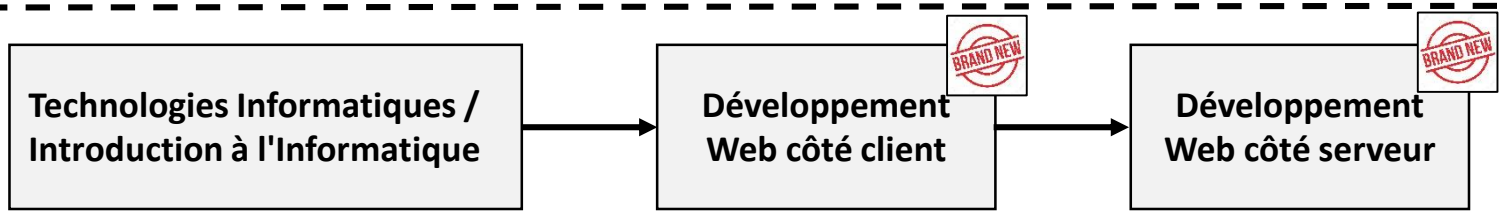
Université de Bretagne Occidentale

Développement web côté client

Cours 1 : HTML/CSS/JavaScript

Hai Nam TRAN
Université de Bretagne Occidentale – L2 INFO

Introduction



- **Cette UE contient 10 séances de 2h**
  - **CM : 3 séances (cours, tutoriels, démonstrations)**
  - **TD : 2 séances (exercices JavaSript)**
  - **TP : 5 séances (exercices HTML, CSS, JavaScript + Projet)**
- **Evaluation**
  - **Projet : 50%**
  - **Examen final : 50%**
- **Intervenant**
  - **Jean Philippe BABAU (jean-philippe.babau@univ-brest.fr)**
  - **Sarah MATTA (sarah.matta@univ-brest.fr)**
  - **Ikram CHOURIB (chourib.ikram@gmail.com)**

Introduction

- **Développement web**
  - ***"…une discipline qui consiste, par l'usage de langages de programmation web, à programmer des sites web ou applications web (ou web mobile) destinés à être publiés…"***
  - ***"… processus d'écriture d'un site ou d'une page web dans un langage technique…"***
  - ***"…. Le processus de développement web comprend, entre autres, la conception de sites web, le développement de contenu web, l'élaboration de scripts côté client ou côté serveur…"***
  - **La science de traitement et de présentation de l'information**

Introduction

- **Modèle client/serveur**
  - **Serveur de fichiers/d'exécutables**
    - Partage de données
    - Diffusion des exécutables des applications partagées
  - **Serveur de configurations**
    - Diffusion des paramètres de configuration partagés
  - **Serveur d'applications**
    - Serveur de calcul
    - Serveur web **WE ARE HERE !!!**
    - Software-as-a-service (SaaS, cloud)

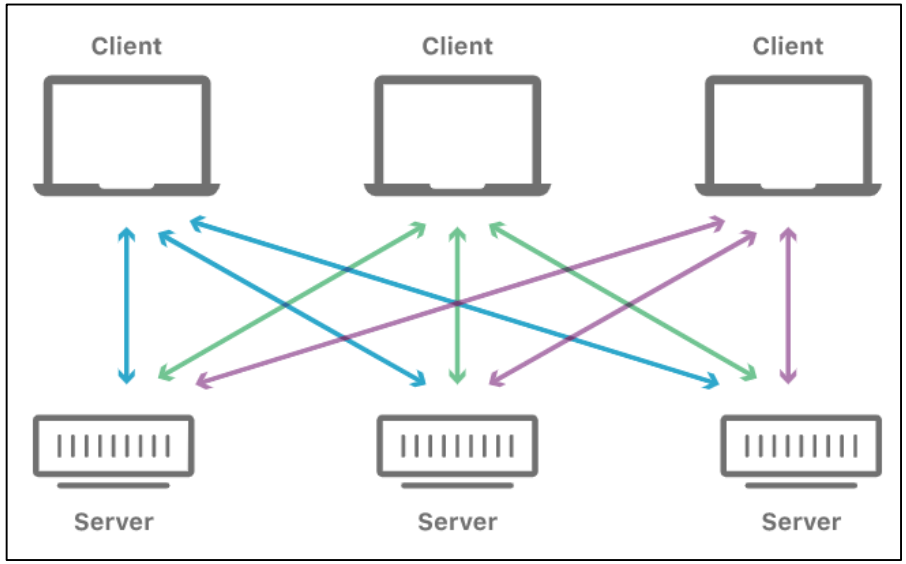
Introduction

- **Modèle client/serveur**
  - **Intérêts**
    - Installation et mise à jours des applications au niveau des serveurs
    - Cohérence entre postes: OS, configurations de base, logiciels
    - Sécurité
  - **Inconvénients et limites**
    - Performances
    - Risque d’indisponibilité générale (pannes serveur ou réseau)
    - Gestion des configurations spécifiques
    - Sécurité

