Conception de sites web dynamique

HTML – CSS – JAVASCRIPT – PHP
BASE DE DONNÉES

IF04U050

David Bouchet

david.bouchet.paris5@gmail.com

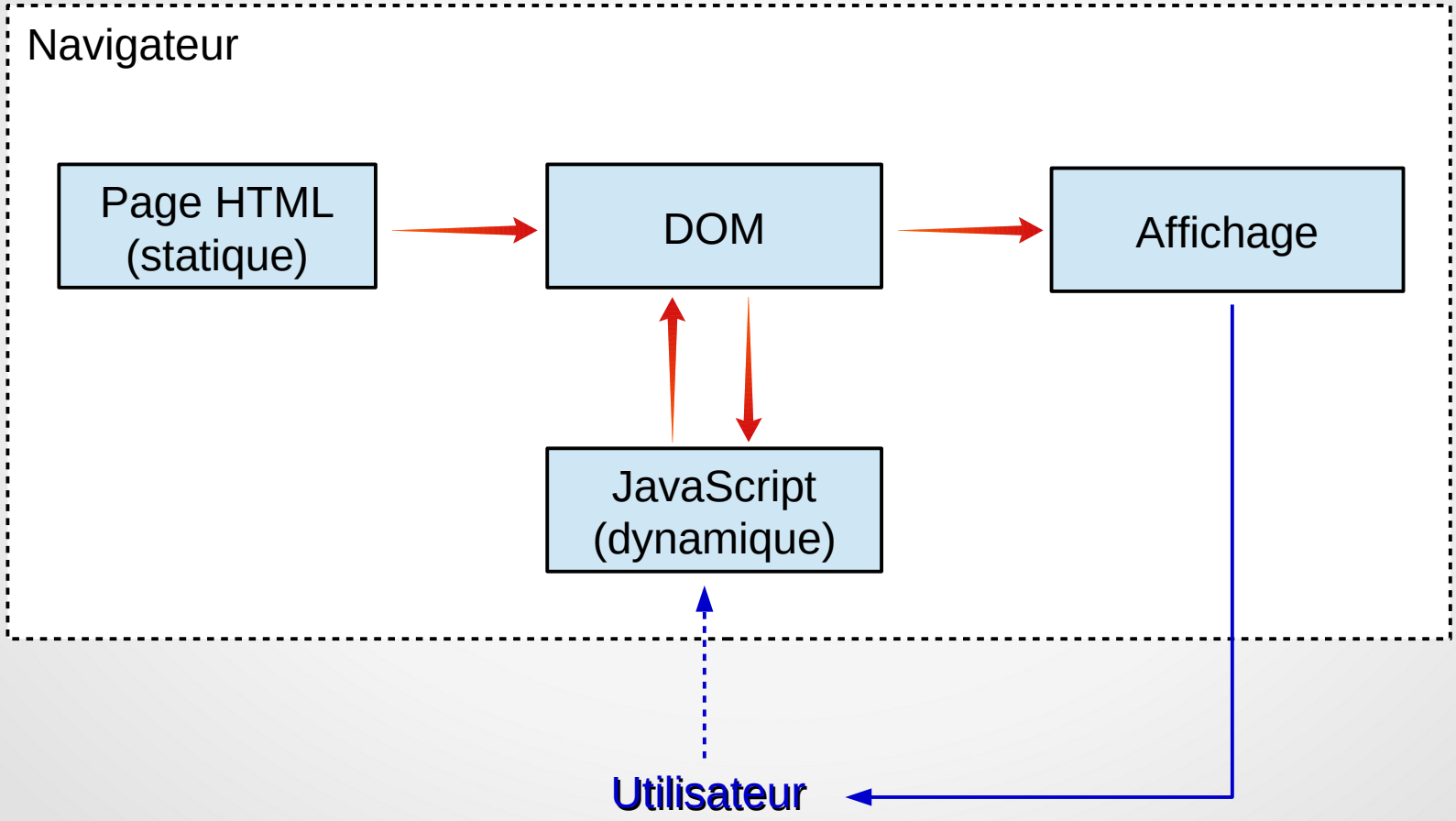
JAVASCRIPT
2e partie

DOM – *Document Object Model* (1)

