou lorsque nécessaire) (AJAX)

→ création/modification dynamique de la page web avec JavaScript (via le DOM)

• Une interface plus sophistiquée

→Librairies Javascript (JQuery, Dojo,...)

Démo

JavaScript - Généralités

- Le Javascript est un langage de programmation **interprété** (SpiderMonkey, JScript, V8)
- Le Javascript est un langage de programmation **orienté objet** (avec prototype)
- Le Javascript ne concerne **pas que de la programmation de Client Web** (même si…)

- **Pas de procédure d'écriture** sur disque (attention on est chez le client!)
- Sans rapport direct avec le **langage Java**
- Standardisé sous le nom **ECMAScript** (actuellement version 6 sortie en 2015)

http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf

- Lectures conseillées :
  - https://**developer.mozilla.org**/fr/docs/JavaScript/Reference
  - https://developer.mozilla.org/fr/docs/JavaScript/Guide
  - http://**fr.openclassrooms.com**/informatique/cours/
    - tout-sur-le-javascript
    - dynamisez-vos-sites-web-avec-javascript/introduction-au-javascript
  - http://**www.w3schools.com**/js/

JavaScript – petit historique

En **1995**, Brendan EichEn développe le LiveScript, un langage de script pour les serveurs de Netscape.

La même année une version pour client de ce langage appelé JavaScript est intégré dans le navigateur Netscape.

Les Boîtes de dialogues

Simple information

```
alert(‘Attention !’); // Affiche « Attention ! ».
```

Commentaires ou /* */

Saisie d’une valeur

```
var userName = prompt('Entrez votre prénom :');
alert(userName); // Affiche le prénom entré par l'utilisateur
```

Demande de confirmation

```
if (confirm('Voulez-vous supprimer ce fichier ?'))
{
        …
        alert('Le fichier a bien été supprimé !');
}
```

Insérer du JavaScript dans une page web

HelloWorld.html:

```
<!DOCTYPE html>
<html>
	<head>
		<title>Hello World!</title>
	</head>
	<body>
		<script> alert('Hello world!'); </script>
	</body>
</html>
```

Insérer du JavaScript dans une page web

HelloWorld.html:

```
<!DOCTYPE html>
<html>
	<head>
	<title>Hello World!</title>
	 </head>
	<body>
		<script src="hello.js"></script>
	</body>
</html>
```

hello.js:

```
alert('Hello world!');
```

Insérer du JavaScript dans une page web

- Insertion directe du code avec balise <script>

- Insertion par fichier externe préférable

- Plutôt à la fin du body que dans le head pour éviter de retarder le chargement de la page

```
<html>
<head>…
<body>
…
<script src="monscript.js"></script>
</body></html>
```

Quelques éléments de syntaxe – Variables et types

Déclaration de variables (≠PHP)

```
var myVariable1, myVariable2 = 4, myVariable3;
var myVariable1, myVariable2;
myVariable1 = myVariable2 = 2;
```

Typage dynamique (= PHP)

```
var number = 2; (number = flottant 64 bits)
alert(typeof number); // Affiche : « number »
var text = 'Mon texte'; //' ou " même effet
alert(typeof text);  // Affiche : « string »
var aBoolean = false;
alert(typeof aBoolean); // Affiche : « boolean »

number =‘test’;
alert(typeof number); // Affiche : « string »
```

Quelques éléments de syntaxe – Opérateurs

Opérateurs arithmétiques

```
+,-,*,/,%,+=,-=,++,--
number += 5; // équivalent à number = number+5;
var text = 'Bonjour ';
text += 'toi';
alert(text); // Affiche « Bonjour toi ».
```

Opérateurs de comparaison (=PHP)

```
==, !=,
=== contenu et type égal,
!== contenu ou type différent, >=,>,<,<=
var number = 4, text = '4', result;
result = number == text;alert(result); // Affiche  « true »
result = number === text;alert(result); //Affiche « false »
```