Introduction

- **Côté client**

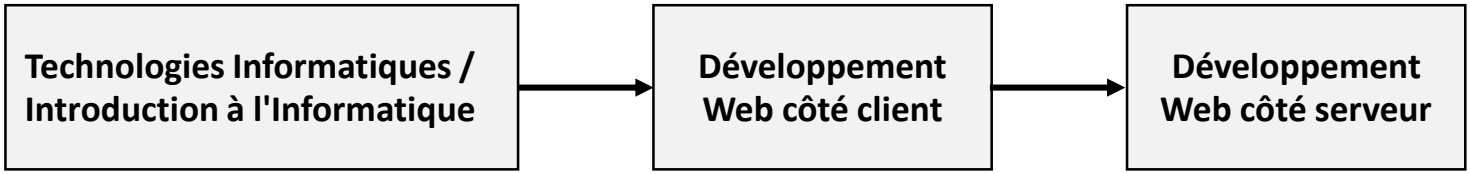
https://www.cloudflare.com/



  - **Tout ce qui s'affiche ou se déroule sur le client (appareil de l'utilisateur).**
    - Le contenu de la page - HTML
    - La mise en page – CSS
    - Le comportement - JavaScript

Introduction

- **Côté serveur**
  - **à voir plus tard…**



Projet

- **Cette année, un projet doit s'inscrire dans le cadre des deux thématiques ci-dessous**
  - **Cvdufutur : votre CV … mais, en 2032**
  - **Une association**

Projet

- **Dans le projet, les étudiants doivent démontrer leurs compétences en HTML, CSS et JS.**
  - **L'objectif est de développer par vous-même le site Web et d'explorer, d'apprendre et d'acquérir une meilleure compréhension du développement Web**
    - Tous les projets sont strictement vérifiés automatiquement par un vérificateur de plagiat et "manuellement" par les enseignants
    - Les étudiants doivent déposer chaque semaine les versions actuelles de leurs projets sur le Moodle.
    - Les étudiants doivent être capables d'expliquer toutes les parties de leurs projets lors de l'évaluation
    - Toute tentative de plagiat détectée entraînera une note 0 du projet (qui contribue à 50% dans la note finale de l'UE) et également les sanctions disciplinaires correspondantes

Evaluation

- **La note de projet sera donnée en prenant en compte les éléments ci-dessous :**
  - **Les devoirs/comptes rendus réalisés**
  - **Qualité de travail de l'ensemble d'exercices de TP**
  - **État d'avancement de projet après chaque séance de TP**
    - Il faut déposer avant la fin de chaque séance de TP tous les exercices + la version actuelle de votre projet
    - Toutes les absences doivent être justifiées
  - **Résultat finale de projet en fonction des 5 jalons**
    - HTML & CSS
    - JavaScript
    - Validation de données
    - Base de données
    - Bootstrap
  - **Présentation de projet**

Université de Bretagne Occidentale

HTML

Cours 1 : HTML/CSS/JavaScript

Hai Nam TRAN
Université de Bretagne Occidentale – L2 INFO

Structure minimale d'une page HTML5

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Ma première page</title>
  </head>

  <body>
    Contenu de ma première page.
  </body>
</html>
```

Les balises

- **HTML est un langage à balises.**
  - **Une balise est ce qui est entouré de chevrons < et >**
    - Les **conteneurs** : <x attribut="valeur">contenu</x>
      - <x> : balise ouvrante
      - </x> : balise fermante
    - Les **marqueurs** : <x attribut="valeur"**/**>
      - Le "/" n'est pas obligatoire mais permet de distinguer un marqueur d'une balise ouvrante
  - **Aux balises s'ajoutent des attributs qui les complètent en fournissant des informations supplémentaires**
    - <balise attribut="valeur">
    -

Les balises

- **Rappel**
  - <!DOCTYPE html>
  - <html></html>
  - <head></head>
  - <body></body>
  - <meta charset="utf-8" />
  - <title></title>
  -
  - <hr/>
  -
  - <h1></h1>
  - <h2></h2>
  -
  -
  - <mark> </mark>
  - <blockquote> </blockquote>
  -
  -
  -
  -
  - <table></table>
  - <tr><tr>
  - <td><td>
  - <a>
  - <img>

Devoir à rendre avant le 23/10
Q1 : donner un exemple de chaque balise HTML dans la liste

CSS

Cours 1 : HTML/CSS/JavaScript

Hai Nam TRAN
Université de Bretagne Occidentale – L2 INFO

CSS - Cascading Style Sheets

- **Permet de mettre en forme le code HTML**
  - **Choix de couleur de texte, sélection de la police de caractères, de la taille de la police, de l'image de fond...**
  - **Mise en page du site : menu à gauche en vertical, en-tête en haut toujours visible...**
- **Le CSS s'écrit au choix**
  - **Dans un fichier .css**
  - **Dans l'en-tête <head></head>**
  - **Dans les balises du HTML via l'attribut**

Structure d'un fichier CSS

- **Un fichier CSS contient des règles CSS**
  - **Les noms (sélecteurs) des balises dont on souhaite modifier l'aspect.**
  - **Des propriétés CSS**
  - **Une valeur pour chaque propriété**

