JavaScript

Magali Contensin

Plan

- Présentation de JavaScript
- ECMAScript : le noyau du langage JavaScript
- Manipulation des objets du document
  - Des débuts du dynamisme au DOM
  - Modifier une page web avec JS, DOM et CSS
  - Formulaires
- JavaScript discret

Présentation de JavaScript

- Langage de programmation orienté-objet interprété par le navigateur

  syntaxe proche de Java

- Histoire
  - Netscape 2.0 **JavaScript** – mars 1996
  - Microsoft IE 3.0 **JScript** (en partie compatible avec JS) – août 1996
  - Rendu public par Netscape, standardisé par l'ECMA en juin 1997
- Evénementiel : permet de réagir à des actions de l'utilisateur

  Ajout de dynamisme dans la page web
  - Vérification de formulaires
  - Avant CSS : roll-over, menus
  - Modification de la structure, du contenu et du style de la page
- Deux parties :
  - Syntaxe du langage : **ECMAScript**
  - Manipulation des objets du document : **DOM**

Présentation de JavaScript

insertion de code JS dans la page web

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
    <title>test JS</title>
    <script type="text/javascript" src="script.js"></script>
</head>
<body>
```

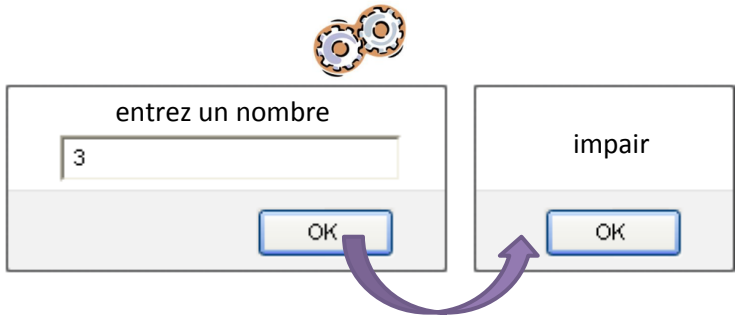
1

```
</body>
</html>
```

```
function pair(){
    nb = prompt('entrez un nombre', '');
    (nb%2 == 0) ? alert('pair') : alert('impair');
}
```

script.js

1



Le **code** peut être placé dans :

(1) fichier externe .js

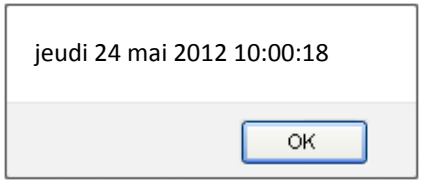
Présentation de JavaScript

insertion de code JS dans la page web

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>test JS</title>
   <script type="text/javascript" src="script.js"></script>
</head>
<body>
   <script type="text/javascript">
   <!--
      function afficheDate(){
         var aujourdhui = new Date();
         alert(aujourdhui.toLocaleString());
      }
   // -->
   </script>
   <noscript><p>activez JS !</p></noscript>
</body>
</html>
```

1

2




```
function pair(){
   nb = prompt('entrez un nombre', '');
   (nb%2 == 0) ? alert('pair') : alert('impair');
}
```

script.js

1

Le **code** peut être placé dans :

(1) fichier externe .js

(2) élément script

Présentation de JavaScript

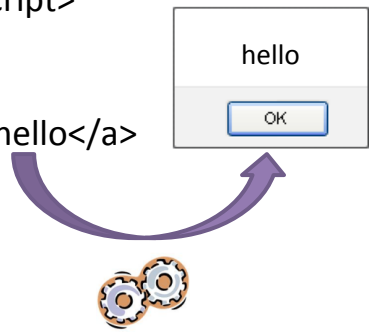
insertion de code JS dans la page web

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>test JS</title>
   <script type="text/javascript" src="script.js"></script>
</head>
<body>
    <script type="text/javascript">
    <!--
      function afficheDate(){
        var aujourdhui = new Date();
        alert(aujourdhui.toLocaleString());
      }
    // -->
    </script>
    <noscript><p>activez JS !</p></noscript>
    <div>
      <h1>Test JS</h1>
      <a href="javascript:alert('hello')">hello</a>
    </div>
</body>
</html>
```

1

2

3




```
function pair(){
   nb = prompt('entrez un nombre', '');
   (nb%2 == 0) ? alert('pair') : alert('impair');
}
```

script.js

1

Le **code** peut être placé dans :

(1) fichier externe .js

(2) élément script

(3) attribut contenant URL

Présentation de JavaScript

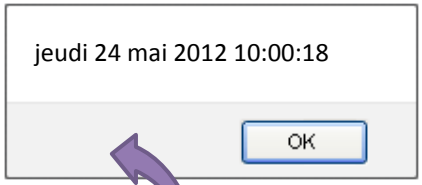
insertion de code JS dans la page web

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>test JS</title>
   <script type="text/javascript" src="script.js"></script>
</head>
<body>
   <script type="text/javascript">
   <!--
      function afficheDate(){
         var aujourdhui = new Date();
         alert(aujourdhui.toLocaleString());
      }
   // -->
   </script>
   <noscript><p>activez JS !</p></noscript>
   <div>
      <h1>Test JS</h1>
      <a href="javascript:alert('hello')">hello</a>
      <a href="javascript:afficheDate()">date</a>
   </div>
</body>
</html>
```

1

2

3




```
function pair(){
   nb = prompt('entrez un nombre', '');
   (nb%2 == 0) ? alert('pair') : alert('impair');
}
```

script.js

1

Le **code** peut être placé dans :

(1) fichier externe .js

(2) élément script

(3) attribut contenant URL

Présentation de JavaScript

insertion de code JS dans la page web

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>test JS</title>
   <script type="text/javascript" src="script.js"></script>
</head>
<body>
   <script type="text/javascript">
   <!--
      function afficheDate(){
         var aujourdhui = new Date();
         alert(aujourdhui.toLocaleString());
      }
   // -->
   </script>
   <noscript><p>activez JS !</p></noscript>
   <div>
      <h1>Test JS</h1>
      <a href="javascript:alert('hello')">hello</a>
      <a href="javascript:afficheDate()">date</a>
      <form action="test.php">
         <div><input type="button" value="clic" onclick="pair()"></div>
      </form>
   </div>
</body>
</html>
```

(1)

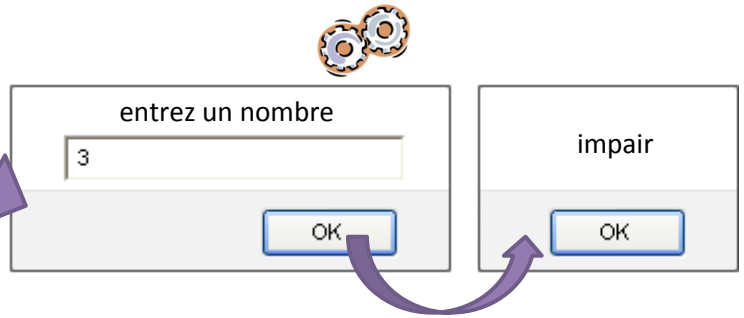
(2)

(3)

(4)

```
function pair(){
   nb = prompt('entrez un nombre', '');
   (nb%2 == 0) ? alert('pair') : alert('impair');
}
```

script.js (1)



Le **code** peut être placé dans :

- (1) fichier externe .js
- (2) élément script
- (3) attribut contenant URL
- (4) attribut événementiel
  onclick, onchange, onselect,…

Plan

- Présentation de JavaScript
- **ECMAScript : le noyau du langage JavaScript**
- Manipulation des objets du document
  - Des débuts du dynamisme au DOM
  - Modifier une page web avec JS, DOM et CSS
  - Formulaires
- JavaScript discret

ECMAScript

bases du langage

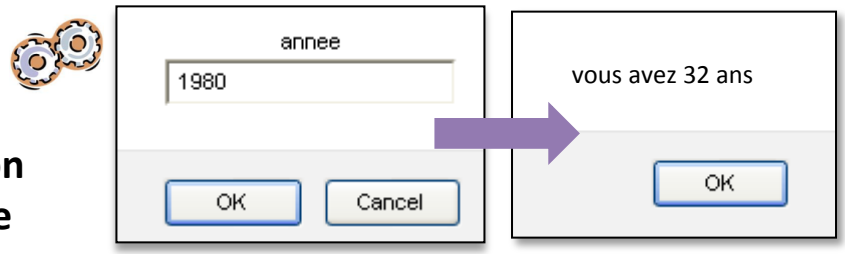
variable

types
boolean
number
string
object
null
undefined

marque de fin d'instruction

création objet

invocation méthode




```
var ne = 0;
var age = 0;
var txt = '';
var date_courante = new Date();
var annee_courante = date_courante.getFullYear();
var exp = /^[12]\d{3}$/;     // format 1xxx ou 2xxx

/* tant que l'utilisateur n'a pas saisi une
   annee correcte afficher boite dialogue */
do {
     ne = prompt('annee ', '');

} while (  !exp.test(ne )   ||   (ne > annee_courante));

age = annee_courante - ne;

txt= 'vous avez ' + age + ' ans';
(age%10 == 0) ? alert(txt+' ... nouvelle decennie') : alert(txt);
```

commentaires

marques de bloc

opérateur conditionnel ternaire

opérateurs

affectation
= += -= *= /= %=

logiques ! && ||

relationnels
< <= > >= == != === !==

arithmétiques + - */ %

concaténation +

autres opérateurs
incrémentation ++
décrémentation --
séquentiel ,
objets : this, in

ECMAScript

bases du langage

- Instructions conditionnelles

```
if (expr){
  // code si expression true
}
else {
  // code si expression false
}
```

```
var today = new Date();
switch (today.getDay()){
   case 0 :
   case 6 : jour = "week-end"; break;
   case 1 :
   case 2 :
   case 4 :
   case 5 : jour = "semaine"; break;
   case 3 : jour = "mercredi"; break;
   default : jour = "non defini";
}
alert(jour);
```

- Instructions itératives

```
while (expr){
  ...
}
```

```
do{
  ...
} while (expr);
```

```
for (exp1 ; exp2 ; exp3){
   ...
}
```

```
for (var i in objet){
   ...
}
```

- Instructions d'interruption :

  break, **continue**

ECMAScript

bases du langage

- Fonctions

Déclaration

```
function nom_fonction([arg1,...,argN]){
   ...
   return expression;  // optionnel
}
```

Appel

```
nom_fonction([arg1, arg2, ...]) ;
```

Les arguments sont passés par valeur
Les objets sont passés par adresse

- Mathématiques

Math

Ne peut pas être utilisé avec new

Constantes

```
Math.PI
```

Méthodes

ceil, floor, round
min, max
random
abs, sqrt, pow
trigonométrie : sin, cos, …

```
alert(Math.ceil(7.48));
alert(Math.random());
```

ECMAScript

bases du langage

- Tableaux

Création