Opérateurs logiques

```
&&, ||, !
```

Structures de contrôle - Instructions conditionnelles

```
var floor = parseInt(prompt("Entrez l'étage :"));
if (floor == 0) {
    alert('Vous vous trouvez déjà au rez-de-chaussée.');
} else if (-2 <= floor && floor <= 30) {
    alert("Direction l'étage n°" + floor + ' !');
} else {
    alert("L'étage spécifié n'existe pas.");}
```

```
var rep = prompt('Choisissez 1 ou 2');
switch (rep) {
    case ‘1’:
        alert(‘reponse=1');
         break;
    case ‘2’:
       alert(‘reponse=2');
       break;
    default:alert(« ni 1, ni 2");}
```

Structures de contrôle - Boucles

```
var iter=1;
while (iter < 5)
{
        alert('Itération n° ' + iter);
        iter++;
}
//-----------------------------------------

var iter=1;
do {
        alert('Itération n° ' + iter);
        iter++;
}
while (iter<5);
```

Structures de contrôle - Boucles

```
for (var iter = 0; iter < 5; iter++)
{
	alert('Itération n°' + iter);
}


//----------------------------------------
var text="";
var i = 0;
for (; i < 10; )
{
	text += "carre de "+i+":"+i*i"\n";
	i++;
}
alert(text);
```

Les fonctions

```
var first=2,second=3; //variables globales
function maFonction1() {
	return first+second; //retour pas obligatoire
}
//-----------------------------------------------
function maFonction2() {
	var first=2,second=3; //variables locales (≠PHP)
	return first+second;
}
//-----------------------------------------------
// fonction avec parametres
function maFonction3(first, second) {
	return first+second;
}
Appel maFonction3(2,3);
//-----------------------------------------------
Paramètre résultat ? passer par un objet
Arguments facultatifs, fonctions anonymes, closure …
```

Les tableaux – création, accès, modification

```
var tab=['orange',3,'melon',8];//préféré
console.log(tab);
console.log(tab[0]+tab[2]);//Affiche orangemelon
console.log(tab[1]+tab[3]);// Affiche 11

var t=new Array(5,"Bonjour",290,1.414, "Toto",false,true);
t["txt"] = "Bienvenue à toi !";
t["est_majeur"] = true;
t["pi"] = 3.14;

var table = new Array("Pierre", "Paul", "Jacques");
alert(table.length);// Affiche 3
table[5] = "Toto";
alert(table.length);//Affiche 6

var table = new Array("Pierre", "Paul", "Jacques");
table[5] = "Toto";
table.sort(); //table=[Jacques,Paul,Pierre,Toto,undef,undef]
```

Les tableaux – Autres fonctions

```
// pour des nombres
table.sort(function (a, b) {
  if (a < b) {return -1;}
  else if (a > b) {return 1;}
  else {return 0;}
});

var t= new Array(1,6,8,3,12);
var r= t.pop(); // r contiendra 12
var r= t.shift(); // r contiendra 1
console.log(t);
t.push(5); // t=(6,8,3,5)
t.unshift(2);// t=(2,6,8,3,5)

var x=t.indexOf(8); //x=2
Aussi lastIndex, slice, concat,etc…
```

Les tableaux - Parcours

```
function affiche(tab)
{
	for(var i=0; i<tab.length; i++)
	{
		console.log(tab[i]);
	}
}

function affiche(tab)
{
	 for(var indice in tab)
	 {
		console.log(tab[indice]);
	 }
}
```

Les objets en JavaScript

```
Tout est objet : var s="test"; var i=s.length;
var car={type:'Fiat',model:500,color:'white'};//type objet
   //JSON (JavaScript Object Notation…)

Accès aux propriétés p.color ou p["color"]

Ajout dynamique de propriétés car.annee='2012'; (=PHP)

Parcours de l'objet :
For (var i in car){console.log(car[i]);}

Ajout de méthode
var car ={    type:'Fiat',model:500,color:'white',
              fulltype : function()
              { return this.type + " " + this.model;}
          }
OU car.fulltype=function(){return this.type+" "+this.model;}
console.log(car.fulltype());
```

