



Javascript

Introduction



Introduction Javascript

+ HTML

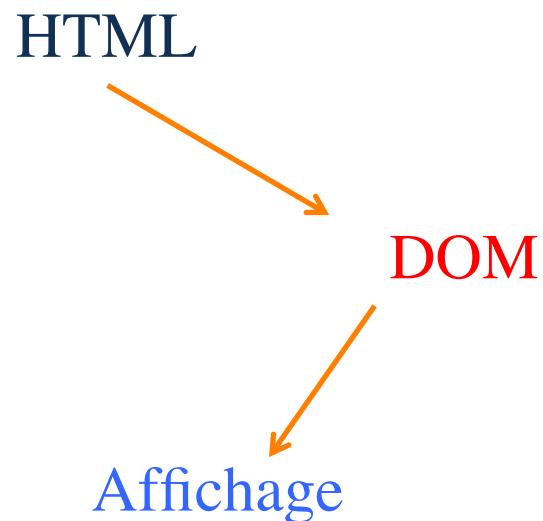
```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
  </head>
  <body>
    <p>Document HTML</p>
  </body>
</html>
```

Introduction Javascript

- + Comment modifier dynamiquement le contenu d'une page HTML ?
 - + Sur la base d'une action de l'utilisateur
 - + Parce que des informations dans la page ont changé
 - + ...