```
var tab1 = new Array(val1, val2, …);
var tab2 = [val1, val2, …];
var tab3 = new Array();
```

Accès

tab[i] avec i >= 0

Propriétés

Taille d'un tableau : length

Parcours

for, for in, while

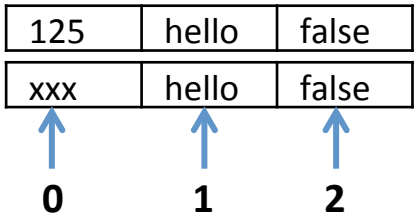
Méthodes

join, reverse, sort, …

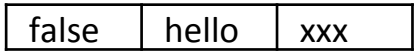
```
var tab = new Array(125, 'hello', false);
tab[0] = 'xxx';

var contenu = '';
for (var i=0; i < tab.length; i++){
      contenu += tab[i]+', ';
}
alert(contenu);

contenu = '';
tab.reverse();
for (i in tab){
      contenu += tab[i]+', ';
}
alert(contenu);
```



length : 3



ECMAScript

bases du langage

- Chaînes de caractères

Création

```
var ch1 = new String('hello !');
var ch2 = 'hello !';
ch2 = "test";
```

Caractères spéciaux

Caractère d'échappement : \
Caractères espacement : \n \t \f

Propriétés

Taille d'une chaîne : **length**

Méthodes

substring, toLowerCase, toUpperCase
split, …
Conversion :
parseInt, parseFloat

```
<input type="text" name="nb1"> +
<input type="text" name="nb2"> =
<input type="text" name="res">
```

```
<input type="button" value="calculer"
  onclick="res.value = nb1.value + nb2.value">
```

12 + 44 = 1244 calculer

```
<input type="button" value="calculer"
  onclick="res.value =parseFloat(nb1.value) +
                      parseFloat(nb2.value)">
```

12 + 4 = 16 calculer

ECMAScript

bases du langage

- Expressions régulières

Déclaration

```
var exp = new RegExp('modele'['options']);
var exp = /modele/[options]
Options :   i insensible à la casse
            m multi-ligne
            g toutes les correspondances
```

Méthodes

exp.test(ch)	true si correspondance
ch.search(exp)	indice 1ere correspondance
exp.exec(ch)	tableau des correspondances
ch.replace(exp, ch2)	remplace corresp par ch2

```
var exp = /(\d{2}) (\w+) (\d{4})/;
var ch = "24 mai 2012";
if (exp.test(ch)){
    alert(ch.replace(exp,
                "annee : $3, mois : $2, jour : $1"));
        // annee : 2012, mois : mai, jour : 24
}
```

Indicateurs d'occurrence

{n}	exactement n fois	*	{0,}
{n,}	au moins n fois	+	{1,}
{n,m}	entre n et m fois	?	{0,1}

Mode glouton par défaut (paresseux ajouter **?**)

Caractères spéciaux

|	ou	.	tout caractère sauf \n
\t	tabulation	\n	saut de ligne
\0	car. nul	\	car. d'échappement

Classes de caractères

[abc]	un caractère parmi a, b ou c		
[a-z]	intervalle : un caractère de a à z		
[^ab]	un caractère autre que a ou b		
\d	un chiffre	\D	tout sauf un chiffre
\w	[a-zA-Z0-9–]	\W	tout sauf mot
\s	espacement	\S	tout sauf car. esp.

Correspondances dans la chaîne

^	début	$	fin

Mémorisation

(x)	Mémoriser sous expression x

ECMAScript

bases du langage

- Dates et heures

Création

var date_courante = new **Date**([timestamp]);
ma_date = new Date(an, mois, jour[, h, mn, s, ms]);

Méthodes pour les dates

getDay	
getMonth	setMonth
getFullYear	setFullYear
getDate	setDate

Méthodes pour les heures

getHours	setHours
getMinutes	setMinutes
getSeconds	setSeconds
getMilliseconds	setMilliseconds

toLocaleString

```
var jour = new Date();
alert(jour.getFullYear());
// 2012

var noel = new Date(2011, 11, 25);
alert(noel.toLocaleString());
// dimanche 25 décembre 2011 00:00
```

ECMAScript

bases du langage

- Objets

Création

Pas de classe, fonction constructeur ou notation { }

Création d'objet : **new**

```
function calculeAge() {
   var auj = new Date();
   var age_courant = auj.getFullYear() - this.annee;
   return age_courant;
}
// constructeur
function humain(nom, prenom, habite, an){
  // proprietes
  this.nom = nom;
  this.prenom = prenom;
  this.pays = habite;
  this.annee = an;
  this.age = calculeAge;    // methode
}
```

Exceptions

```
try {
   // code où une exception peut être levée
} catch (e){
    // traitement en cas d'exception
}
```

```
var client1 = new  humain('Dupont','Jean',
                          'France', 1960);
alert(client1.prenom);     // Jean
alert(client1.age());      // 52
```

Notation alternative objet ECMA v3

```
client1 = {
      'nom':'Dupont',
      'prenom':'Jean',
      'pays':'France',
      'annee':1960,
      'age': calculeAge
}
```

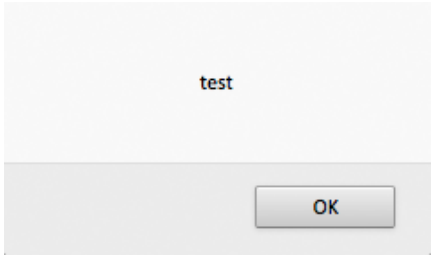
ECMAScript

ECMA 5ème édition

- Strict mode

```
"use strict";
x = 'test';
alert('test');
```


```
"use strict";
var x;
x = 'test';
alert('test');
```



ECMAScript

ECMA 5ème édition

- JSON (JavaScript Object Notation) – RFC 4627
  - Sérialisation de données
  - 4 types primitifs : chaînes, nombres, booléens, null
    - 2 types structurés :
      - Tableaux
      - Objets

1	bonjour	5	12.4	true	*objet*

```
[
    1,
    "bonjour",
    5,
    12.4,
    true,
    {
        'nom':'Smith',
        'prenom':'John',
        'annee':1960,
    }
]
```

ECMAScript

ECMA 5ème édition

- JSON (JavaScript Object Notation)
  - Objet JSON Méthodes
    - stringify : *serialize* JSON.stringify(arg)
    - parse : *unserialize* JSON.parse(arg)

```
var tab = new Array(1, 'bonjour', 5, 12.4, true);
alert(JSON.stringify(tab));          // [1,"bonjour",5,12.4,true]

var client1 = new  humain('Dupont', 'Jean', 'France', 1960);
alert(JSON.stringify(client1));      // {"nom":"Dupont","prenom":"Jean","pays":"France","annee":1960}

var contact = JSON.parse('{"nom":"Smith","prenom":"John","annee":1960}');
alert(contact.prenom);               // John