```
sélecteur {
        propriété : valeur;
        propriété : valeur;
}
```

```
h1
{
    color: green;
    font-size: 20pt;
    font-weight: bold;
}
p
{
    color: red;
}
em, strong
{
     font-weight: bold;
}
```

Distinguer les balises : class et id

- **Exemple**
  - **On veut que seulement certains paragraphes soient mis sous une certaine forme**
  - **Plutôt que de mettre un attribut style à chacune des balises concernée on peut utiliser les attributs suivants valables sur n'importe quelle balise**
    - class
    - id

```
<h1 class="classe1"> </h1>
<p class="classe1"> </p>
```

```
.classe1
{
        color: red;
}
```

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

```
#ceParagraphe
{
        color: red;
}
```

Les propriétés CSS

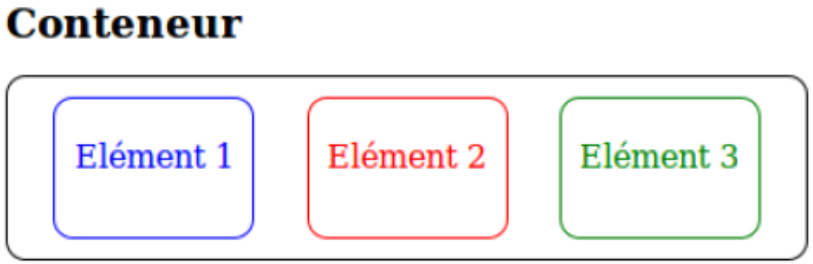
- **Rappel**
  - Taille du texte
  - Police
  - Mise en forme du texte
  - Alignement
  - Couleur du texte
  - Couleur de fond
  - Image de fond
  - La transparence
  - Les bordures
  - Changement d'apparence dynamique au survol, au click
  - Taille de bloc
  - Les marges

Devoir à rendre avant le 23/10 :
Q2: donner un exemple de chaque propriété CSS dans la liste

Flexbox

extrait de la partie Web – L1 Technologies Informatique

Avec Flexbox une page est vue comme un conteneur qui contient plusieurs éléments.
Dans une page Web on peut avoir plusieurs conteneurs contenant eux-mêmes d'autres conteneurs.



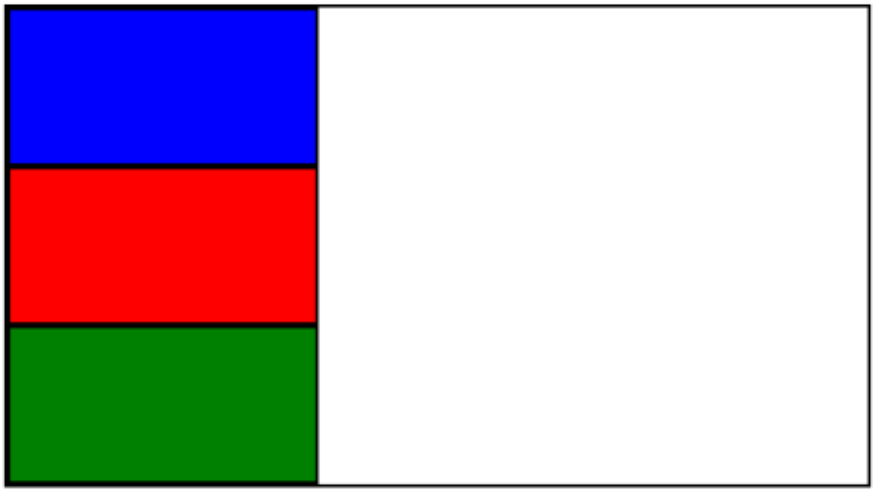
En HTML un conteneur sera une balise et chaque élément également (en général un `div`) :

```
<div id="conteneur">
<div id="element1"></div>
<div id="element2"></div>
<div id="element3"></div>
</div>
```

Flexbox

Par défaut les éléments se mettent les uns au-dessus des autres :

```
#conteneur
{
    border : 1px black solid;
}
```



Flexbox

extrait de la partie Web – L1 Technologies Informatique

Propriété `display : flex;`

```
#conteneur
{
    border : 1px black solid;