Introduction Javascript

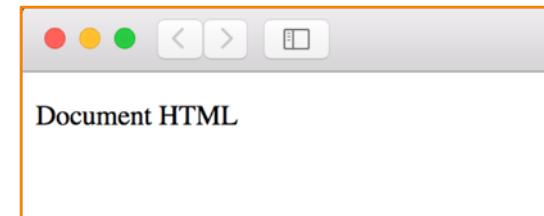
- + Comportement du navigateur web



Introduction Javascript

+ HTML

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
  </head>
  <body>
    <p>Document HTML</p>
  </body>
</html>
```



The screenshot shows a browser's developer tools DOM inspector. The left panel displays the DOM tree:

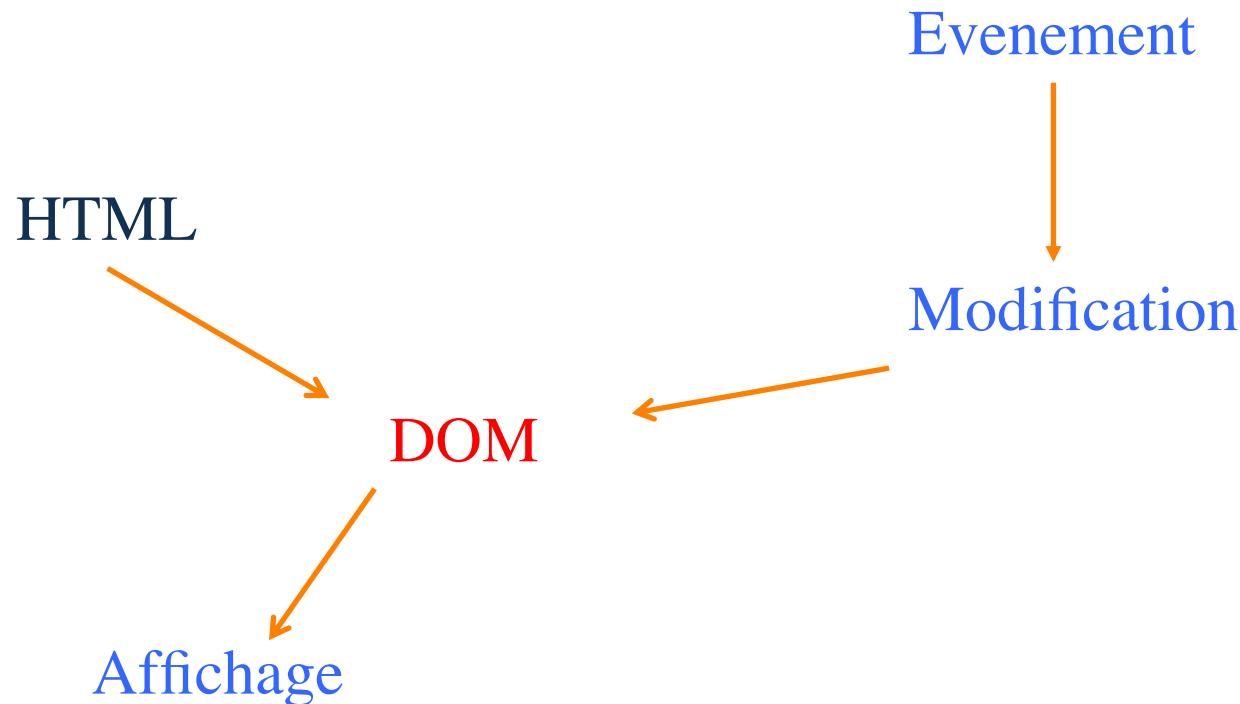
```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8">
  </head>
  <body>
    <p>Document HTML</p> = $0
  </body>
</html>
```

The `<p>Document HTML</p>` node is selected, highlighted with a gray background. The right panel, titled "Propriétés", lists the properties of the selected `p` element:

- `accessKey: ""`
- `align: ""`
- `assignedSlot: null`
- `attributes: NamedNodeMap {length: 0}`
- `baseURI: "file:///Users/kwy"`
- `childElementCount: 0`
- `childNodes: NodeList [#text]`
- `children: HTMLCollection []`
- `classList: DOMTokenList {length: 0}`
- `className: ""`

Introduction Javascript

+ Comportement du navigateur web



Introduction Javascript

- + Javascript
 - + Langage de programmation
 - + Exécuté par le navigateur
 - + Permettant de modifier le contenu du DOM, et donc de l'affichage
 - + Permettant de communiquer avec le serveur (AJAX)

Introduction Javascript

- + Dans Javascript
 - + Elements de langages de programmation
 - + Variables
 - + Fonctions
 - + Instructions de contrôle
 - + Tests
 - + Boucles
 - + Fonctions permettant de manipuler le DOM
 - + Fonctions permettant d'interagir avec l'utilisateur
 - + Fonctions permettant de communiquer
 - + Interception d'evenements (trigger / déclencheur)

Elements de syntaxe Javascript



Elements de syntaxe Javascript

+ L'élément de référence : **document**

document permet

- + D'accéder aux informations contenues dans la page
- + D'accéder aux informations sur le navigateur
- + D'accéder aux fonctions permettant de manipuler le contenu du document

Elements de syntaxe Javascript

- + Afficher un message dans une page html :

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
  </head>
  <body>
    <p>Document HTML</p>
    <script type="text/javascript">
      <!--
        //masquage du script pour les anciens navigateurs
        document.writeln("Nous sommes le 20/01/2019");
      -->
    </script>
  </body>
</html>
```



Elements de syntaxe Javascript

- + Qu'est-ce qui s'est passé ?
 - + Le navigateur a analysé le contenu de la page
 - + Il a affiché le contenu de la balise p
 - + Il a découvert une balise **script**, de type javascript
Il en a exécuté le contenu
 - + Il a affiché, à l'endroit où il se trouve, les informations demandées : nous sommes le 20/01/2019

Elements de syntaxe Javascript

- + Et si je dois faire la même chose à plusieurs endroits ?
 - + Définir une **fonction**

```
function maFonction() {  
    Ce que doit faire la fonction  
}
```
 - + La fonction est définie au choix :
 - + Dans la partie entête (head) de la page
=> utilisation uniquement dans cette page
 - + Dans un fichier à part qui va être inclus dans la page : fichier javascript avec une extension js
=> permet une utilisation là où on le souhaite

Elements de syntaxe Javascript

+ Fonction déclarée dans la page

```
<!DOCTYPE html>
<html lang="fr-fr">
    <head>
        <title>Document HTML</title>
        <meta charset="UTF-8"/>
        <script type="text/javascript"> <!--
            function afficherDate() {
                document.writeln("Nous sommes le 20/01/2019");
            }
        --> </script>
    </head>
    <body>
        <p>Document HTML</p>
        <script type="text/javascript"> <!--
            afficherDate();
        --> </script>
    </body>
</html>
```