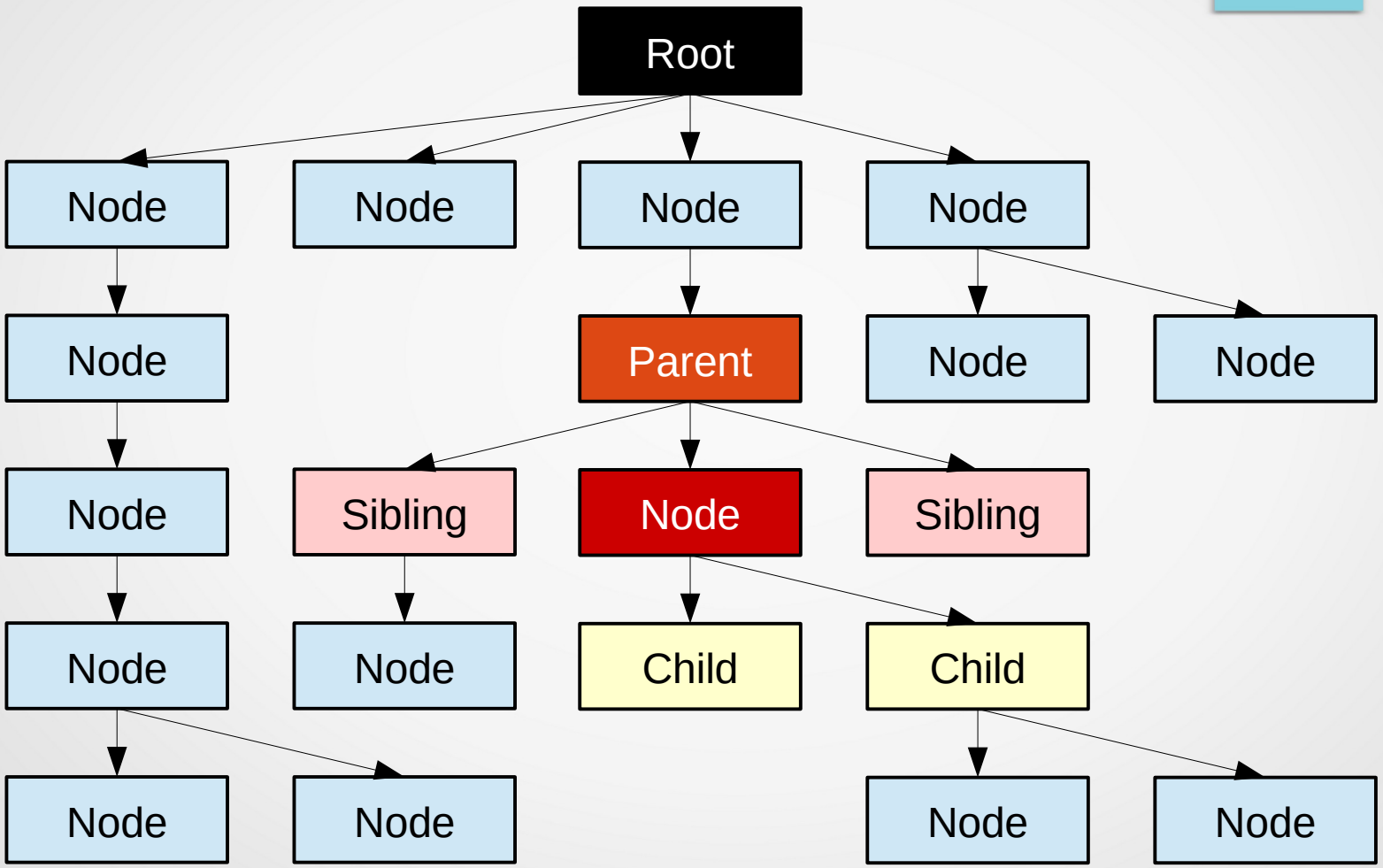
Le *Document Object Model* ou DOM est une plate-forme et une interface qui permet aux programmes d'accéder et de modifier dynamiquement le contenu, la structure et le style d'un document.

Le DOM est un standard du W3C.

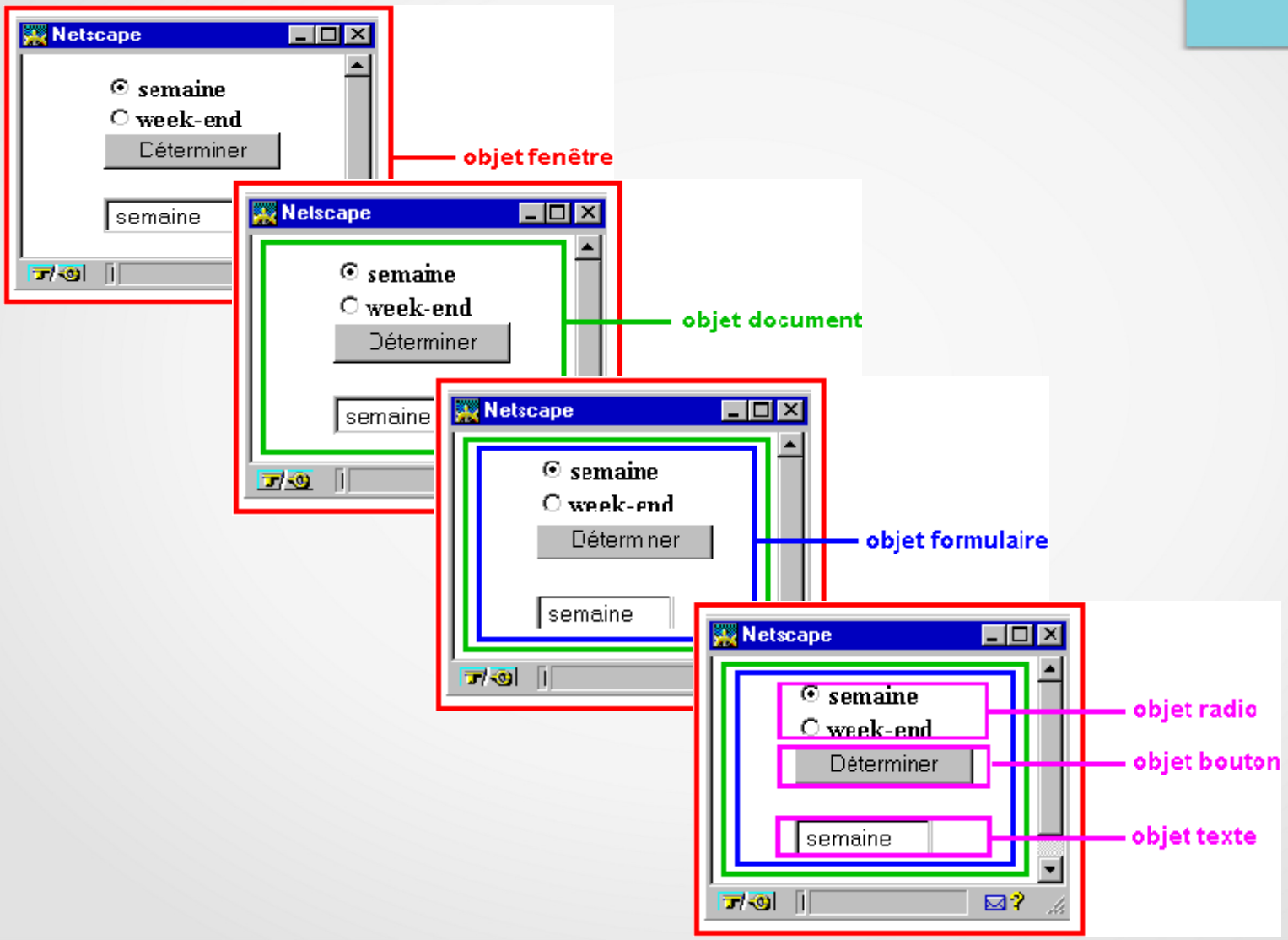
DOM – *Document Object Model* (2)