    display : flex;
}
```

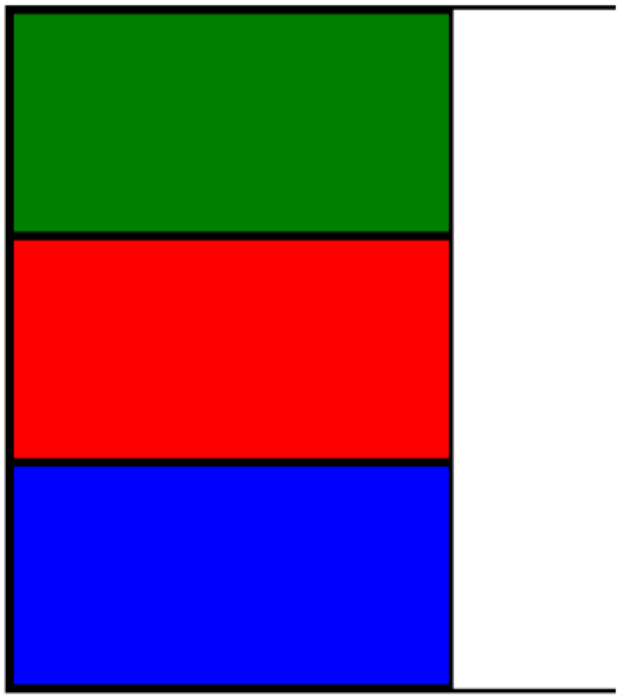
Les éléments se placent par défaut en ligne !

Flexbox

extrait de la partie Web – L1 Technologies Informatique

Propriété `flex-direction` qui peut prendre les valeurs :

- `row` en ligne (par défaut)
- `column` en colonne
- `row-reverse` en ligne inversés
- `column-reverse` en colonne renversés



`column-reverse`

On peut changer l'ordre des blocs sans modifier le HTML !

Flexbox

extrait de la partie Web – L1 Technologies Informatique

La propriété `flex-wrap` peut prendre les valeurs :

`no-wrap`

`wrap`

`wrap-reverse`

`no-wrap` a écrasé chaque bloc pour qu'ils puissent rentrer dans le conteneur.

Aligner sur l'axe principal

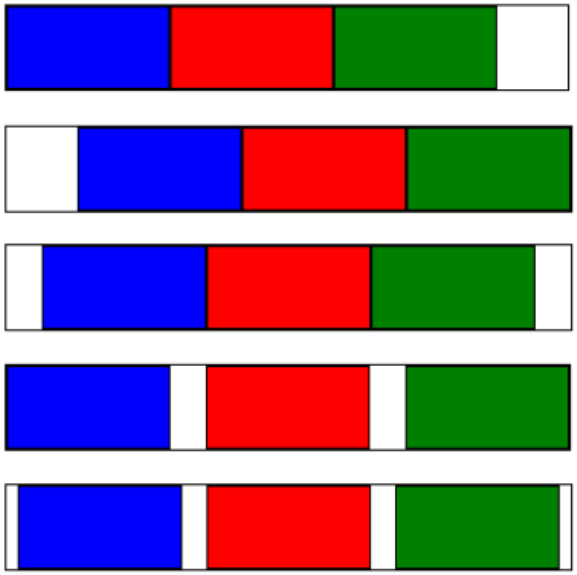
L'organisation des éléments avec Flexbox se déclare soit horizontalement (par défaut) soit verticalement.
Cette déclaration définit l'axe principal.
L'axe secondaire (cross axis) est alors l'autre axe perpendiculaire à l'axe principal. Il est donc :

- vertical si l'axe principal est horizontal
- horizontal si l'axe principal est vertical

Pour aligner sur l'axe principal, utiliser `justify-content` dont les valeurs possibles sont :

- `flex-start` : éléments alignés au début (par défaut)
- `flex-end` : éléments alignés à la fin
- `center` : éléments alignés au centre
- `space-between` : éléments répartis sur tout l'axe
- `space-around` : éléments répartis sur tout l'axe avec espace autour de chacun

Aligner sur l'axe principal



flex-start

flex-end

center

space-between

space-around

```
#conteneur
{
    border : 1px black solid;
    width : 350px;
    display : flex;
    flex-direction : raw;
    justify-content : space-around;
}
```

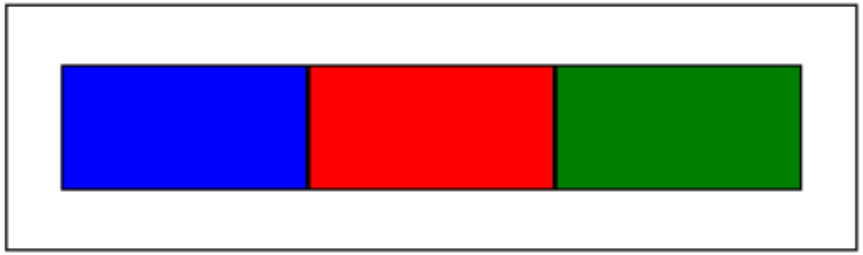
On peut évidemment le faire aussi verticalement :
`flex-direction : column;`

Aligner sur l'axe secondaire

On utilise pour cela la propriété `align-items` qui peut prendre les valeurs :

- `stretch` : les éléments s'étendent sur tout l'axe (par défaut)
- `flex-start` : alignés au début
- `flex-end` : alignés à la fin
- `center` : alignés au centre