Elements de syntaxe Javascript

+ Fonction déclarée dans un fichier distinct

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
    <script type="text/javascript" src="test.js" ></script>
  </head>
  <body>
    <p>Document HTML</p>
    <script type="text/javascript">
      <!--
        afficherDate();
      -->
    </script>
  </body>
</html>
```



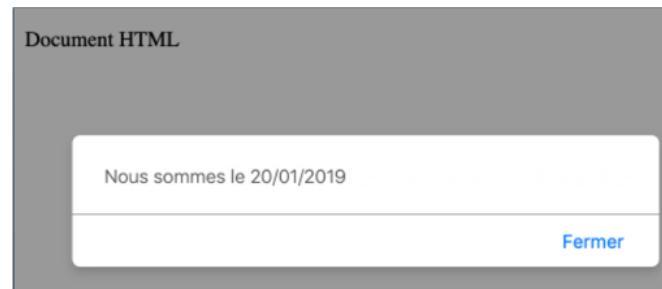
```
function afficherDate() {
  document.writeln("Nous sommes le 20/01/2019");
}
```

test.js

Elements de syntaxe Javascript

- + Afficher un message d'alerte dans une fenêtre séparée :
alert("message");

```
function afficherDate() {  
    alert("Nous sommes le 20/01/2019");  
}
```



Elements de syntaxe Javascript

+ Les variables en Javascript

- + Une variable porte un nom = suite de caractères sans espaces, et si possible sans accents, commençant par une lettre.
- + Typage dynamique (en fonction du contexte)

+ Définir une variable

```
var maVariable;
```

+ Mettre une valeur dans une variable

```
var maVariable = 5;
```

```
var mavariable2 = maVariable + 1
```

```
var maVariable3 = "test = "+ maVariable2;
```

Elements de syntaxe Javascript

- + Opérateur
 - + Utilisation des opérateurs classiques
 - + Arithmétique : + - * /
 - + Compareurs : == != < <= > >=
 - + Comparateur + test de typage : === !==
 - + Booléen : && || !

Elements de syntaxe Javascript

+ Instructions

+ Terminées par ;

+ Commentaires :

/* ... */

//

+ Tests :

if (cond) {instructions_si_vrai;}

if (cond) {instructions_si_vrai;} else {instructions_si_faux;}

switch (element) case ... :... break;... default: ... }

+ Boucles

for (init; condition; incrément) { ... }

while (condition) { ... }

Elements de syntaxe Javascript

+ Fonctions

+ Création

```
function nomFonction1(arguments) {
```

```
    ...
```

```
}
```

```
function nomFonction2(arguments) {
```

```
    ...
```

```
    return value;
```

```
}
```

+ Appel

```
nomFonction1(paramètres);
```

```
var maVariable = nomFonction2(paramètres);
```

Elements de syntaxe Javascript

+ Objets

Object, Array, Date, ...

+ Création = opérateur new

+ Appel de méthode : objet.methode(paramètres)

+ this = objet manipulé

```
function afficherDate() {  
    var date = new Date();  
    alert("Nous sommes le " + date.getDate() + "/" + (date.getMonth() + 1) + "/" + date.getFullYear());  
}
```

Elements de syntaxe Javascript

- + Créer ses propres objets

- + A partir d'Object

```
var pierre = new Object();
pierre.nom = "Durand";
pierre.prenom = "Pierre";
```

- + A partir d'un bloc
= JSON

```
var pierre = {
    nom : "Durand",
    prenom : "Pierre",
    nomprenom : function() {
        return this.nom+" "+this.prenom;
    }
}
```

- + A partir d'une fonction