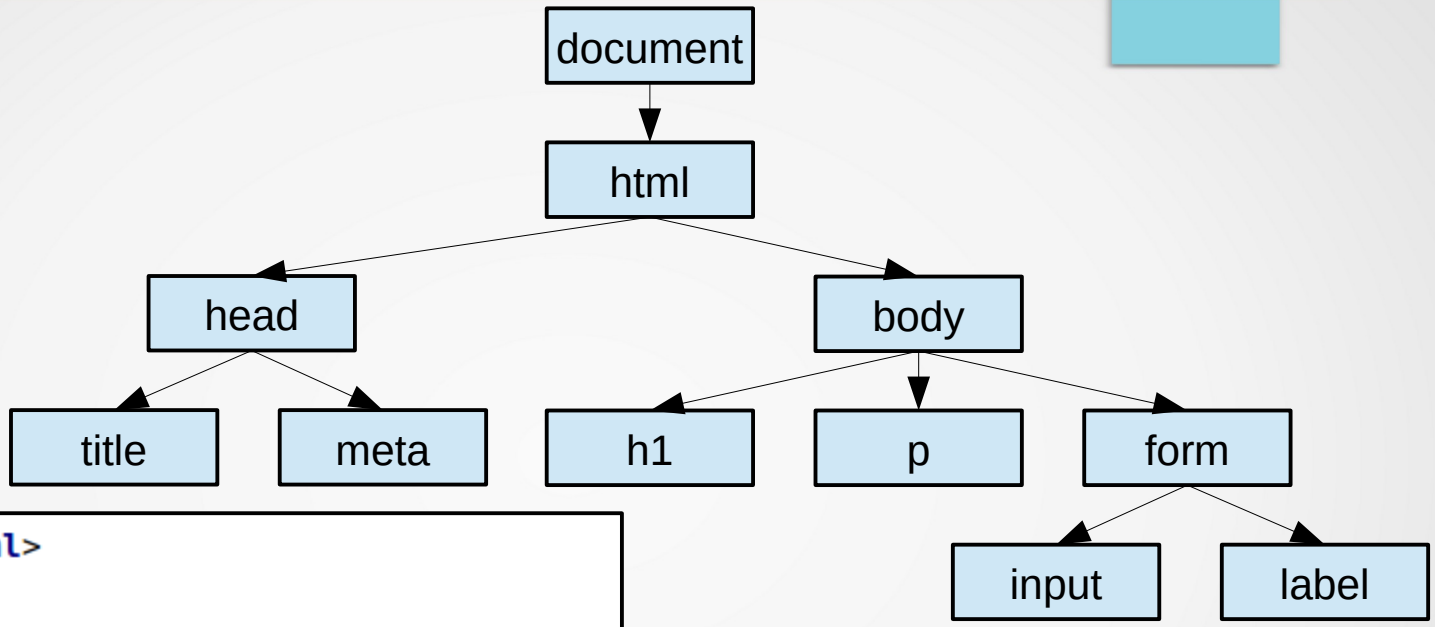
DOM – Terminologie



DOM – Approche visuelle



DOM – Document HTML




```
<!DOCTYPE html>
<html>
    <head>
        <title>Formulaire</title>
        <meta charset="utf-8">
    </head>
    <body>
        <h1>Formulaire</h1>
        <p>Cochez si vous être d'accord</p>
        <form name="form" method="post">
            <input type="checkbox" id="ok">
            <label for="OK">OK</label>
        </form>
    </body>
</html>
```