```
#conteneur
{
    border : 1px black solid;
    width : 350px;
    height : 100px;
    display : flex;
    flex-direction : raw;
    justify-content : center;
    align-items : center;
}
```



Flexbox

Devoir à rendre avant le 23/10
Q3 : Résumer et expliquer la principe Flexbox dans > 0.5 page

Université de Bretagne Occidentale

JavaScript

Cours 1 : HTML/CSS/JavaScript

Hai Nam TRAN
Université de Bretagne Occidentale – L2 INFO

Références

1. https://developer.mozilla.org/en-US/
2. https://javascript.info/
3. https://www.w3schools.com/js/default.asp
4. https://www.fil.univ-lille.fr/~routier/enseignement/licence/tw1/spoc/#chap1

Introduction

- **Nous avons fait des pages statiques en L1: mise à part quelques effets (survol de souris, sélection ...) le fond et la forme sont prédéterminés et sont chargés par le client pour être affichés tels quels**
  - **Avec JavaScript on peut modifier l'apparence dynamiquement, par exemple changer la couleur de fond sur le survol d'un paragraphe particulier**
  - **JavaScript s'effectue côté client. Son code est du coup intégré dans un fichier du site et est interprété par le navigateur**
  - **JavaScript est un langage de programmation (au même titre que VB) qui n'a rien à voir avec Java**
  - **JavaScript peut être intégré directement dans un fichier HTML ou être écrit dans un fichier indépendant d'extension `.js`**

Introduction

- **JavaScript (JS)**
  - **A été initialement créé pour “rendre les pages web vivantes”**
    - Côté client, dans un navigateur web
  - **scripts : peuvent être écrits directement dans une page HTML et exécutés automatiquement au chargement des pages**
    - Pas besoin d'une compilation pour fonctionner
  - **Aujourd'hui**
    - Framework : Bootstrap, jquery, vue.js
    - côté serveur : node.js
    - tensorflow.js : machine learning, AI
    - …
- **Dans cette UE**
  - **Côté client, dans un navigateur web**
    - Référence : https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference (principal manuel avec des exemples et d’autres informations)
  - **Framework : Bootstrap**

Développement JS : éditeurs de texte

- **Visual Studio Code**
  - **A utiliser si vous avez pas d'un éditeur de texte préféré**
  - **Cross-platform : Windows, Linux, et macOS**
  - **Gratuit, leger**
- **Les autres**
  - **nano, vi, notepad++, sublime text, visual studio…**
  - **… utilisez à vos risques et périls ☺**

Ajouter du code JavaScript à la page

- **Les balises <script></script>**
  - **Peuvent être insérés dans n'importe quelle partie d'un document HTML**

```
<!DOCTYPE HTML>
<html>
<body>
    <p>Voici mon script</p>
    <script>
    alert( 'Hello, world!' );
    </script>
</body>
</html>
```

- **Scripts externes**
  - **Placer le code JS dans un fichier séparé**
  - **Le fichier de script est attaché à HTML avec l'attribut src**
    - <script src="../chemin/vers/votre/script1.js"></script>
    - <script src="../chemin/vers/votre/script2.js"></script>

Structure du code

- **Chaque instruction est généralement écrite sur une ligne distincte terminé par un point virgule**
  - Les points virgules peuvent être omis dans la plupart des cas mais **ne faites pas ça…**
- **Commentaires**
  - **sur une ligne : // (deux barres obliques)**
  - **multi lignes : /* … */**
    - commencé par **/***
    - terminée par ***/**

Les variables

- **Déclarer une variable**
  - **`let nom_de_variable`**
    - Déclaration de variable moderne
  - **`var nom_de_variable`**
    - similaire à let mais n'a pas de portée limitée aux blocs
    - mais la visibilité est étendue à la fonction ou globale (si la variable est déclarée hors fonction)
  - **`const nom_de_variable`**
    - Déclarer une constante
- **Nom de variable**
  - **Le nom ne doit contenir que des lettres, des chiffres, des symboles $ et _**
  - **Le premier caractère ne doit pas être un chiffre**
  - **La casse est importante**
  - **Les mots réservés (`let, class, return, function`) ne peuvent pas être utilisés**

Les types de données

number	let n = 123; n = 12.345;
bigint	let n = 12345678901234567890123456789012345678901234567890n
string	let str = "Hello";
boolean	let drapeau = true
null	let n = null;
undefined	let n;
object	à avoir plus tard
symbol	

Interaction

- **alert**
  - affiche un message

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

- **prompt**
  - affiche un message demandant à l'utilisateur de saisir du texte. Il renvoie le texte ou null, si vous cliquez sur le bouton Annuler/Esc

```
let n = prompt("Entrez un entier", 5);
```

- **confirm**
  - affiche un message et attend que l'utilisateur appuie sur "OK" ou "Annuler"
  - retourne true pour OK et false pour Annuler/Esc

```
let ok = confirm("OK ?");
```

Opérateurs

Addition	+
Soustraction	-
Multiplication	*
Division	/
Modulo	%
Exponentiation	**
Affectation	=
Incrémentation	++
Décrémentation	--
Binaire	AND(&), OR(|), XOR(^), NOT(~)
Comparaison	>, <, =>, <=, ==, ===, !=

Conditionnelle et boucles