```
function personne(n, p) {
    this.nom = n;
    this.prenom = p;
}
var pierre = new personne( "Durand", "Pierre");
```

Elements de syntaxe Javascript

+ Interagir avec la page :

- + Pour l'instant, on n'a pas gagné grand chose : peu d'interaction avec l'utilisateur.
- + Comment changer le comportement de la page en fonction des actions de l'utilisateur ?
Utilisation de la programmation évènementielle
= réagir en fonction des évènements qui surviennent.

Utilisation de **déclencheurs** = demander à un élément de la page de réagir en fonction de ce qui se passe

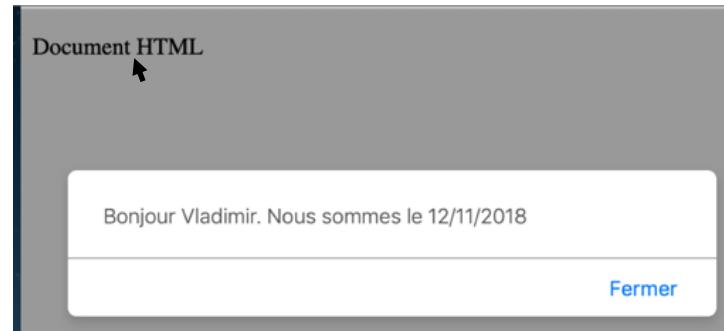
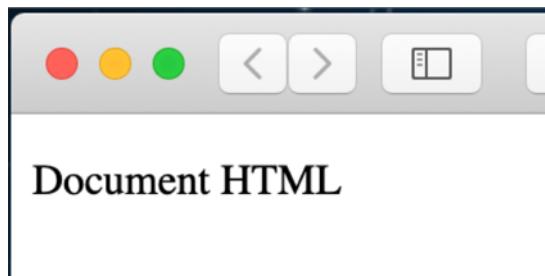
Elements de syntaxe Javascript

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
    <script type="text/javascript" src="test.js" ></script>
  </head>
  <body>
    <p onmouseover="afficherDate('Vladimir', new Date());">Document HTML</p>
  </body>
</html>
```

onmouseover = lorsque la souris passe au dessus de l'élément

```
function afficherDate(nom, date) {
  alert("Bonjour " + nom + ". Nous sommes le " + date.getDate() + "/" + (date.getMonth() + 1) + "/" + date.getFullYear());
}
```

Elements de syntaxe Javascript



La souris passe au dessus du texte

Elements de syntaxe Javascript

- + Les déclencheurs
 - + Réagissent sur la base d'évènements
 - + Souris (clic, déplacement, ...)
 - + Clavier
 - + ...
 - + Dépendent des éléments sur lesquels ils sont placés
 - + Certains ne sont disponibles QUE en HTML5

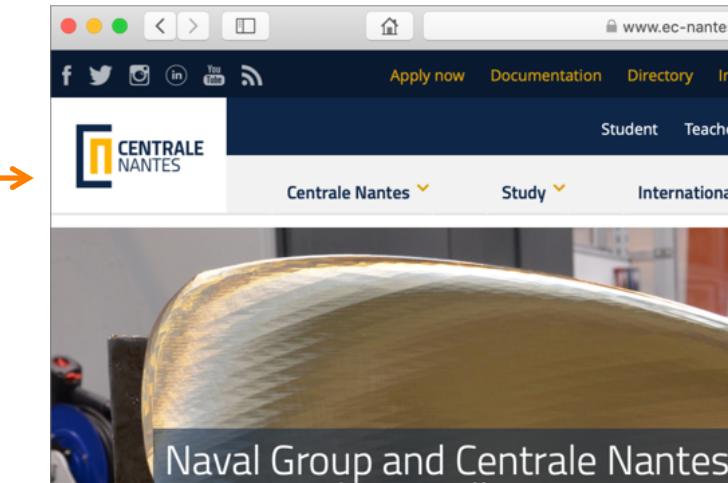
Elements de syntaxe Javascript

- + Quelques déclencheurs :
 - + onClick : clic sur le bouton de la souris
 - + onDoubleClick : double-clic sur le bouton de la souris
 - + onMouseDown : bouton de la souris enfoncé sur un élément input
 - + onMouseUp : bouton de la souris relâchée sur un élément input
 - + onMouseOver : curseur survolant l'objet
 - + onMouseOut : curseur quittant le survol de l'objet
 - + onKeyDown : touche du clavier enfoncée
 - + onKeyPress : la touche du clavier enfoncée est relâchée
 - + onKeyUp : touche du clavier relâchée
 - + onLoad : l'élément est chargé dans la page (ouverture du document)
 - + onSelect : du texte a été sélectionné dans l'élément
 - + onSubmit : Soumission d'un formulaire
 - + onChange : l'élément a été modifié (input, select, ...)
 - + ...