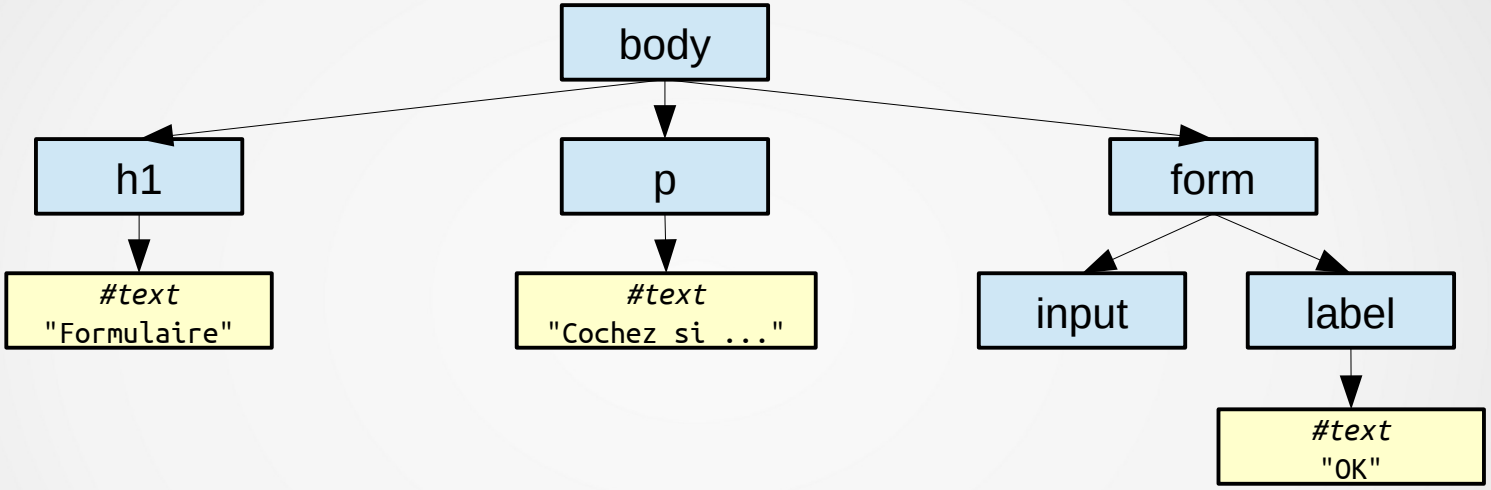
Types de nœuds

Pour l’instant, seuls les nœuds d’éléments ont été représentés.

Il existe toutefois d’autres types de nœuds :

Les nœuds textuels

Nœuds textuels (1)



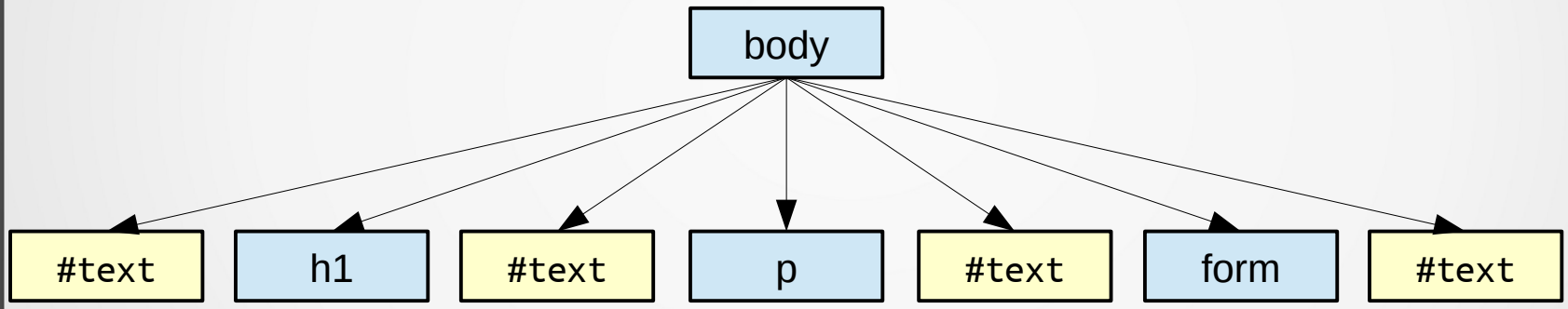

```
<body>
    <h1>Formulaire</h1>
    <p>Cochez si vous être d'accord</p>
    <form name="form" method="post">
        <input type="checkbox" id="ok">
        <label for="OK">OK</label>
    </form>
</body>
```

Nœuds textuels (2)

Attention !

Le texte séparant deux balises, c'est-à-dire le retour à la ligne et les espaces, sont également pris en compte comme des nœuds textuels.