Constructeur

```
function Person(nick,age,sex,work,friends)
{
    this.nick = nick; // en JS une fonction est un objet
    this.age  = age;
    this.sex  = sex;
    this.work = work;
    this.friends = friends;
}
// remarque : pas de classe !!
// On crée des variables qui vont contenir une instance de l'
objet Person :
var seb = new Person('Sébastien', 23, 'm', ‘Coiffeur', []);
var lau = new Person('Laurence', 19, 'f',  ‘dentiste', []);

alert(seb.nick); // Affiche : « Sébastien »
alert(lau.nick); // Affiche : « Laurence »
```

Exemples de code manipulant des objets


```
seb.nick = 'Bastien'; // On change le prénom
seb.age  = 18;        // On change l'âge

var myArray =
[
  new Person('Sébastien', 23, 'm',  'coiffeur', []),
  new Person('Laurence', 19, 'f',  'dentiste', []),
  new Person('Ludovic', 9, 'm',  'etudiant', []),
  new Person('Pauline', 16, 'f', 'etudiante', []),
  new Person('Guillaume', 16, 'm',  'dessinateur', []),
];

seb.friends.push(new Person('Johann', 19, 'm',  'DJ', []));
```

Définition de méthodes dans le constructeur


```
function Person(nick, age, sex, work, friends)
{
    this.nick = nick;
    this.age  = age;
    this.sex  = sex;
    this.work = work;
    this.friends = friends;

    this.addFriend =
    function(nick, age, sex, work, friends)
       {
              this.friends.push(new Person(nick,…));
       };
}

seb.addFriend('Johann', 19, 'm',  'DJ', []);
```

Prototype


```
function Person(nick, age, sex, work, friends)
{
    this.nick = nick;
    this.age  = age;
    this.sex  = sex;
    this.work = work;
    this.friends = friends;
}

Person.prototype.addFriend =
function(nick, age, sex, work, friends)
{
      this.friends.push(new Person(nick,…));
};

seb.addFriend('Johann', 19, 'm',  'DJ', []);
```

Héritage Prototypal

```
Objet Person
{
    nick :…
    …
    friends:…
    __proto__ : { addfriend }
   //Objet partagé par toutes les instances
}

function Etudiant(nick, age, sex, work, friends, domaine,
niveau)
{
 appel au constructeur Person…
 this.domain=domaine;
 this.niveau=niveau;
}
//Recopie du prototype de person dans celui d'étudiant
Etudiant.prototype = Object.create(Person.prototype);
```

Héritage Prototypal

```
Etudiant.prototype.change_domaine =
function(domaine)
{
       this.domaine=domaine;
};

Objet Etudiant    *recherche la méthode :
{                  1 - au niveau de l'instance,
    nick :…        2 - au niveau du prototype
       …           3 - au niveau du prototype du prototype
    friends:…         …
    domaine:…
    niveau:…
    __proto__ : { change_domaine }
        __proto__ : { addfriend }
}
```

Les objets prédéfinis

Array -> gestion des tableaux
String -> gestion des chaînes de caractères
Date   -> Accesseurs champs dates
RegExp ->Expressions régulières
Math un objet préexistant (pas de new nécessaire)
      Math.abs(),Math.cos(),Math.Max(),Math.random()…
…

Objets prédéfinis dans le contexte d'exécution d'une application web (Browser Object Model):
window (objet implicite alert, en fait window.alert())
window.**document** --> la page web
window.**navigator** (info sur navigateur), window.**history** **(**historique de navigation**),** window.**screen** (écran), window.**location** (URL),…

Gestion des évènements

Nom de l'événement	Action pour le déclencher
click	Cliquer (appuyer puis relâcher) sur l'élément
dblclick	Double-cliquer sur l'élément
mouseover	Faire entrer le curseur sur l'élément
mouseout	Faire sortir le curseur de l'élément
mousedown	Appuyer (sans relâcher) sur le bouton gauche ...
mousemove	Faire déplacer le curseur sur l'élément
keydown	Appuyer (sans relâcher) sur une touche de ...
keyup	Relâcher une touche de clavier sur l'élément
keypress	Frapper (appuyer puis relâcher) une touche de ...
focus	« Cibler » l'élément
blur	Annuler le « ciblage » de l'élément
change	Changer la valeur d'un élément de formulaire ...
select	Sélectionner le contenu d'un champ de texte ...
onload	La page a fini de se charger
...	