Elements de syntaxe Javascript

- + Fonctionnement des déclencheurs :
 - + Utilisé comme un attribut dans la balise
 - + La valeur de l'attribut déclencheur est la suite d'instructions à exécuter => il peut s'avérer utile d'utiliser des fonctions
 - + Les déclencheurs **retournent** une valeur (true ou false) indiquant si l'événement doit être traité ou non. Par défaut, on considère que true est retourné.
 - + Si true est retourné, l'événement est considéré comme non traité et le traitement par défaut s'applique
 - + Si false est retourné, l'événement est considéré comme traité et le traitement par défaut ne s'applique pas.

Elements de syntaxe Javascript



```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
  </head>
  <body>
    <p><a href="http://www.ec-nantes.fr" onclick="return true;">Document HTML</a></p>
  </body>
</html>
```

Elements de syntaxe Javascript



```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
  </head>
  <body>
    <p><a href="http://www.ec-nantes.fr" onclick="return false;">Document HTML</a></p>
  </body>
</html>
```

Elements de syntaxe Javascript

+ Et avec les fonctions ?

```
function retournerValeur() {  
    return true;  
}
```

```
<!DOCTYPE html>  
<html lang="fr-fr">  
    <head>  
        <title>Document HTML</title>  
        <meta charset="UTF-8"/>  
        <script type="text/javascript" src="test.js" ></script>  
    </head>  
    <body>  
        <p><a href="http://www.ec-nantes.fr" onclick="return retournerValeur();">Document HTML</a></p>  
    </body>  
</html>
```

Elements de syntaxe Javascript

- + Et si on modifiait les éléments du DOM ?
 - + **document** : l'objet global qui est à la racine du DOM.
Tout est dans **document**.
 - + **this** : référence à l'objet dans lequel est le déclencheur

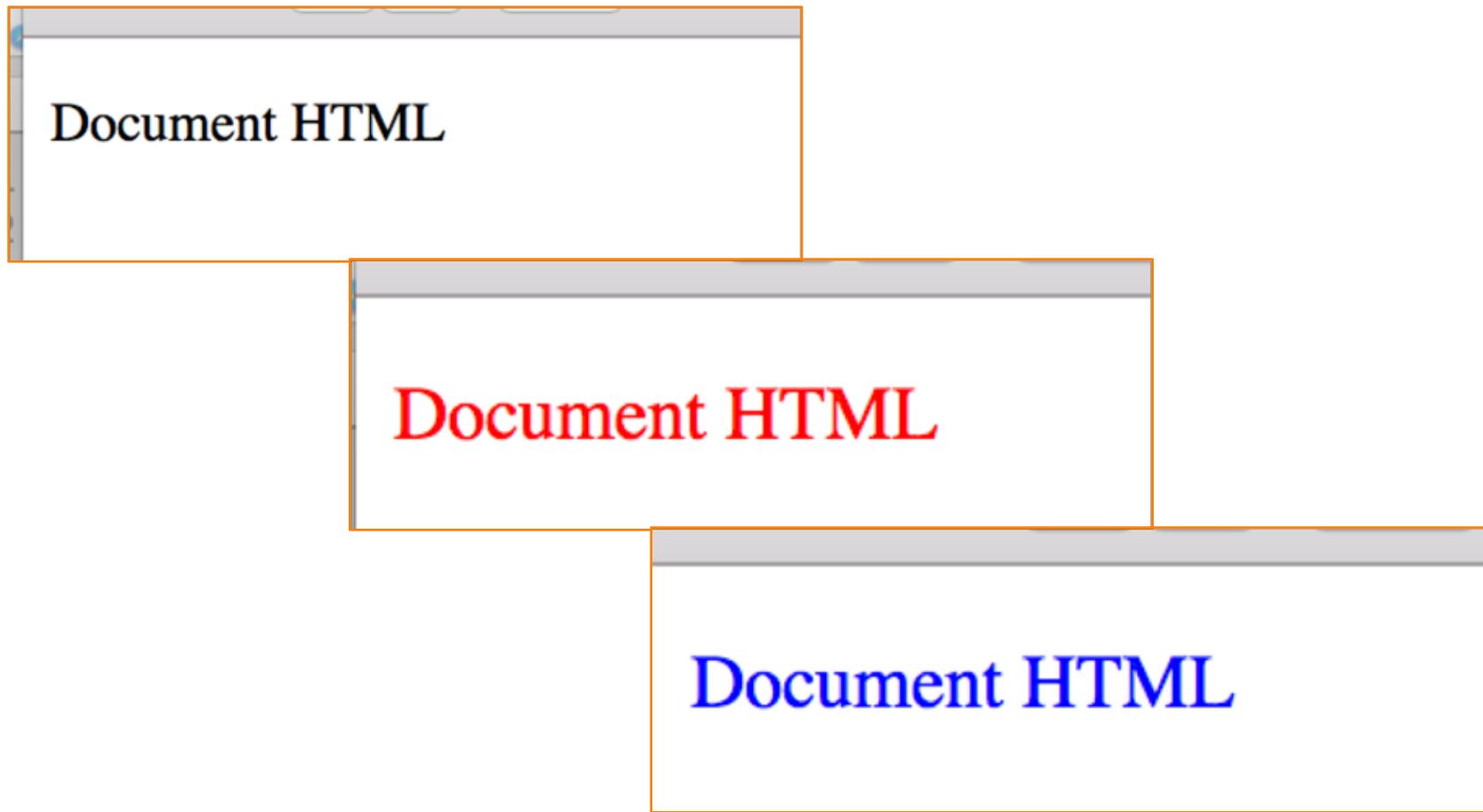
Elements de syntaxe Javascript

```
function afficher(element, couleur) {  
    element.style.color = couleur;  
}
```

```
<!DOCTYPE html>  
<html lang="fr-fr">  
    <head>  
        <title>Document HTML</title>  
        <meta charset="UTF-8"/>  
        <script type="text/javascript" src="test.js" ></script>  
    </head>  
    <body>  
        <p onmouseover="afficher(this, 'red');" onmouseout="afficher(this, 'blue');>Document  
HTML</p>  
    </body>  
</html>
```