```
if (condition) {
    //instructions
}
else {
    //instructions
}
```

```
let i = 2
if (i < 3) {
    alert(i);
}
```

```
while (condition) {
    //instructions
}
```

```
let i = 3;
while (i) {
    alert(i);
    i--;
}
```

```
do {
    //instructions
} while (condition)
```

```
for (début; condition; étape)
{
    //instructions
}
```

```
for (let i = 0; i < 3; i++) {
        alert(i);
}
```

Fonctions

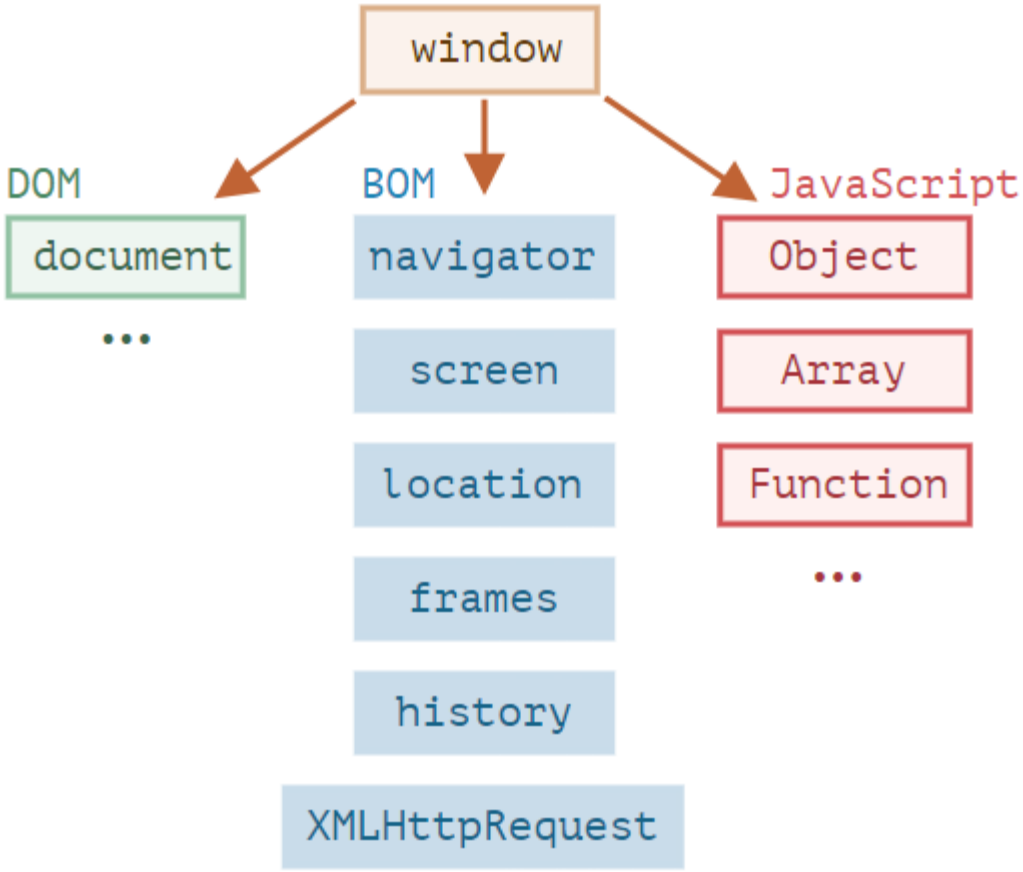
```
function name(param1, param2, ... paramN) {
    //instructions
}
```

```
function hello(name) {
    alert("Hello " + name + " !");
}

hello("Toto");
```

```
function sum(a, b) {
    return a + b;
}
```

DOM et BOM


window
DOM
document
...
BOM
navigator
screen
location
frames
history
XMLHttpRequest
JavaScript
Object
Array
Function
...

DOM et BOM

- **Document Object Model (DOM) représente tout le contenu de la page sous forme d'objets pouvant être modifiés.**
  - **L'objet document est le "point d'entrée" principal de la page**

```
document.body.style.background = "red";
```

```
let fname = document.getElementById("fname").value;
let lname = document.getElementById("lname").value;
alert("Hello " + fname + " " + lname);
```

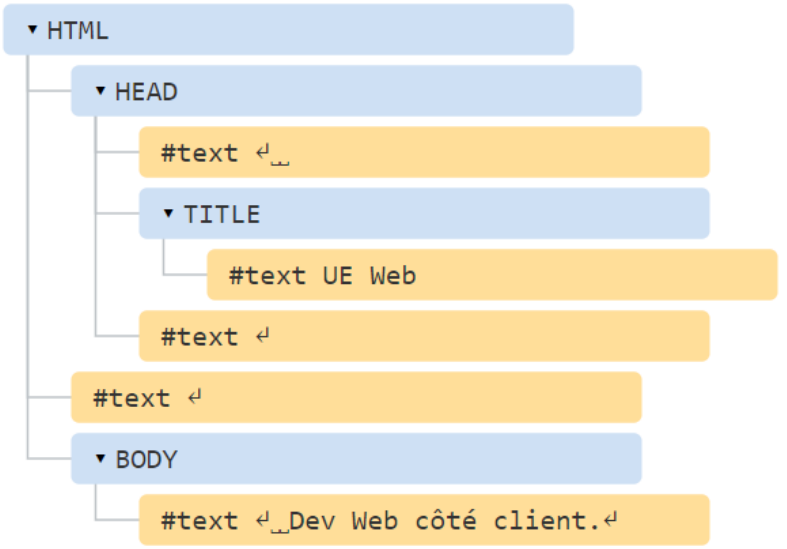
- **Browser Object Model (BOM) contient des objets supplémentaire fourni par le navigateur**
  - **navigateur, écran, location,…**