var t = JSON.parse('[1, "hello", false]');
alert( t[1] );                       // hello
```

Plan

- Présentation de JavaScript
- ECMAScript : le noyau du langage JavaScript
- **Manipulation des objets du document**
  - Des débuts du dynamisme au DOM
  - Modifier une page web avec JS, DOM et CSS
  - Formulaires
- JavaScript discret

Manipulation des objets du document

des débuts du dynamisme au DOM

- Netscape & IE <= 4 accès à un nombre restreint d'éléments de la page :
  - images — document.images[0].src
  - champs de formulaires — document.forms[0].elements[0].name
  - liens hypertextes — document.links[0].href
- DHTML - modification dynamique de documents HTML

  Modification du style - 2 méthodes incompatibles :
  - Netscape 4 à 6 — document.layers
  - IE >= 4 — document.all
- DOM (Document Object Model) – W3C

  Indique comment accéder dynamiquement au contenu, au style et à la structure de documents XML et HTML
  - Modification des contenus des éléments et attributs (textes, images, …)
  - Modification de la structure du document : ajout / suppression / modification d'éléments
  - Modification du style de la page

Manipulation des objets du document

dynamisme

- Compte à rebours (**setTimeout**), exécution périodique de code (**setInterval**)
- Boîtes de dialogue (**alert**, **confirm**, **prompt**) et barre d'état (**status**)
- Déterminer la configuration de l'utilisateur
  - écran **screen**.width
  - navigateur **navigator**.userAgent
- Fenêtres : ouverture, fermeture, déplacement, redimensionnement
- URL window.**location**.replace('http://www.google.fr');
- Historique <input type="button" value="back" onclick="window.**history**.back()">
- Cookies document.**cookie**
- Images : redimensionnement, roll-over

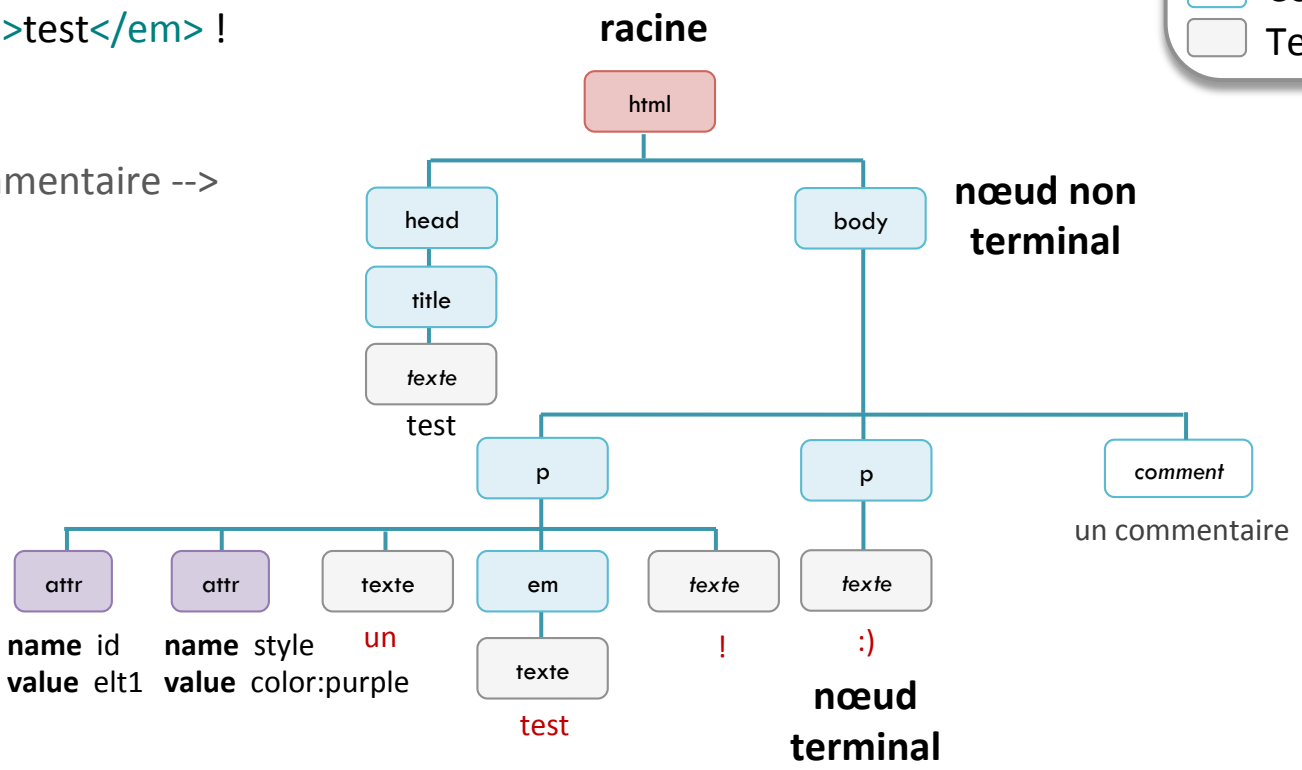
  document.images[0].**width** *= 2

  <img src="terre.jpg" alt="planetes" onmouseover="this.**src**='lune.jpg'" onmouseout="this.src='terre.jpg'">
- Modification de la structure, du contenu et du style de la page

Manipulation des objets du document

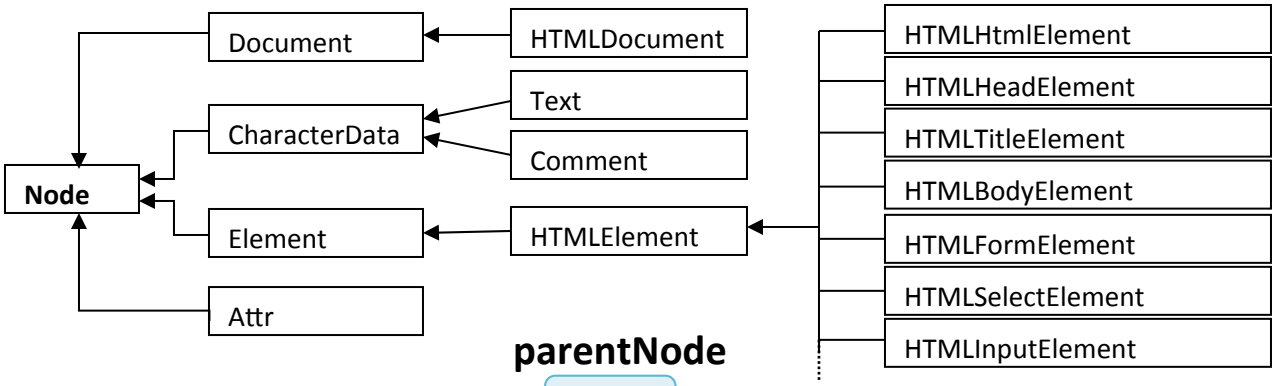
DOM – arbre du document

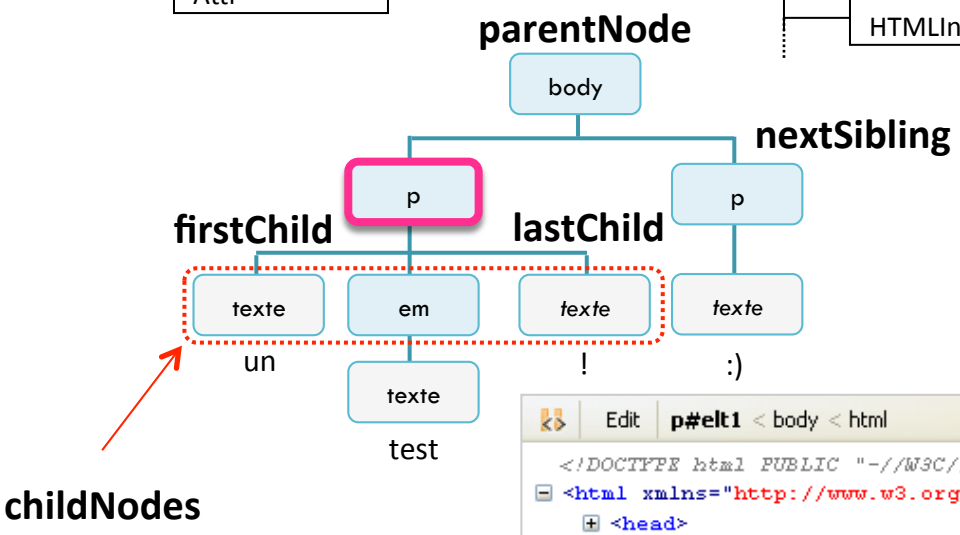
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
   <title>test</title>
</head>
<body>
  <p  id="elt1" style="color:purple">
       un <em>test</em> !
  </p>
  <p>:)</p>
  <!-- un commentaire -->
</body>
</html>
```



Manipulation des objets du document

DOM – propriétés des nœuds





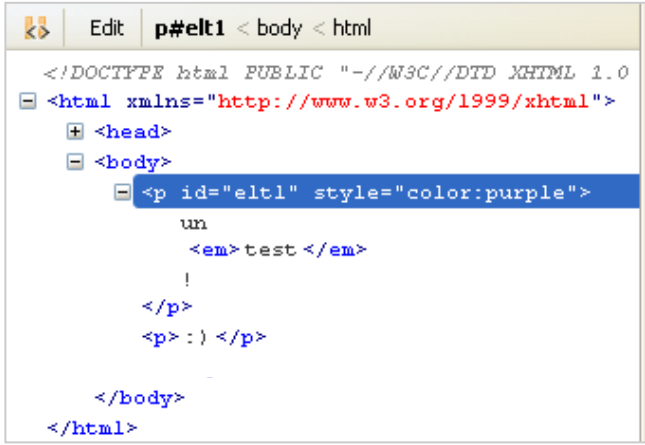
node

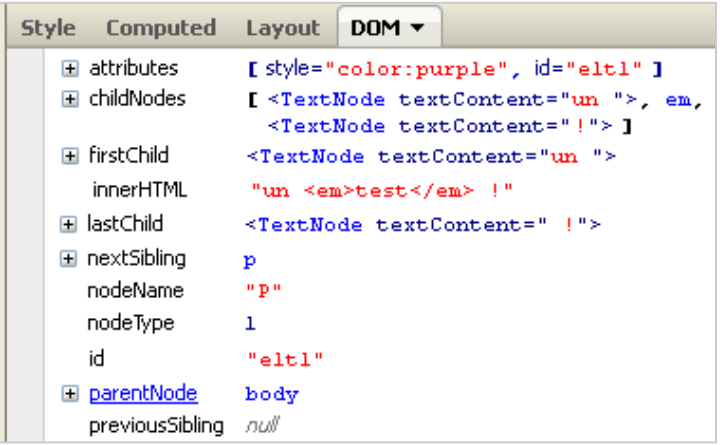
nodeName	nodeType
parentNode	childNodes
firstChild	lastChild
nextSibling	previousSibling
attributes	

élément	**élément HTML**
tagName	className title id

attribut	
name	value

texte	
data	length





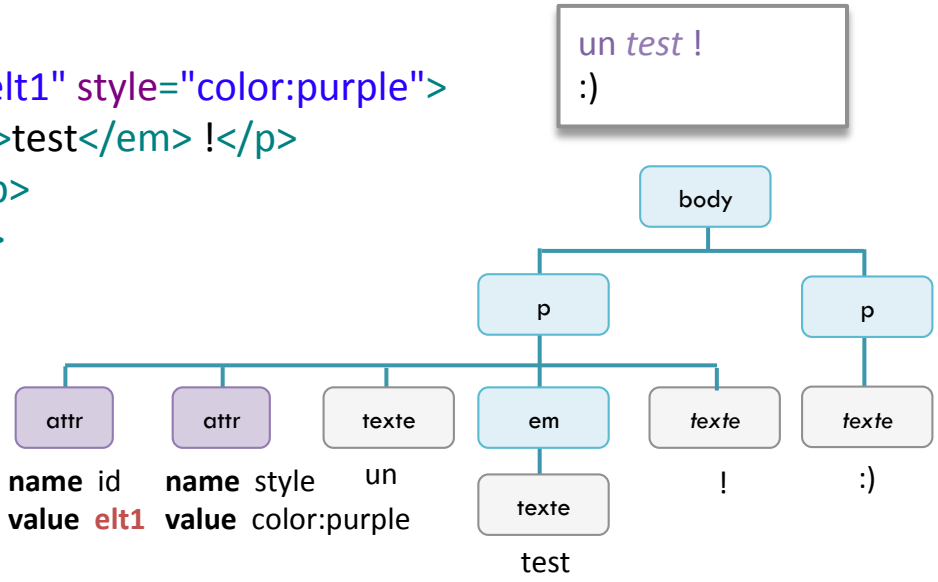
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID document.**getElementById**('*id_element*') -> noeud

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



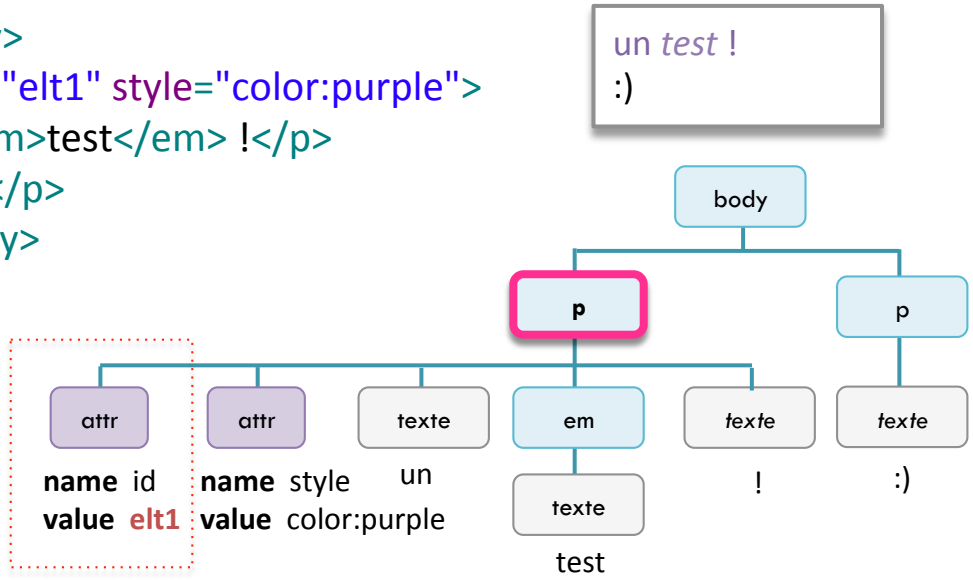
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID document.**getElementById**('*id_element*') -> noeud

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```




```
var el = document.getElementById('elt1');
```