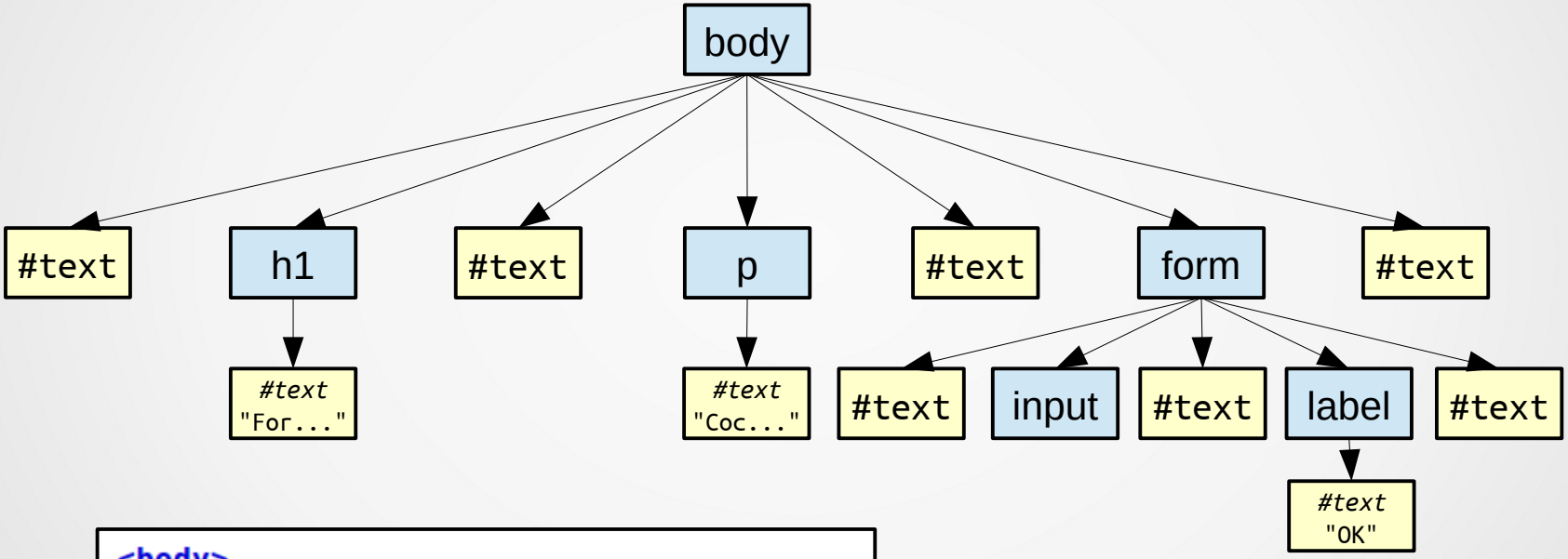
Les enfants de l'élément *body* sont donc :




```
<body>
    <h1>Formulaire</h1>
    <p>Cochez si vous être d'accord</p>
    <form name="form" method="post">
        <input type="checkbox" id="ok">
        <label for="OK">OK</label>
    </form>
</body>
```

Nœuds textuels (3)

Les descendants complets de l'élémement *body* sont donc :




```
<body>
    <h1>Formulaire</h1>
    <p>Cochez si vous être d'accord</p>
    <form name="form" method="post">
        <input type="checkbox" id="ok">
        <label for="OK">OK</label>
    </form>
</body>
```

Document – Propriétés et méthodes

L’objet document est accessible directement dans le code JavaScript à l’aide du mot clé *document*.

L’interface de l’objet *document* propose des propriétés et des méthodes facilitant la navigation dans le DOM.

Nous en verrons quelques-unes parmi les plus utiles.

Voir la documentation pour une description complète de l’objet document :
https://developer.mozilla.org/fr/docs/Web/API/Document

Récupérer les éléments du DOM (1)

Principales méthodes de l'objet *document* qui permettent de récupérer les éléments du DOM :

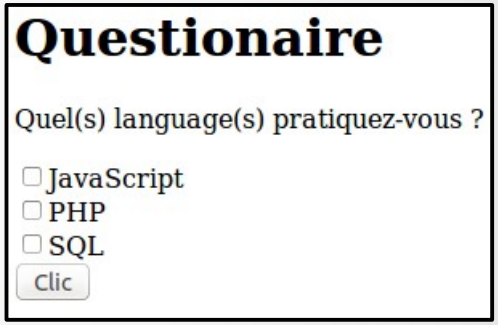
- getElementById()
  Renvoie l'élément dont l'ID est celui spécifié.
- getElementsByName()
  Renvoie une liste des éléments ayant le nom donné.
- getElementsByTagName()
  Renvoie une liste des éléments ayant le nom de balise donné.
- getElementsByClassName()
  Renvoie une liste des éléments ayant le nom de classe donné.
- querySelector()
  Renvoie le premier élément qui correspond au groupe de sélecteurs passés en paramètre.
- querySelectorAll()
  Renvoie une liste des éléments correspondent au groupe de sélecteurs passés en paramètre.

Récupérer les éléments du DOM (2)

```javascript
// Récupère l'élément CheckBox à partir de son identifiant.
var checkBoxJS = document.getElementById("js");

// Récupère tous les éléments "label".
var labels = document.getElementsByTagName("label");

// Récupère le premier élément "label".
var labelJS = labels[0];

// Récupère tous les éléments de classe "info".
var info = document.getElementsByClassName("info");

// Récupère le premier élément "input" de type "button".
var button = document.querySelector("input[type='button']");

// Récupère tous les éléments "input".
var inputs = document.querySelectorAll("input");
```

JavaScript

Affichage




```html
<body>
    <h1 class="info">Questionaire</h1>
    <p>Quel(s) language(s) pratiquez-vous ?</p>
    <form name="form" method="post">
        <input type="checkbox" id="js"><label for="js">JavaScript</label><br>
        <input type="checkbox" id="php"><label for="php">PHP</label><br>
        <input type="checkbox" id="sql"><label for="sql">SQL</label>
        <br><input type="button" value="Clic">
    </form>
</body>
```

HTML

Récupérer les éléments du DOM (3)

Quelques propriétés de l'objet *document* permettent d'accéder directement à certains éléments du DOM :

- head
  Renvoie l'élément *head* du document.
- body
  Renvoie l'élément *body* du document.
- anchors
  Renvoie toutes les ancres du document.
- links
  Renvoie tous les liens du document.
- images
  Renvoie toutes les images du document.
- forms
  Renvoie tous les formulaires du document.