```
alert(location.href);
```

L'arbre DOM

- **Selon le modèle d'objets de document (DOM), chaque balise HTML est un objet.**
  - **Les balises imbriquées sont des "enfants" de celle qui les entoure**

```
<!DOCTYPE HTML>
<html>
<head>
        <title>UE Web</title>
</head>
<body>
        Dev Web côté cliente
</body>
</html>
```



  - **Étant donné un nœud DOM, nous pouvons aller vers ses voisins immédiats en utilisant les propriétés de navigation.**
    - `parentNode, childNodes, firstChild, lastChild, previousSibling, nextSibling`

Recherches: getElement*

- **getElementById()**
  - **Si un élément a l'attribut id, on peut atteindre cet élément en utilisant la méthode**

```
<input type="text" id="fname" value="Aabecede">

<script>
    let fname = document.getElementById("fname");
    alert("Hello " + fname.value);
</script>
```

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

<script>
    let divH = document.getElementById("divH");
    divH.style.background = "red";
</script>
```

Recherches: getElement*

- **getElementsByTagName(tag)**
  - **cherche les éléments avec la balise donné et renvoie l'ensemble de ces éléments**
- **getElementsByClassName(className)**
  - **renvoie les éléments qui ont la classe CSS donnée**
- **getElementsByName(name)**
  - **renvoie les éléments qui ont l'attribut name, dans tout le document.**

Propriétés et attributs de nœud

- **Différence ?**
  - **Les propriétés – sont ce qui se trouve dans les objets DOM.**
  - **Les attributs – sont ce qui est écrit en HTML.**

Propriétés	Attributs
nodeType nodeName/tagName innerHTML outerHTML value/data textContent hidden	… à revoir : balises HTML et leurs attributs

Modification du document

- **En manipulant le DOM avec JS, on peut :**
  - **Créer de nouveaux nœuds**
    - document.createElement(nom_de_balise) : crée un élément avec la balise donnée
  - **Insertion et suppression**
    - node.append(...nodes or strings) : insère dans node, à la fin
    - node.prepend(...nodes or strings) : insère dans node, au début
    - node.before(...nodes or strings) : insère juste avant node
    - node.after(...nodes or strings) : insère juste après node
    - node.replaceWith(...nodes or strings) : remplace node
    - node.remove() : supprime le node
  - **Modifier les propriétés et attributs de nœuds**
    - Note : utilisation principale dans cette UE

Styles et classes

- **Pour styler un élément :**
  -
  -
    - Avant, nous avons manipuler **class** et **style** avec un fichier **style.css**
- **JavaScript peut modifier les classes et les propriétés de style d'un élément**
  - **Class**
    - elem.className : la valeur de l'attribut class
    - elem.classList.add/remove("class") : ajoute ou enlève la classe
    - elem.classList.toggle("class") : ajoute la classe si elle n'existe pas, sinon enlève-la.
    - elem.classList.contains("class") : vérifie pour la classe donnée, renvoie true/false
  - **Style**
    - elem.style.*** : à revoir tous les propriétés CSS

Evènements de navigateur

- **Certaines actions sur des éléments d'un document web génèrent un événement**
  - **actions de l'utilisateur via le clavier ou la souris**
    - click, contextmenu, mouseover, mouseout, mousedown, mouseup, mousemove
    - keydown, keyup
  - **changement d'état**
    - change, focus, submit
  - **chargement d'un élément**
    - load
- **Programmation événementielle**
  - **Lier une fonction à l'occurrence d'un événement sur un élément**
    - On parle d'abonnement de la fonction à l'élément pour l'événement
    - Une fonction (**event handler/listener**) est déclenchée (exécutée) lorsque l'événement se produit sur cet élément cible (target)

Evènements de navigateur

- **Méthode d'abonnement**
  - **Attribut HTML : onclick**

```
<input value="Click me" onclick="alert('Click!')" type="button">
```

  - **Propriété DOM : onclick**

```
<input id="elem" type="button" value="Click me">
<script>
    elem.onclick = function() {
        alert('Thank you'); };
</script>
```

  - **addEventListener**
    - elem.addEventListener(event, handler, [options])
      - event : l'événement concerné – click, load, change, mouseover…
      - handler : la fonction qui je appelée lorsque l'événement se produit
      - options : à voir plus tard…

Que faire avec JavaScript?

- **Modifier contenu HTML**

```
<!DOCTYPE html>
<html>
<head>
    <script>
        function myFunction() {
            document.getElementById("demo").innerHTML = "Hello JavaScript!"
        }
    </script>
</head>
<body>
    <h2>What Can JavaScript Do?</h2>
    <p id="demo">JavaScript can change HTML content.</p>
    <button type="button" onclick='myFunction()'>Click Me!</button>
</body>
</html>
```

Que faire avec JavaScript?

- **Modifier contenu HTML**

Que faire avec JavaScript?