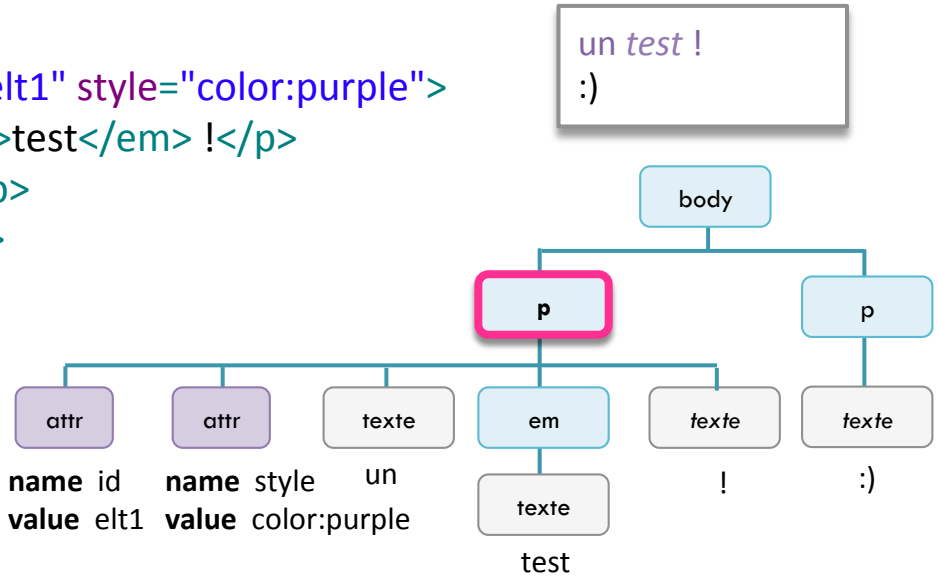
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID document.**getElementById**('*id_element*') -> noeud

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```




```
var el = document.getElementById('elt1');
alert(el.tagName);                    // P
```

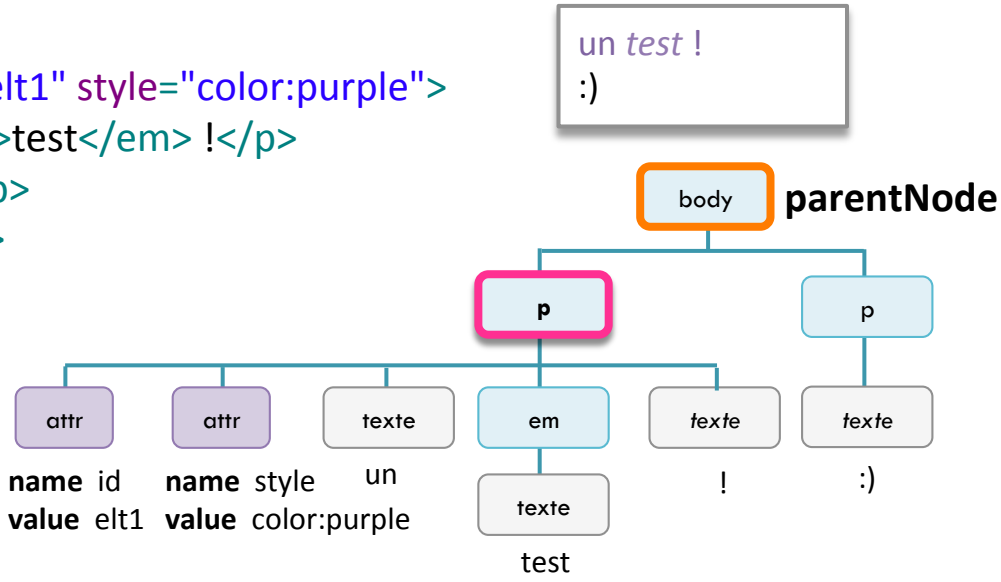
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID document.**getElementById**('*id_element*') -> noeud

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```




```
var el = document.getElementById('elt1');
alert(el.parentNode.nodeName);     // BODY
```

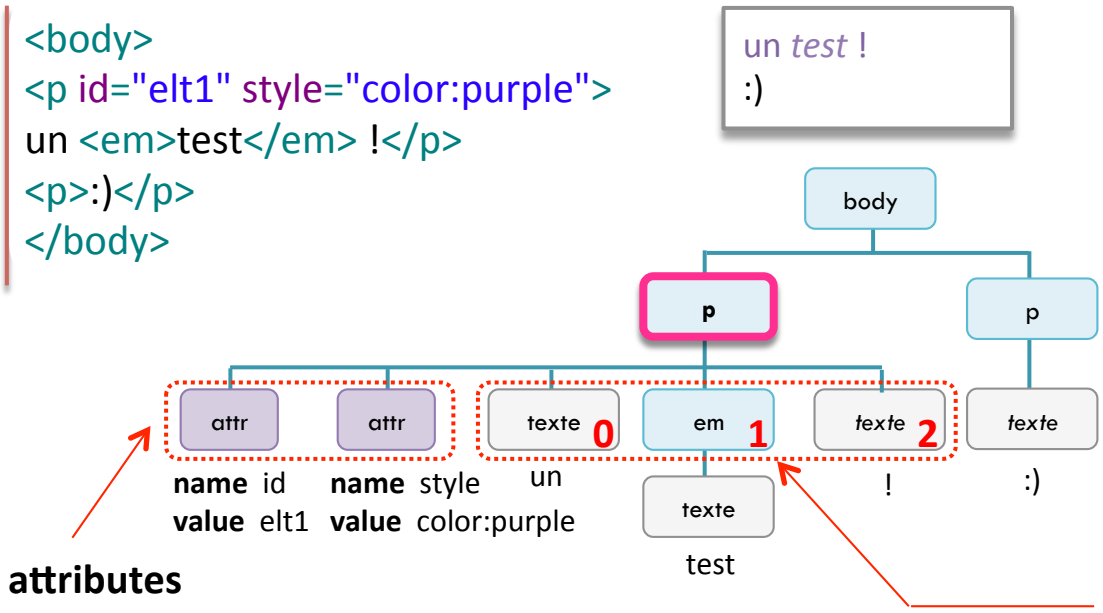
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID document.**getElementById**('*id_element*') -> noeud

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

accès à un nœud : [i] ou item(i)
avec i >= 0

```
var el = document.getElementById('elt1');
alert(el.childNodes[1].nodeName)     // EM
```

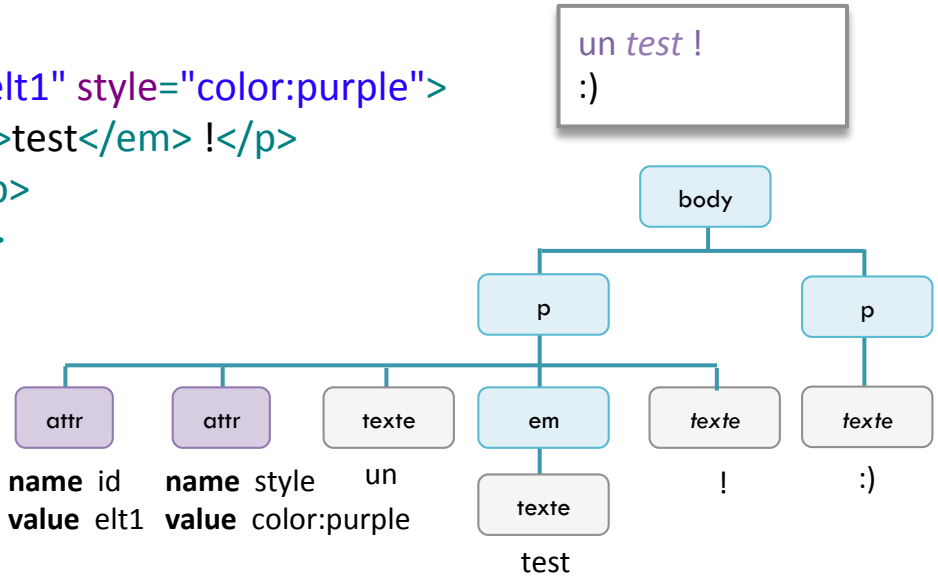
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

nom : getElementsByTagName

accès à un nœud : **[**i**]** ou item**(**i**)**
avec i >= 0

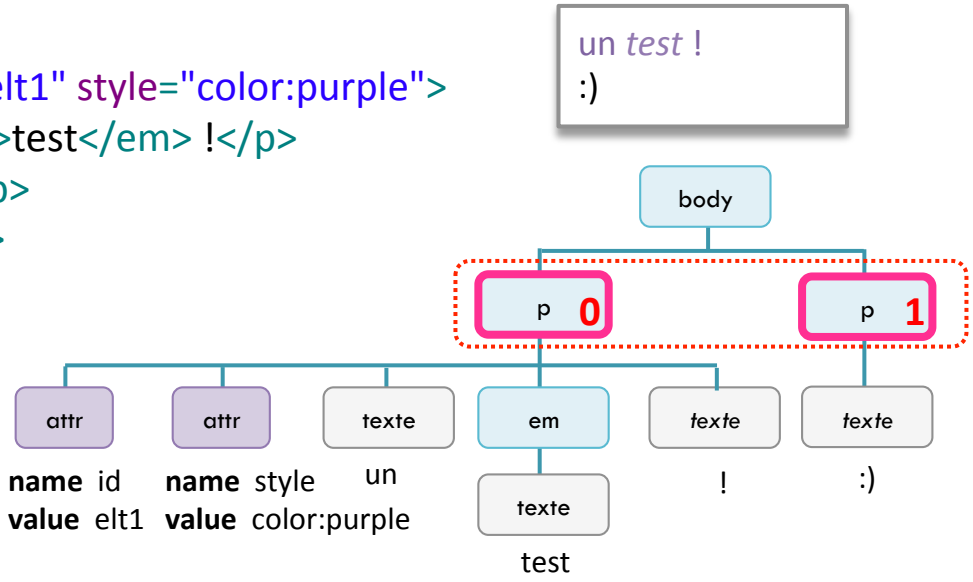
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

nom : getElementsByTagName

accès à un nœud : **[**i**]** ou item**(**i**)**
avec i >= 0

```
var tab_el = document.getElementsByTagName('p');
```