Récupérer les éléments du DOM (4)

Si un élément possède un attribut *id* ou un attribut *name*, alors une simple variable JavaScript permet d'accéder à cet élément.

Attention ! S'il existe plusieurs attributs *name* identiques ou un attribut *id* identique à l'attribut *name* d'un autre élément. Ce type de conflit peut-être géré différemment selon les navigateurs. Il est donc préférable d'utiliser des noms et des identifiants uniques.

Exemple :

```
<form name="form"  method="post">
    <input type="checkbox" id="js"><label for="js">JavaScript</label><br>
    <input type="checkbox" id="php"><label for="php">PHP</label><br>
    <input type="checkbox" id="sql"><label for="sql">SQL</label>
    <br><input type="button" value="Clic">
</form>
```

La variable *form* contient une référence sur l'élément `<form name="form" ...>`.
La variable *js* contient une référence sur l'élément `<input id="js" ...>`.
La variable *php* contient une référence sur l'élément `<input id="php" ...>`.
La variable *sql* contient une référence sur l'élément `<input id="sql" ...>`.

Manipuler un élément

Une fois qu'un élément a été récupéré, une multitude de propriétés et de méthodes permettent de le manipuler.

Nous en verrons quelques-unes parmi les plus utiles.

Voir la documentation pour une description complète de l'objet élément :
https://developer.mozilla.org/fr/docs/Web/API/Element

Attribut d’un élément (1)

Il est possible de manipuler les attributs d’un élément grâce aux méthodes suivantes :

- hasAttributes()
  Vérifie si l'élément possède au moins un attribut.
- hasAttribute(name)
  Vérifie si l'élément possède ou non l'attribut spécifié.
- getAttribute(name)
  renvoie la valeur d'un attribut.
- setAttribute(name, value)
  Ajoute un nouvel attribut ou change la valeur d'un attribut existant.
- removeAttribute(name)
  Supprime l’attribut spécifié.
- attributes
  Renvoie tous les attributs.

Attribut d'un élément (2)

La plupart des attributs peuvent être lus ou écrits directement comme une propriété de l'élément. Attention toutefois, certains noms d'attribut peuvent changer. Par exemple :

- class → className
- for → htmlFor

```
// Récupère la valeur de l'attribut "method" d'un formulaire.
var form = document.forms[0];
var m = form.method;

// Récupère la valeur de l'attribut "type" d'un élément "text".
var input = document.getElementById("moyen");
var t = input.type;

// Récupère la valeur de l'attribut "class" d'un élément "p".
var p = document.querySelector("p");
var c = p.className;

// Récupère la valeur de l'attribut "for" d'un élément "label".
var label = document.querySelector("label");
var f = label.htmlFor;
```

Attribut d'un élément (3)

```
<script>
    function chgImg(nom, fichier) {
       tab = document.getElementsByName(nom);
       for (var i=0; i<tab.length; i++) { tab[i].src = fichier; }
    }
</script>
```

dans le head

```
<img height="200" width="200" name="indicateur" src="triste.jpg">
<div class="c2"
  onMouseOut="chgImg('indicateur','triste.jpg')"
  onMouseOver="chgImg('indicateur','content.jpg')" >
  Passe la souris sur moi
</div>
<img height=200 width=200 name="indicateur" src="triste.jpg">
```

dans le body

Manipuler les styles d’un élément (1)

Il est possible de lire ou d’écrire le style d’un élément grâce à son attribut *style*.

Le nom de la propriété CSS devra être utilisé sans tiret et avec une majuscule sur la première lettre des mots liés. La première lettre de la variable restera minuscule (lowerCamelCase).

Par exemple :
color → color (pas de changement).
background-color → backgroundColor.
font-size → fontSize

Manipuler les styles d'un élément (2)

```
<head>
    <title>Changement de couleur</title>
    <meta charset="utf-8">
    <script>
        function chgCouleur(balise, couleur)
        {
            tab = document.getElementsByTagName(balise);
            for (var i = 0; i < tab.length; i++)
                tab[i].style.color = couleur;
        }
    </script>
</head>
<body>
    <a href="#" onClick="chgCouleur('p','red')">
        Mettre tous les paragraphes en rouge
    </a>
    <p>Lorem ipsum dolor sit amet ...</p>
    <p>Encore un paragraphe ...</p>
    <p>Et puis un autre ...</p>
</body>
```

Mettre tous les paragraphes en rouge

Lorem ipsum dolor sit amet ...

Encore un paragraphe ...

Et puis un autre ...

Mettre tous les paragraphes en rouge

Lorem ipsum dolor sit amet ...

Encore un paragraphe ...

Et puis un autre ...

Manipuler les styles d'un élément (3)

```
<head>
    <meta charset="utf-8">
    <title>Changement de style</title>
    <style>
      .nouveau_style {color: red; font-size: 20pt;}
    </style>
    <script>
        function chgClass(balise, classe)
        {
            tab = document.getElementsByTagName(balise);
            for (var i = 0; i < tab.length; i++)
                tab[i].className = classe;
        }
    </script>
</head>
<body>
    <a href="#" onClick="chgClass('p','nouveau_style');">
        Changer de style
    </a>
    <p>Lorem ipsum dolor sit amet ...</p>
    <p>Encore un paragraphe ...</p>
    <p>Et puis un autre ...</p>
</body>
```

Changer de style

Lorem ipsum dolor sit amet ...