Elements de syntaxe Javascript



Elements de syntaxe Javascript

- + Que peut-on utiliser / modifier dans les éléments ?
Pratiquement tous les attributs
 - + class = l'attribut class de l'élément
 - + classList = liste des classes de l'élément
 - + style = le style qui lui est appliqué
 - + id = l'attribut id de l'élément
 - + textContent : le contenu d'une balise de texte
 - + value : la valeur contenue dans un input, un select, ...
 - + selectedIndex= l'index sélectionné dans un select
 - + tagName = le nom de la balise
 - + ...

Elements de syntaxe Javascript

- + Que peut-on utiliser / modifier dans les éléments ?
 - + style = l'attribut style de l'élément
 - + style.color = couleur du texte
 - + style.backgroundColor : couleur de fond
 - + ...
 - + display : la façon dont l'élément est affiché (gestion des bordures)
 - + top, left, right, bottom, width, height, ...

Elements de syntaxe Javascript

- + Accéder / Manipuler les éléments de la page ?
 - + id : élément unique dans la page
`document.getElementById(idDeElement)`
l'objet correspondant à l'id en question
 - + La balise
`document.getElementsByTagName (nomBalise)`
-> tous les objets correspondant à la balise mentionnée
 - + firstChild, lastChild
Le premier / dernier élément fils d'un objet
 - + nextSibling, previousSibling
L'élément de même niveau situé après, avant
 - + ...
 - + null = pas d'objet correspondant

Elements de syntaxe Javascript

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
    <script type="text/javascript" src="test.js" ></script>
  </head>
  <body>
    <p id="montexte">Document HTML</p>
    <p onmouseover="changeTexte('je change');" onmouseout="changeTexte('je reviens');" >je
change tout</p>
  </body>
</html>
```

```
function changeTexte(leTexte) {
  var element = document.getElementById("montexte");
  element.textContent = leTexte;
}
```

Elements de syntaxe Javascript

Document HTML

je change tout

je change

je change tout

je reviens

je change tout

Elements de syntaxe Javascript

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
    <script type="text/javascript" src="test.js" ></script>
  </head>
  <body>
    <p id="montexte">Document HTML</p>
    <p onmouseover="changeTexte(1);" onmouseout="changeTexte(2);>je change tout</p>
  </body>
</html>
```

```
function changeTexte(valeur) {
  var element = document.getElementById("montexte");
  if (valeur == 1) {
    element.textContent = "Je change";
  } else {
    element.textContent = "Je reviens";
  }
}
```

Elements de syntaxe Javascript

Document HTML

je change tout

je change

je change tout

je reviens

je change tout

Elements de syntaxe Javascript

+ Modifier le DOM pour associer les déclencheur

Comment sont définis les déclencheurs dans le DOM ?