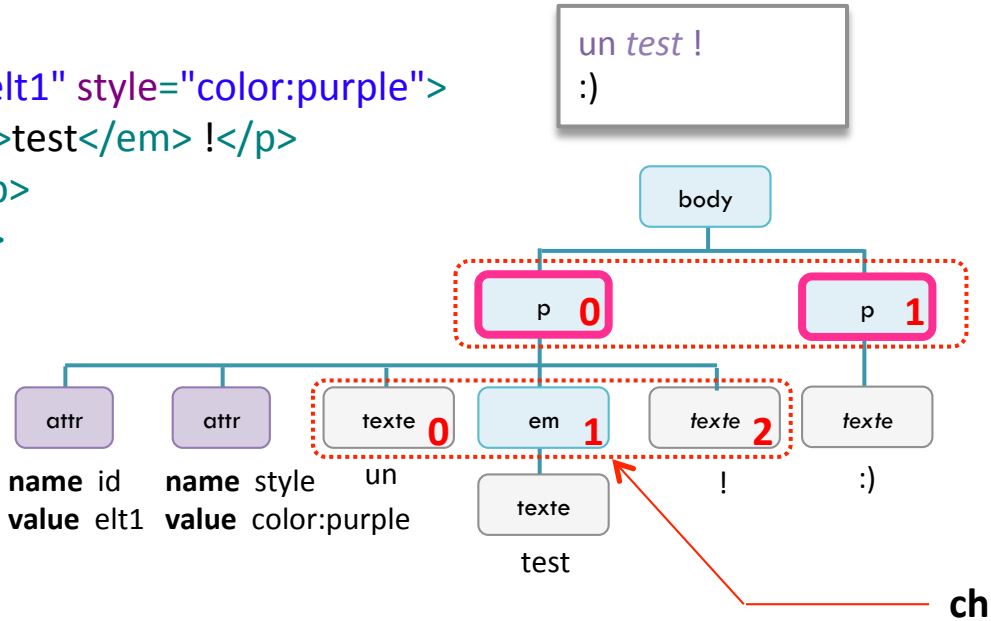
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

nom : getElementsByTagName

accès à un nœud : [i] ou item(i)
avec i >= 0

```
var tab_el = document.getElementsByTagName('p');
alert(tab_el[0].childNodes[1].nodeName);  // EM
```

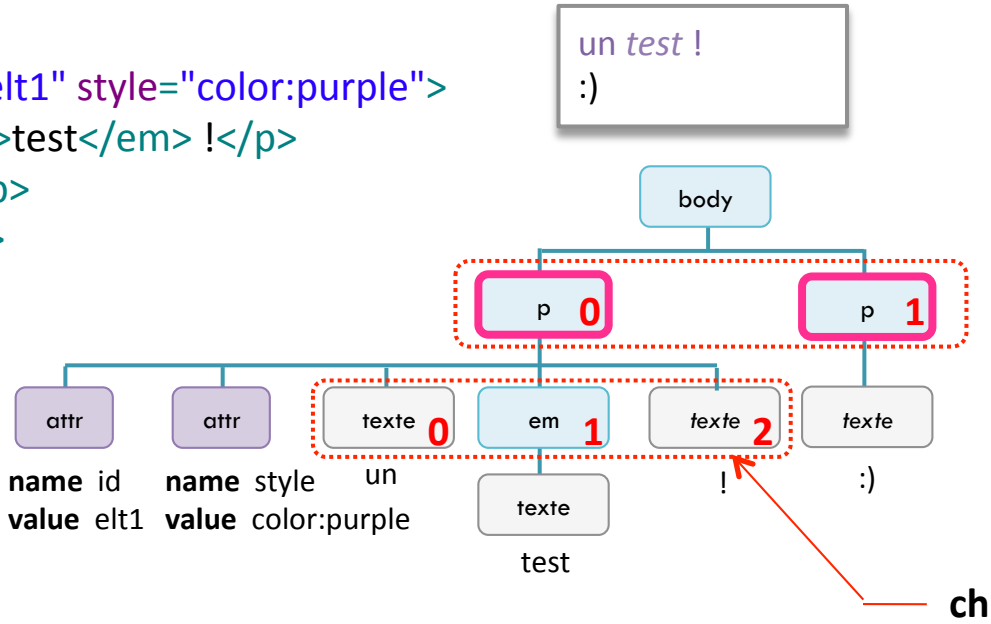
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

nom : getElementsByTagName

accès à un nœud : [i] ou item(i)
avec i >= 0

```
var tab_el = document.getElementsByTagName('p');
alert(tab_el[0].childNodes[2].nodeName);  // #text
```

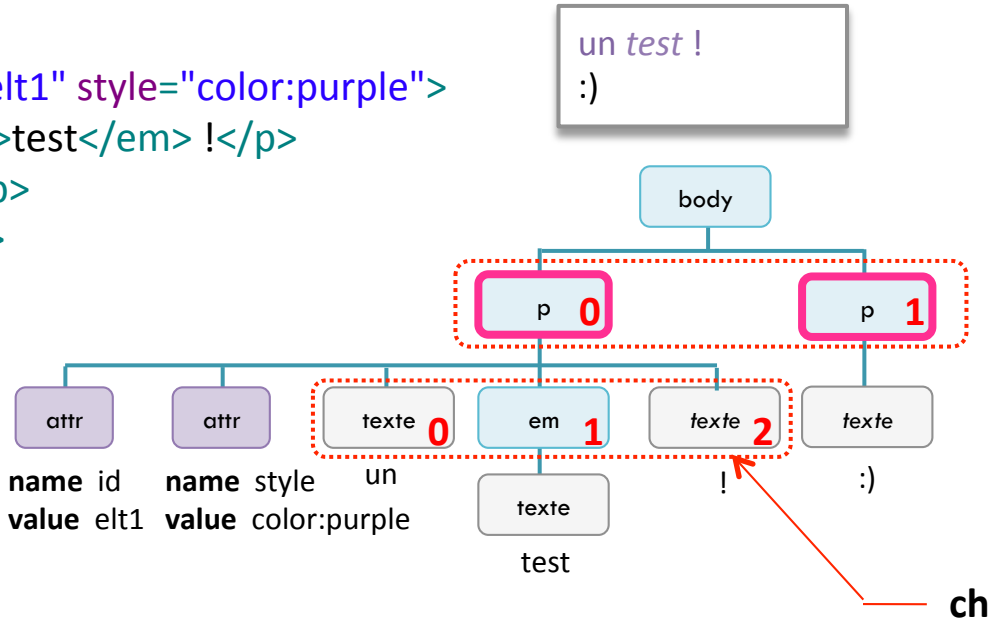
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes

nom : getElementsByTagName

accès à un nœud : [i] ou item(i)
avec i >= 0

```
var tab_el = document.getElementsByTagName('p');
alert(tab_el[0].childNodes[2].data);  // !
```

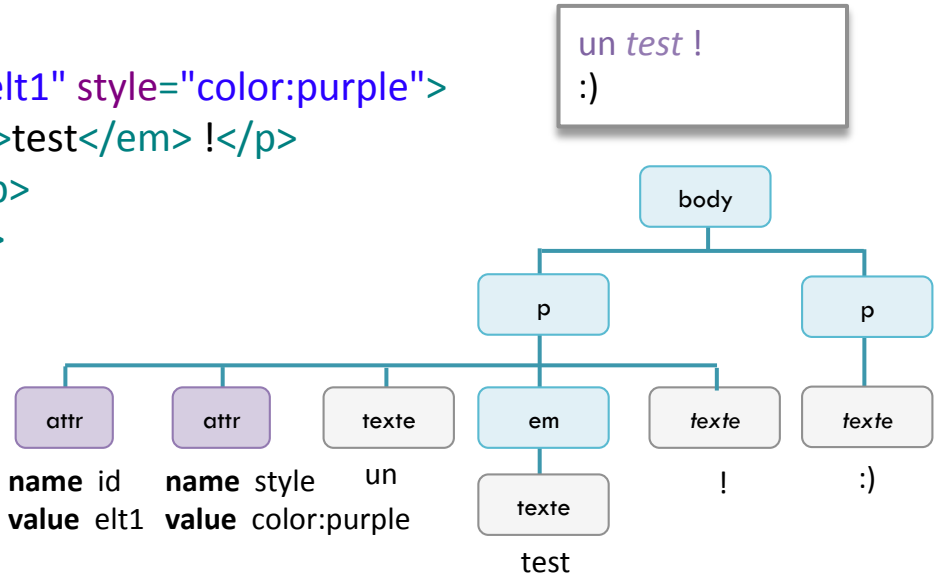
Manipulation des objets du document

DOM – accès aux nœuds

Accès par

ID	document.**getElementById**('*id_element*')	-> noeud
Nom d'élément	document.**getElementsByTagName**('*nom_element*')	-> collection de nœuds
Valeur de l'attribut name	document.**getElementsByName**('*val_attribut*')	-> collection de nœuds

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Collection = tableau de nœuds

prédéfinies : forms, images, links
fils d'un nœud : childNodes
attributs d'un nœud : attributes
attr. name : getElementsByName
nom : getElementsByTagName

accès à un nœud : **[**i**]** ou item**(**i**)**
avec i >= 0

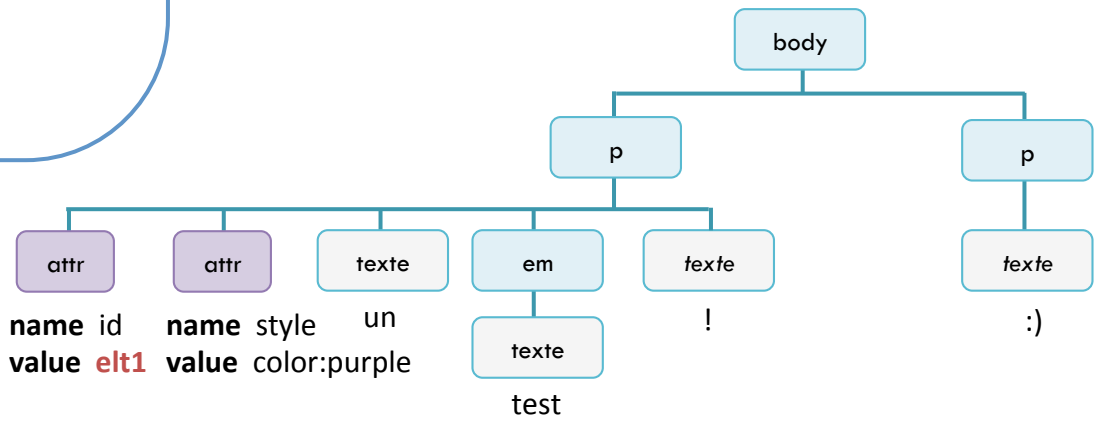
Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```

un *test* !
:)