Encore un paragraphe ...

Et puis un autre ...

Changer de style

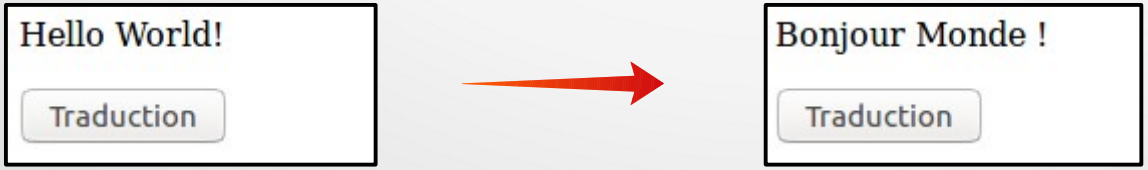
Lorem ipsum dolor sit amet ...

Encore un paragraphe ...

Et puis un autre ...

Modification de contenu

```
<html>
    <head>
        <title>Modifier un contenu</title>
        <meta charset="utf-8">
        <script>
            function changeText(id, newText)
            {
                document.getElementById(id).innerHTML = newText;
            }
        </script>
    </head>
    <body>
        <p id="p1">Hello World!</p>
        <input type="button"
               onclick="changeText('p1','Bonjour Monde !')"
               value="Traduction">
    </body>
</html>
```



Les objets node (1)

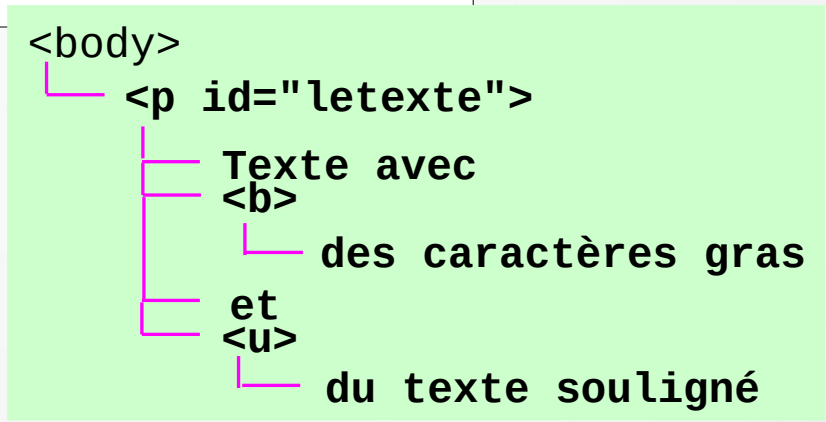
Propriétés :

- childNodes = tableau des nœuds enfants
- nodeValue = texte du nœud
- firstChild = premier nœud enfant

Les objets node (2)

```
<body>
 <p id="letexte">
 Texte avec <b>des caractères gras</b>
 et <u>du texte souligné</u></p>
</body>
```

Sous-arbre du DOM correspondant :




```
p = document.getElementById("letexte")
p.childNodes.length    →   4
p.firstChild.nodeValue    →   Texte avec
p.childNodes[1].childNodes[0].nodeValue    →   des caractères gras
```

Les objets node (3)

Méthodes:

- **createElement()** : crée un nouvel élément HTML
- **createTextNode()** : crée un node texte
- **appendChild()** : ajoute un nœud fils à un node
- **removeChild()** : supprime un nœud fils

Les objets node (4)

```
<div id="div1">
 <p id="p1"> Premier paragraphe </p>
 <p id="p2"> Deuxième paragraphe</p>
</div>


<script>
  var para = document.createElement("p");
  document.getElementById("div1").appendChild(para);
  var texte = document.createTextNode("3eme paragraphe");
  para.appendChild(texte);
</script>
```

Les objets node (5)

```
<script>
function restriction() {
   var parent = document.getElementById('restricted');
   var enfant = parent.firstChild;
   parent.removeChild(enfant) ;
}
</script>
```

```
Etes-vous ma mère ? <br>
oui <input type="radio" onClick="restriction()"> <br>
non <input type="radio"> <br>
<ul>
  <li> Mes TP de java </li>
  <li> Mon projet de programmation </li>
  <li id="restricted"> Mes photos de soirée </li>
</ul>
```

Formulaires

Ex :

```
<body>
<form name="formulaire">
        <input type="text" name="un_champ_txt">
```

Accès au formulaire : `document.forms[0]` OU `document.formulaire`

Accès à un élément : `document.formulaire.elements[0]`
OU `document.formulaire.un_champ_txt`

Méthodes : `submit()` et `reset()`

Evénements `onsubmit` et `onreset`

Formulaires : champs **input**

text, button, radio, checkbox, submit, reset

Propriétés : name type value defaultvalue size maxlength
checked disabled readOnly class style

Méthodes : focus() et blur()

Evénements onfocus, onblur et onClick

Ex :

```
<form name="fo">
  <label> Titre </label>
  <input type="text" name="titre"
   onfocus="this.style.backgroundcolor='yellow'"
   onblur="verif(this.value)"/>
   ...
</form>
```

Formulaires : champs **`select`**

Propriétés de `select` :

`name`	Nom de la liste
`size`	Nombre de lignes à afficher
`multiple`	Sélection multiple autorisée
`disabled`	Grisage de la liste
`style`	Style de la liste
`selectedIndex`	indice de la ligne courante
`options`	tableau des options