The screenshot shows the DOM tree on the left and the properties panel on the right. In the DOM tree, a `<p>` element is selected. Two blue arrows point from the highlighted code in the DOM tree to the corresponding event handler properties in the properties panel.

DOM Tree:

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>...</head>
  <body>
    <p id="monTexte">Document HTML</p>
    <p onmouseout="changeTexte(2); onmouseover="changeTexte(1);>je change tout</p> = $0
  </body>
</html>
```

Properties Panel:

- onmouseout: function(event)
- onmouseover: function(event)
- arguments: null
- caller: null
- length: 1
- name: "onmouseover"
- prototype: onmouseover {}
- Prototype Function
- onmouseup: null

Elements de syntaxe Javascript

+ Et si on modifiait les fonctions dans le DOM...

```
<!DOCTYPE html>
<html lang="fr-fr">
  <head>
    <title>Document HTML</title>
    <meta charset="UTF-8"/>
    <script type="text/javascript" src="test.js" ></script>
  </head>
  <body>
    <p id="montexte">Document HTML</p>
    <p id="declencheur">je change tout</p>
    <script>
      <!--
      onloadHandler();
      -->
    </script>
  </body>
</html>
```

```
function changeTexte(valeur) {
  var element = document.getElementById("montexte");
  if (valeur == 1) {
    element.textContent = "Je change";
  } else {
    element.textContent = "Je reviens";
  }
}

function declencheurOver(event) {
  changeTexte(1);
}
function declencheurOut(event) {
  changeTexte(2);
}

function onloadHandler() {
  var pDeclenche = document.getElementById("declencheur");
  pDeclenche.onmouseover = declencheurOver;
  pDeclenche.onmouseout = declencheurOut;
}
```

Elements de syntaxe Javascript

Document HTML

je change tout

je change

je change tout

je reviens

je change tout

Elements de syntaxe Javascript

- + On pousse un peu les modifications du DOM
 - + Créer un nouvel élément:
`var elt = document.createElement(nomBalise);`
 - + Créer un nouvel élément texte :
`var textContent = document.createTextNode(text);`
 - + Ajouter un élément à la liste des fils d'un nœud :
`unNoeud.appendChild(nouveauFils);`
 - + Ajouter un nœud avant l'un des nœuds d'un élément
`unNoeud.insertBefore(nouveauFils, avantFils);`
 - + Retirer un nœud de la liste des fils
`unNoeud.removeChild(ancienFils);`

JSON



JSON

- + JavaScript Object Notation
Format de données spécifique à Javascript

Utilisé pour mémoriser des informations et permettre à javascript de les manipuler plus facilement.

JSON

- + Quelques éléments de syntaxe :

- + { ... } : objet
- + Champ:valeur
- + [...] : tableau

- + Exemples :

- + { nom : "valeur", prenom : "valeur" }
- + { personne: { nom : "valeur", prenom : "valeur" } }
- + ["Bleu", "Rouge", "Vert"]

JSON

+ Quelques fonctions à connaître

```
// Crédation
myObj = {name: "John", age: 31, city: "New York"};
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Restauration
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
var myName = obj.name;
```

Et maintenant ?



Et maintenant ?

- + Javascript => pratique
- + Tout un tas d'outils pour vous faciliter la vie
 - + Jquery
 - + AngularJS
 - + ...

Mais vous ne pourrez les maîtriser qu'avec un peu de pratique

Et maintenant ?

- + Qu'est-ce qu'on va pouvoir faire avec Javascript ?
 - + Changer le comportement de la page
 - + Changer le comportement des éléments des formulaires
 - + Communiquer avec le serveur (AJAX)
 - + Changer l'apparence de la page en fonction de ce que fait l'utilisateur est des informations dans la base de données
 - + Complétion automatique
 - + ...