Association d’un évènement à un appel de fonction

```
<!DOCTYPE html>
<html>
<body>
<h1>My First JavaScript</h1>
<p>Click Date to display current day, date, and time.</p>
<button type="button" onclick="myFunction()">Date</button>
<p id="demo"></p>

<script>
function myFunction()
{
      alert(Date());
}
</script>

</body>
</html>
```

Association d’un évènement à un appel de fonction

```
Association sans passer par un attribut de balise html,
permet de définir plusieurs gestionnaires :

document.getElementById("myBtn").addEventListener
("click", //évènement
myFunction //fonction appelée);

document.getElementById("myBtn").addEventListener
("click", //évènement
myFunction2 //fonction appelée);


<script>
function myFunction() {alert(Date());}
function myFunction2() {alert(Date());}
</script>
```

Le DOM

- Le DOM (Document Object Model)

- Une API (Application Programming Interface)

- Prend la forme d'un objet (document) ayant des propriétés, et des méthodes permettant accéder (d'où le mot interface) à la page web.

- Le DOM permet d'accéder et de modifier dynamiquement une page web (en fait aussi du XML)

- Uniformisé par le W3C, pas spécifique au navigateur et pas spécifique au langage (DOM level 2 --> 2000, DOM level 3,4 --> 2004)
(voir http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/ecma-script-binding.html)

Exemples de fonctions d'accès et d'édition

```
<body>
  <a id="myLink" href="http://www.un_lien_quelconque.com">Un
 lien modifié dynamiquement</a>
  <script>
    var link = document.getElementById('myLink');
// aussi par tag ou par class
    var href = link.getAttribute('href'); // On récupère l'a
ttribut « href »
    alert(href);
    link.setAttribute('href', 'http://www.siteduzero.com');
 // on peut aussi écrire
 // link.href= 'http://www.siteduzero.com';
 // On édite l'attribut « href »
  </script>
</body>
```

document : un objet représentant la page

En rouge des fonctions d'accès et d'édition des éléments HTML

Exemples de fonctions d'accès et d'édition

```
<!DOCTYPE html>
<html>
<body>
<p>Hello World!</p>
<div id="main">
<p>The DOM is very useful.</p>
<p>This example demonstrates the <b>getElementsByTagName</b>
method</p>
</div>
<p id="demo"></p>
<script>
var x = document.getElementById("main");
var y = x.getElementsByTagName("p");
document.getElementById("demo").innerHTML =
'The first paragraph inside "main" is ' + y[0].innerHTML;
</script>
</body>
</html>
```