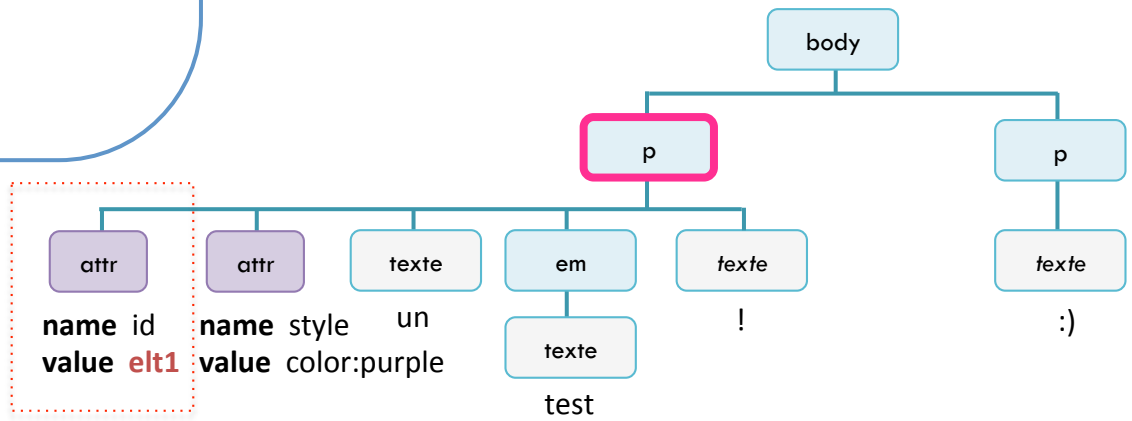
Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

```
var el = document.getElementById('elt1');
```

un *test* !
:)




```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```

Manipulation des objets du document

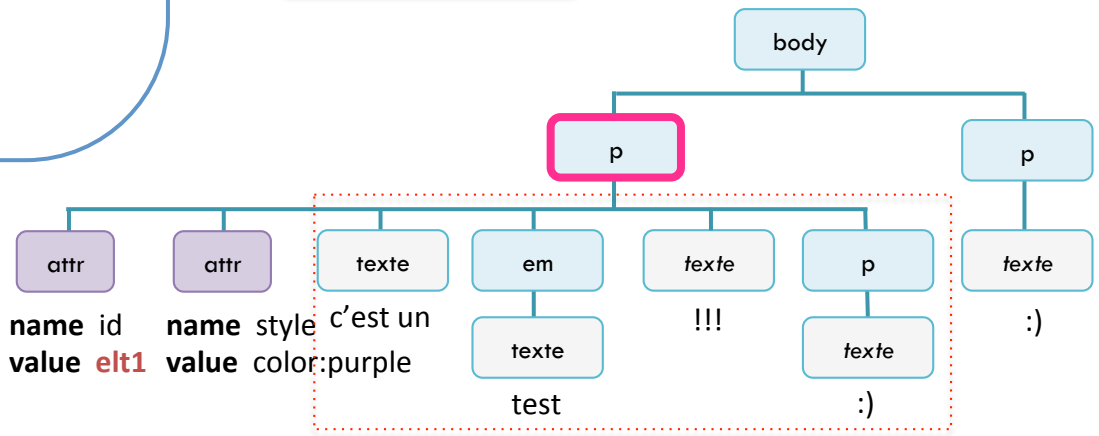
DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

```
var el = document.getElementById('elt1');
el.innerHTML = "c'est un <em>test</em> !!!<p>:)</p>";
```

c'est un *test* !!!
:)
:)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

Contenu

Lire	data
Ajouter	appendData(txt), insertData(i, txt)
Modifier	replaceData(i, j, txt)
Supprimer	deleteData(i, taille)

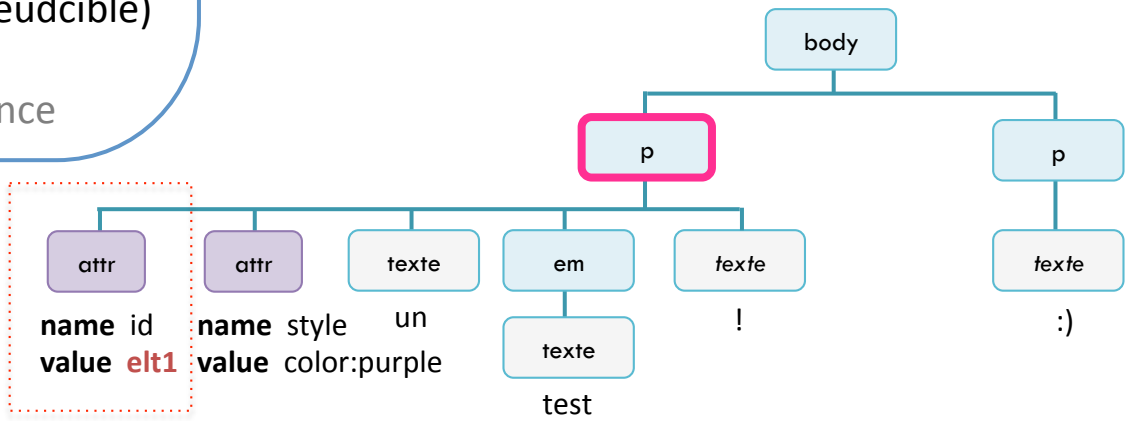
Structure : manipuler les noeuds

Créer	createElement('nom'), createTextNode('texte')
Ajouter	appendChild(nœud) insertBefore(n1,n2)
Supprimer	removeChild(i)
Remplacer	replaceChild(noeudsrc,noeudcible)
Cloner	cloneNode(true) // true = cloner arborescence

```
var el = document.getElementById('elt1');
```

un *test* !
:)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

Contenu

Lire	data
Ajouter	appendData(txt), insertData(i, txt)
Modifier	replaceData(i, j, txt)
Supprimer	deleteData(i, taille)

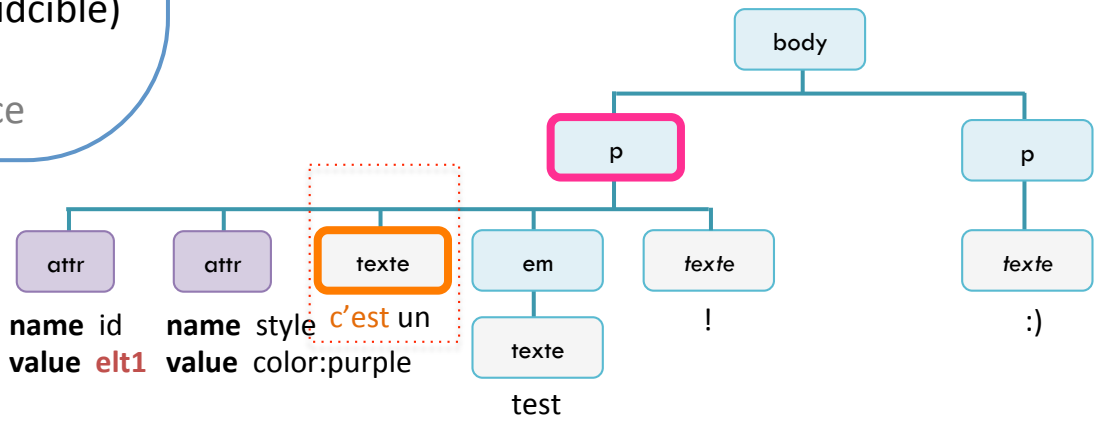
Structure : manipuler les noeuds

Créer	createElement('nom'), createTextNode('texte')
Ajouter	appendChild(nœud) insertBefore(n1,n2)
Supprimer	removeChild(i)
Remplacer	replaceChild(noeudsrc,noeudcible)
Cloner	cloneNode(true) // true = cloner arborescence

```
var el = document.getElementById('elt1');
el.firstChild.insertData(0, "c'est ");
```

c'est un *test* !
:)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

Contenu

Lire	data
Ajouter	appendData(txt), insertData(i, txt)
Modifier	replaceData(i, j, txt)
Supprimer	deleteData(i, taille)

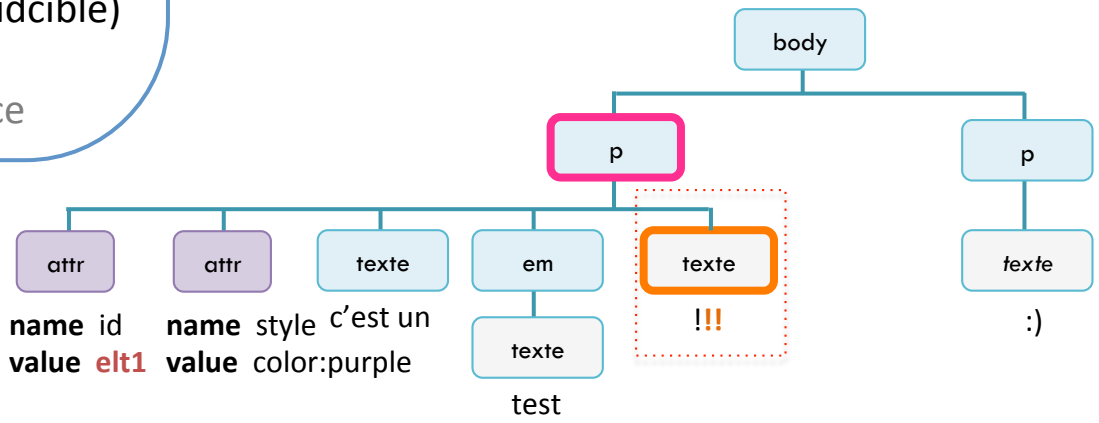
Structure : manipuler les noeuds

Créer	createElement('nom'), createTextNode('texte')
Ajouter	appendChild(nœud) insertBefore(n1,n2)
Supprimer	removeChild(i)
Remplacer	replaceChild(noeudsrc,noeudcible)
Cloner	cloneNode(true) // true = cloner arborescence