Propriétés de `options` :

`length` nombre de lignes

Propriétés de options[i] :

value	Valeur d'une ligne
text	Libellé d'une ligne
defaultSelected	option choisie par défaut
selected	option sélectionnée

Méthodes : `add()` `remove()` `focus()` `blur()`

Formulaires : champs **select** : ex

```
<script> function ajouter() {
// création d'une nouvelle valeur d'option
nouvel_element = new Option(document.forms[0].nouveau.value);
// Positionnement de la nouvelle option à la fin de la liste
document.forms[0].choix.options[document.forms[0].choix.length] =
nouvel_element ;
// Réinitialisation du champ texte
document.forms[0].nouveau.value = ""; } </script>
```

```
<form id="formulaire" >
<select id="choix" size="8">
        <option>Un élément</option>
</select>
<p /> Entrer le nouvel élément à ajouter à la liste :
<input type="text" id="nouveau"><br />
<input type="button" value="Ajouter" onClick="ajouter()">
</form>
```

Formulaires : manips classiques

- Récupérer le nombre de lignes :

  form.**ma_liste**.**options**.**length**

  Formulaire auquel appartient cet élément

  ID donné à l'input select

  Tableau des options

  NB : on peut aussi utiliser : document.getElementById('ma_liste')

- Récupérer l'indice de la ligne sélectionnée :

  i = form.ma_liste.**selectedIndex**

- Récupérer la valeur sur cette ligne :

  form.ma_liste.**options[i]**

- Ajouter une option :

  nouvelle = **new Option**(texte) ;

  form.ma_liste.options.**add**(nouvelle)

- Supprimer une option :

  form.ma_liste.options.**remove**(indice)

Minuteries (1)

Méthodes de l'objet window :

Déclenchement : création d'un objet minuterie :

```
var minuterie = window.setTimeout(nom_fonction, temps en ms)
```

→ appelle une fonction au bout du temps indiqué

OU

```
var minuterie = window.setInterval(nom_fonction, intervalle en ms)
```

→ appelle une fonction à intervalles réguliers

Arrêt :

`clearTimeout(minuterie)` (1er cas)

`clearInterval(minuterie)` (2nd cas)

Minuteries (2)

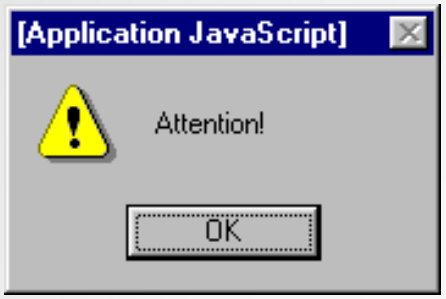
```
var i = 0, c = 1;
var minuteur = window.setInterval("couleur()",500);

function couleur()
{
        if (c == 1)
        {
                document.bgColor="yellow";
                c = 2;
        }

        else
        {
                document.bgColor="aqua";
                c = 1;
        }

        i++;

        if(i >= 10)
                window.clearInterval(minuteur);
}
```

document.write()

Méthodes :

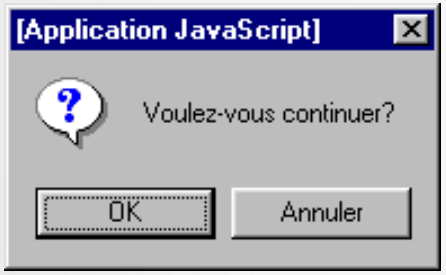
```
<html>  <head>
  <script language="javascript">
   function coucou()
         {
         var texte = "<html> <body> Salut ! </body> </html>" ;
         var win = open("","PageBlanche"); // ouverture page blanche
         win.document.open() ;    // ouverture du flux d'affichage
         win.document.write(texte) ; // écriture du contenu
         win.document.close() ; // ferme le flux d'affichage
         }
   </script>  </head>
<body>
<form>
<input type="button" value="coucou" onclick="javascript:coucou()" />
</form>
</body></html>
```

Boîtes de dialogue



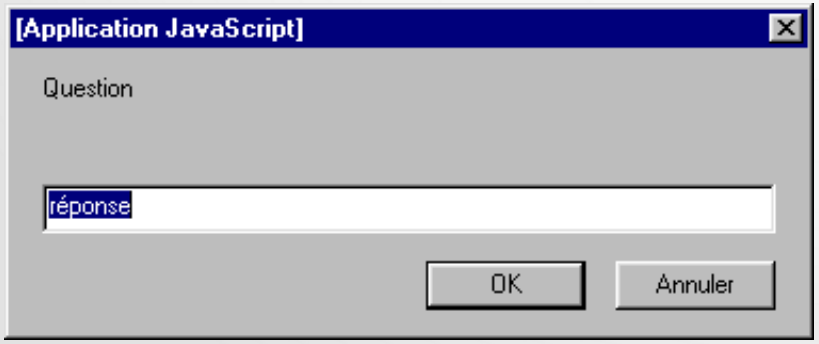

```
alert('Chaîne de caractères');
```

Pour introduire un passage à la ligne : « \n »




```
result = confirm('Chaîne de caractères');
```

retourne TRUE ou FALSE




```
result = prompt('Chaîne',
        'réponse par defaut);
```

→ *retourne la chaîne saisie*
ou null si aucun texte