Modifier le style par l'API DOM

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

On peut aussi faire :
element.className = "mystyle"
```

innerHTML

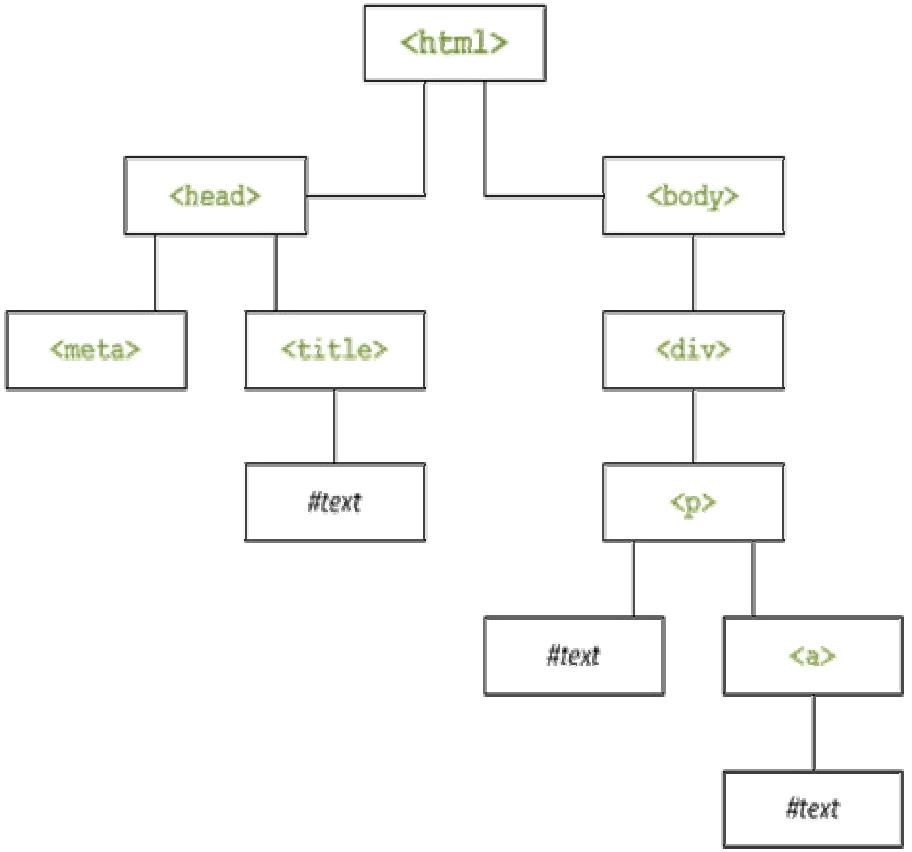
```
<body>
  <div id="myDiv">
    <p>Un peu de texte <a>et un lien</a></p>
  </div>
  <script>
    var div = document.getElementById('myDiv');
    alert(div.innerHTML);

      div.innerHTML = '<blockquote>Je mets une citation à l
a place du paragraphe</blockquote>';
      div.innerHTML+=' et <strong>une portion mise en empha
se</strong>.';

  </script>
</body>
```

innerHTML : le contenu de l'élément

Représentation arborescente d'une page Web

Exemple :

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Le titre de la page</title>
  </head>
  <body>
    <div id="mydiv">
      <p>Un peu de texte
         <a id="myLink">et un lien</a>
      </p>
    </div>
  </body>
</html>
```



Naviguer dans l'arborescence

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Le titre de la page</title>
  </head>
  <body>
    <div>
      <p id="myP"> Un peu de texte, <a>un lien</a> et
        <strong>une portion en emphase</strong></p>
    </div>
    <script>
      var paragraph = document.getElementById('myP');
      var first = paragraph.firstChild;
      var last  = paragraph.lastChild;
      alert(first.nodeName.toLowerCase());-->#text
      alert(last.nodeName.toLowerCase());-->strong
    </script>
  </body>
</html>
```

Naviguer dans l'arborescence

```
<body>
  <div>
    <p id="myP">Un peu de texte <a>et un lien</a></p>
  </div>
  <script>
    var paragraph = document.getElementById('myP');
    var children  = paragraph.childNodes;
    for (var i = 0, c = children.length; i < c; i++)
    {
      if (children[i].nodeType === 1)  // C'est un élément HTML
      {
        alert(children[i].firstChild.data);// et un lien       }
      else // C'est certainement un nœud textuel
      {
        alert(children[i].data); // un peu de texte
      }
    }
  </script>
</body>
element.nextSibling, element.parentNode,…
```

Création, insertion, suppression d'éléments, d'attributs

```
document.createElement('a');
document.write();
-----------------------
Element.appendChild()
Element.insertBefore()
Element.replaceChild()
Element.removeChild()
Element.removeAttribute()

…

Nombreuses méthodes à découvrir et à tester en TP…
```