```
var el = document.getElementById('elt1');
el.firstChild.insertData(0, "c'est ");
el.lastChild.appendData("!!");
```

c'est un *test* !!!
:)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

Contenu

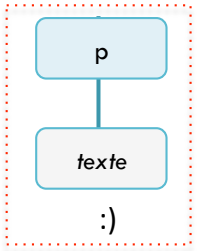
Lire	data
Ajouter	appendData(txt), insertData(i, txt)
Modifier	replaceData(i, j, txt)
Supprimer	deleteData(i, taille)

Structure : manipuler les noeuds

Créer	createElement('nom'), createTextNode('texte')
Ajouter	appendChild(nœud) insertBefore(n1,n2)
Supprimer	removeChild(i)
Remplacer	replaceChild(noeudsrc,noeudcible)
Cloner	cloneNode(true) // true = cloner arborescence

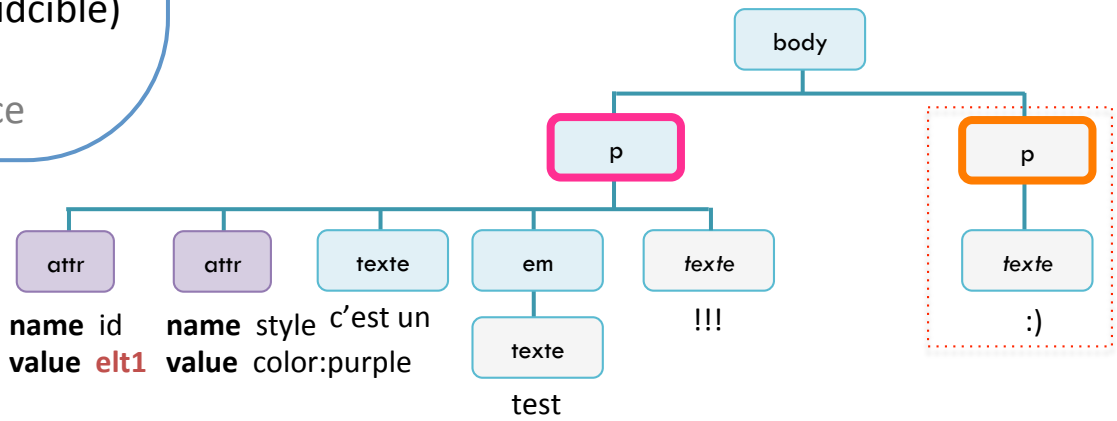
```
var el = document.getElementById('elt1');
el.firstChild.insertData(0, "c'est ");
el.lastChild.appendData("!!");
node = el.nextSibling.cloneNode(true);
```

node



c'est un *test* !!!
:)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier la structure et le contenu

DOM 0 innerHTML, innerText

Contenu

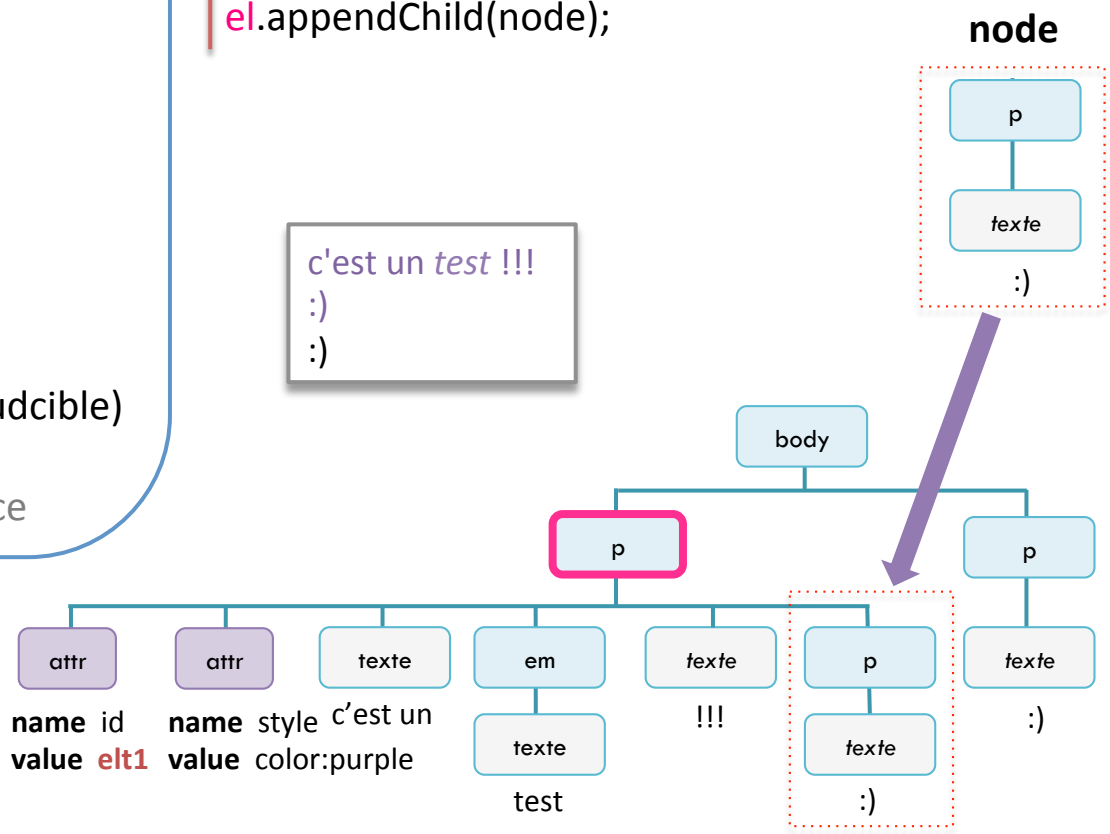
Lire	data
Ajouter	appendData(txt), insertData(i, txt)
Modifier	replaceData(i, j, txt)
Supprimer	deleteData(i, taille)

Structure : manipuler les noeuds

Créer	createElement('nom'), createTextNode('texte')
Ajouter	appendChild(nœud) insertBefore(n1,n2)
Supprimer	removeChild(i)
Remplacer	replaceChild(noeudsrc,noeudcible)
Cloner	cloneNode(true) // true = cloner arborescence

```
var el = document.getElementById('elt1');
el.firstChild.insertData(0, "c'est ");
el.lastChild.appendData("!!");
node = el.nextSibling.cloneNode(true);
el.appendChild(node);
```

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



Manipulation des objets du document

DOM – modifier le style avec JS & CSS

Modifier le style

Propriété CSS :

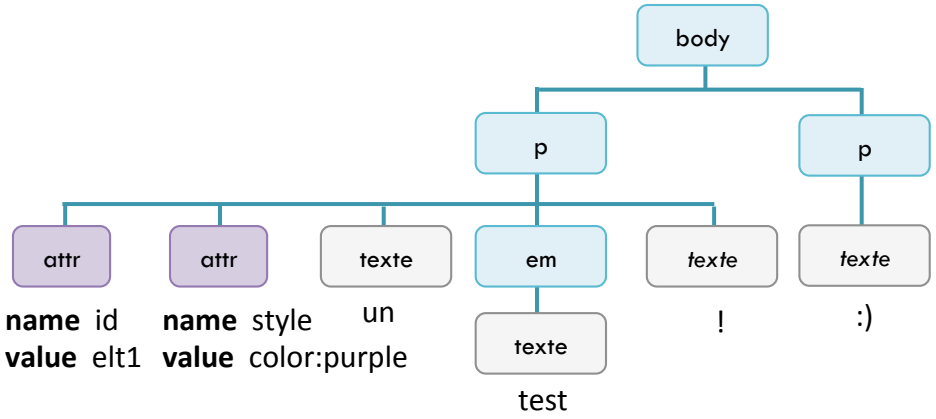
node.**style**.propriete = 'valeur';

Classe de style :

node.**className** = 'nom_classe';

Possibilité de modifier/ajouter/supprimer des règles (accès à la feuille de style CSS)

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



un *test* !
:)

Manipulation des objets du document

DOM – modifier le style avec JS & CSS

Modifier le style

Propriété CSS :

node.**style**.propriete = 'valeur';

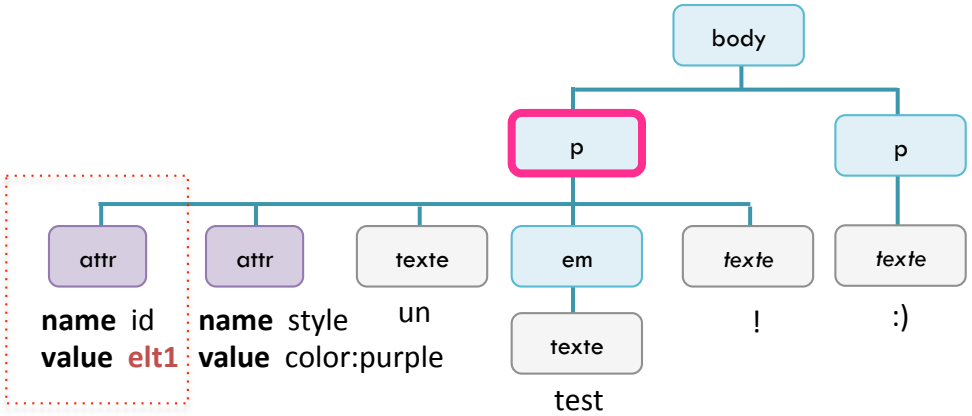
Classe de style :

node.**className** = 'nom_classe';

Possibilité de modifier/ajouter/supprimer des règles (accès à la feuille de style CSS)

```
var el = document.getElementById('elt1');
```

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



un *test* !
:)

Manipulation des objets du document

DOM – modifier le style avec JS & CSS

Modifier le style

Propriété CSS :

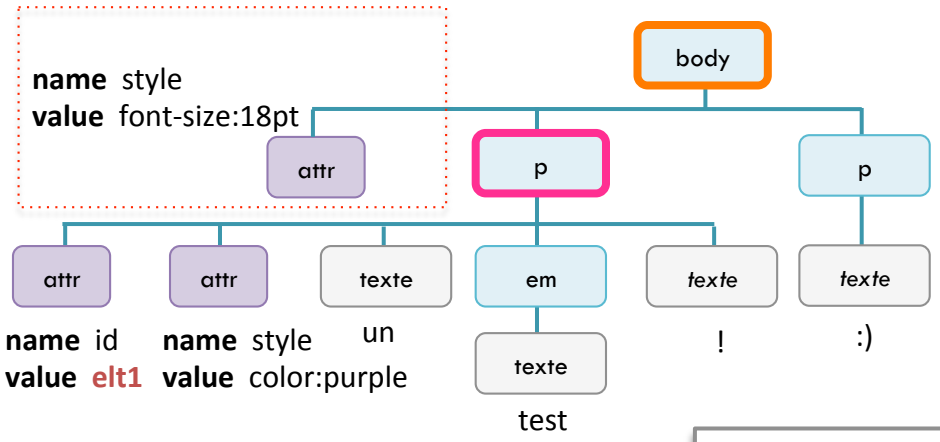
node.**style**.propriete = 'valeur';

Classe de style :

node.**className** = 'nom_classe';

Possibilité de modifier/ajouter/supprimer des règles (accès à la feuille de style CSS)

```
var el = document.getElementById('elt1');
el.parentNode.style.fontSize = '18pt';
```




```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```

un *test* !

:)

Manipulation des objets du document

DOM – modifier le style avec JS & CSS

Modifier le style

Propriété CSS :

node.**style**.propriete = 'valeur';

Classe de style :

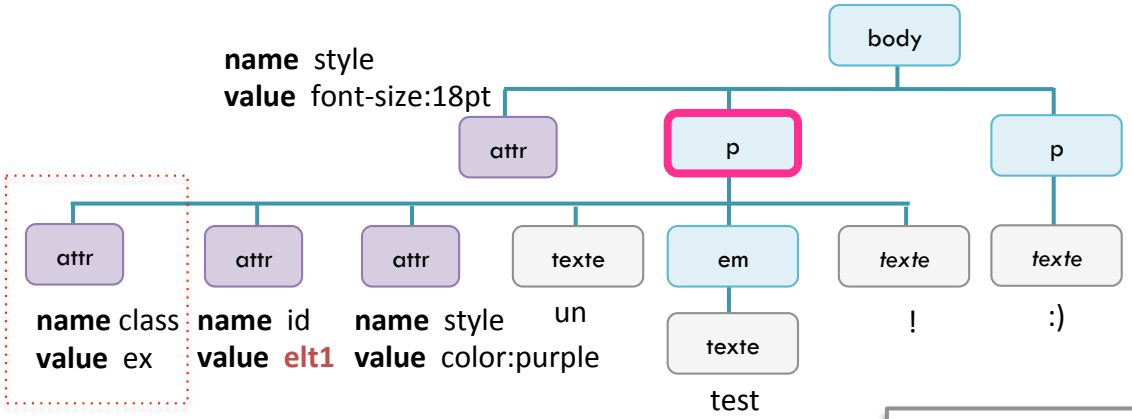
node.**className** = 'nom_classe';

Possibilité de modifier/ajouter/supprimer des règles (accès à la feuille de style CSS)

```
var el = document.getElementById('elt1');
el.parentNode.style.fontSize = '18pt';
el.className = 'ex';
```

```
<style type="text/css">
.ex {
   border:2px dotted silver;
}
</style>
```

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



un *test* !