- **Modifier attributs HTML**

```
<!DOCTYPE html>
<html>
<head>
    <script>
        function turnOn() {
            document.getElementById('myImage').src='pic_bulbon.gif'
        }
        function turnOff() {
            document.getElementById('myImage').src='pic_bulboff.gif'
        }
    </script>
</head>
<body>
    <h2>What Can JavaScript Do?</h2>
    <p>JavaScript can change HTML attribute values.</p>
    <p>In this case JavaScript changes the value of the src (source) attribute of an image.</p>

    <button onclick="turnOn()">Turn on the light</button>
    <img id="myImage" src="pic_bulboff.gif" style="width:100px">
    <button onclick="turnOff()">Turn off the light</button>
</body>
</html>
```

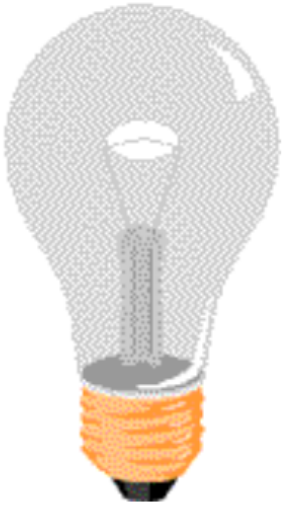
Que faire avec JavaScript?

- **Modifier attributs HTML**

What Can JavaScript Do?

JavaScript can change HTML attribute values.

In this case JavaScript changes the value of the src (source) attribute of an image.



Turn on the light Turn off the light

Que faire avec JavaScript?

- **Modifier CSS**

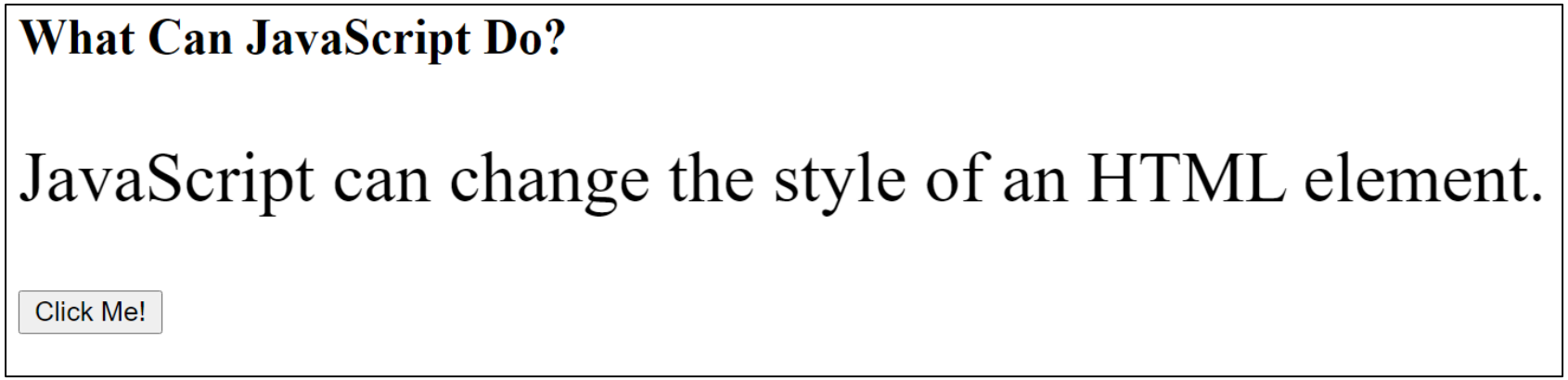
```
<!DOCTYPE html>
<html>
<head>
    <script>
        function changeSize() {
            document.getElementById('demo').style.fontSize='35px'
        }
    </script>
</head>
<body>
    <h2>What Can JavaScript Do?</h2>

    <p id="demo">JavaScript can change the style of an HTML element.</p>

    <button type="button" onclick="changeSize()">Click Me!</button>
</body>
</html>
```

Que faire avec JavaScript?

- **Modifier CSS**



Que faire avec JavaScript?

- **Afficher/cacher éléments HTML**

```html
<!DOCTYPE html>
<html>
<head>
    <script>
        function hideAndShow() {
            var p = document.getElementById('demo');
            var s = p.style.display;
            if(s == 'none') {
                p.style.display = 'block'
            } else {
                p.style.display = 'none'
            }
        }
    </script>
</head>
<body>
    <h2>What Can JavaScript Do?</h2>
    <p id="demo">JavaScript can hide HTML elements.</p>
    <button type="button" onclick="hideAndShow()">Click Me!</button>
</body>
</html>
```

Université de Bretagne Occidentale

Structure d'un site web

Cours 1 : HTML/CSS/JavaScript

Hai Nam TRAN
Université de Bretagne Occidentale – L2 INFO

Structure d’un site web

- **Structure ?**
  - **Comment organiser des resources : images, audios, videos, css, javascript ?**
  - **Comment organiser des pages : index.html, contact.html ?**
  - **Comment connecter les ressources et les pages : hyperlink, balises (img, video, ...) ?**

Une organisation générale

- **web**
  - **media**
    - img
    - video
    - audio
  - **css**
    - style.css
  - **script**
    - script.js
  - **page**
    - contact.html
    - event.html
  - **index.html**