:)

Manipulation des objets du document

DOM – modifier le style avec JS & CSS

Modifier le style

Propriété CSS :

node.**style**.propriete = 'valeur';

Classe de style :

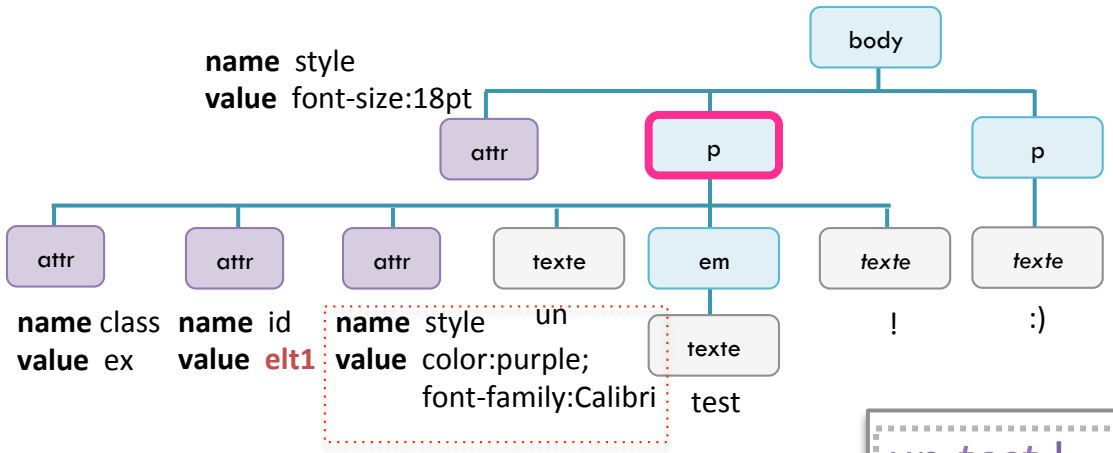
node.**className** = 'nom_classe';

Possibilité de modifier/ajouter/supprimer des règles (accès à la feuille de style CSS)

```
var el = document.getElementById('elt1');
el.parentNode.style.fontSize = '18pt';
el.className = 'ex';
el.style.fontFamily = 'Calibri';
```

```
<style type="text/css">
.ex {
   border:2px dotted silver;
}
</style>
```

```
<body>
<p id="elt1" style="color:purple">
un <em>test</em> !</p>
<p>:)</p>
</body>
```



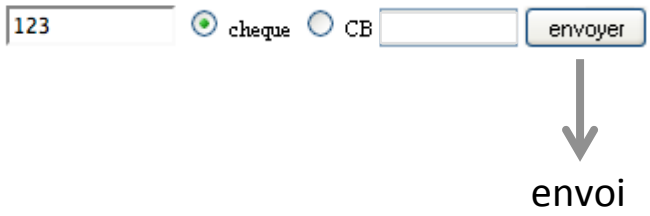
un *test* !

:)

Manipulation des objets du document

DOM – manipuler les formulaires

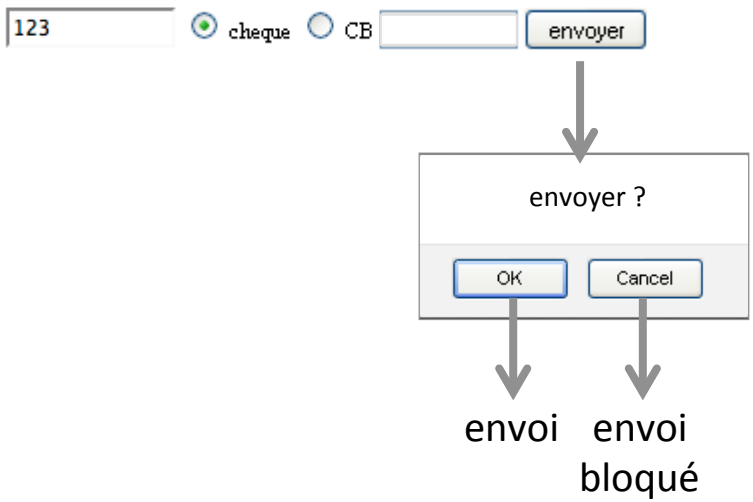
```
<form action='test.php'>
<div>
<input  type="text" name="nb">
<input  type="radio" name="choix" value="1">
        cheque
<input  type="radio" name="choix" value="2">
        CB
<input  type="text" name="cb">
<input  type="submit" value="envoyer">
</div>
</form>
```



Manipulation des objets du document

DOM – manipuler les formulaires

```
<form action='test.php' onsubmit="return confirm('envoyer?')">
<div>
<input  type="text" name="nb">
<input  type="radio" name="choix" value="1">
        cheque
<input  type="radio" name="choix" value="2">
        CB
<input  type="text" name="cb">
<input  type="submit" value="envoyer">
</div>
</form>
```



Evénements

onsubmit, onreset, onclick, onchange, onblur, onfocus, onselect

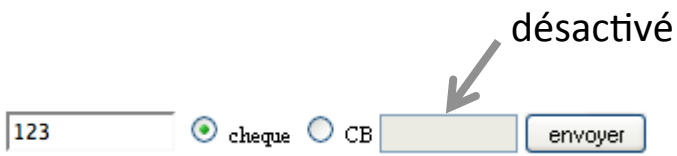
Bloquer un événement

return false;

Manipulation des objets du document

DOM – manipuler les formulaires

```
<form action='test.php' onsubmit="return confirm('envoyer?')">
<div>
<input  type="text" name="nb">
<input  type="radio" name="choix" value="1"
        onclick="cb.disabled=true"> cheque
<input  type="radio" name="choix" value="2"
        onclick="cb.disabled=false"> CB
<input  type="text" name="cb">
<input  type="submit" value="envoyer">
</div>
</form>
```



Evénements

onsubmit, onreset, onclick, onchange, onblur, onfocus, onselect

Bloquer un événement

return false;

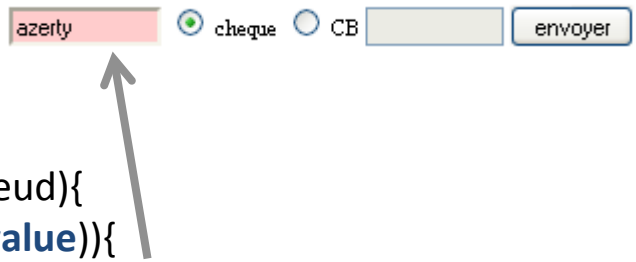
Activer/désactiver des champs

propriétés disabled et readOnly

Manipulation des objets du document

DOM – manipuler les formulaires

```
<form action='test.php' onsubmit="return confirm('envoyer?')">
<div>
<input  type="text" name="nb" onchange="verifNb(this)">
<input  type="radio" name="choix" value="1"
        onclick="cb.disabled=true"> cheque
<input  type="radio" name="choix" value="2"
        onclick="cb.disabled=false"> CB
<input  type="text" name="cb">
<input  type="submit" value="envoyer">
</div>
</form>
```




```
function verifNb(noeud){
    if (isNaN(noeud.value)){
        noeud.className='erreur';
    } else {
        noeud.className='correct';
     }
}
```

```
.erreur { background-color:#FFCCCC }
.correct { background-color:white }
```

propriétés - champs textes

(text, hidden, password, textarea, boutons submit, reset, button)

value defaultValue

propriétés - radio / cases à cocher

value checked

propriétés - listes de choix

selectedIndex options (collection)

prop. options : selected, value, text

Evénements

onsubmit, onreset, onclick, onchange, onblur, onfocus, onselect

Bloquer un événement

return false;

Activer/désactiver des champs

propriétés disabled et readOnly

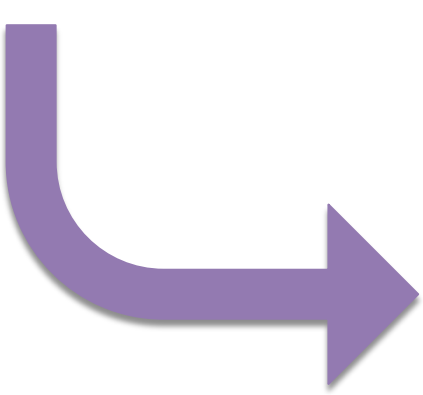
Plan

- Présentation de JavaScript
- ECMAScript : le noyau du langage JavaScript
- Manipulation des objets du document
  - Des débuts du dynamisme au DOM
  - Modifier une page web avec JS, DOM et CSS
  - Formulaires
- **JavaScript discret**

JavaScript discret (*unobstrusive*)

- Séparation structure / **comportement**
- But : faciliter la maintenance

```
<form action='test.php' onsubmit="return confirm('envoyer?')">
<div>
<input  type="text" name="nb" onchange="verifNb(this)">
<input  type="submit" value="envoyer">
</div>
</form>
```




```
<form action='test.php' id="formtest">
 <div>
  <input   type="text" name="nb" id="nb">
  <input   type="submit" value="envoyer">
 </div>
</form>
```

JavaScript discret (*unobstrusive*)

```
function verifNb(){
        if (isNaN(this.value)){
                this.className='erreur';
        } else {
                this.className='correct';
        }
}
function verifForm(e){
        if (confirm('envoyer?')){
                return true;
        } else {
                e.preventDefault(); // bloquer le comportement par defaut
        }
}
// definir les comportements
function comportements(){
        document.getElementById('nb').addEventListener('change', verifNb, false);
        document.getElementById('formtest').addEventListener('submit', verifForm, false);
}
// charger les comportements une fois que toute la page a été chargée
document.addEventListener('DOMContentLoaded', comportements, false);
```

```
<form action='test.php' id="formtest">
 <div>
  <input   type="text" name="nb" id="nb">
  <input   type="submit" value="envoyer">
 </div>
</form>
```

aaa

123

objet concerné par l'événement

envoyer

type d'événement **fonction**

Liens

- Spécifications
  - ECMAScript http://www.ecma-international.org/publications/standards/Ecma-262.htm
  - DOM Level 2 http://www.w3.org/DOM/DOMTR.html
  - JSON http://www.ietf.org/rfc/rfc4627.txt
- Sécurité
  - OWASP (Open Web Application Security Project)
    - http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
    - http://www.owasp.org/index.php/Category:OWASP_Guide_Project
  - CWE / SANS Top 25 Most Dangerous Software Errors 2011

    http://cwe.mitre.org/top25

Questions ?