DOM et ECMAScript

Nicolas Delaforge
Février 2010

What for ?

- DOM : Document Object Model
  - Recommandation W3C
  - Modélisation objet
  - Nombreuses implémentations (plus ou moins complètes)
  - Standard de la manipulation dynamique de contenu XML/HTML

- ECMAScript : Langage d'interaction
  - ECMA (European Computer Manufacturers Association) est un groupe de standardisation européen.
  - ECMAScript est poussé par Adobe/Microsoft/Yahoo/Mozilla
    - Standard utilisé pour Jscript, Javascript et ActionScript.
    - JSON comme standard
    - Projet Tamarin : machine virtuelle ECMAScript incluse dans Flash et Firefox
    - ECMAScript 4 : base pour Javascript 2 et ActionScript 3
  - http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-262.pdf

DOM Level 1

- **Core :** Interfaces de bas-niveau qui permettent de représenter la structure arborescente de n'importe quel document XML.

- **HTML :** Interfaces de haut niveau, basées sur Core Level 1 pour la manipulation simplifiée des documents HTML.

- Parmi les interfaces introduites dans le DOM 1:
  - Document : racine du DOM
  - Node : Noeud de l'arbre
  - Attr : Node de type 2
  - Element : Node de type 1
  - Text : Node de type 3

Node, element, attribute ?

- Un nœud XML est découpé de la manière suivante :
  - `<div id='fake_id'>ceci n'est pas du texte</div>`

Nœud	Type	CODE
`div`	*element*	1
`id='fake_id'`	*attribute*	2
`ceci n'est pas du texte`	*text*	3

- Il existe d'autres nodeTypes pour les commentaires, les scripts, les entités, etc…
  - voir http://www.w3schools.com/Dom/dom_nodetype.asp

Quelques trucs à savoir

- Attention au « mixed content »

```
1. <people>
2. La liste suivante reprend le nom de certaines personnalités :
3. Un <person name="Napoleon Bonaparte" gender="male">empereur</person> bien connu.
4. Ici une <person name="Cleopatra" gender="female">reine d'Egypte</person>
5. Un <person name="Julius Caesar" gender="male">empereur romain</person>
6. Un <person name="Ferdinand Magellan" gender="male">explorateur du XVIe</person>
7. Une <person name="Laura Secord" gender="female">héroïne canadienne de 1812</person>.
8. </people>
```

  - Ce document XML est valide.
    - Il contient 27 nœuds (6 noeuds *element*,10 nœuds *attribute*, 11 nœuds *text*).
- Attention aux espaces blancs
  - pris en compte dans XML
  - pas dans HTML

DOM Level 2 (1/2)

- **Core :**
  - extensions des fonctionnalités du DOM Core 1.
    - Ajoute des interfaces dédiées au XML notamment le fameux *getElementById* et les fonctions de gestion des namespaces.

- **Views :**
  - permet à des programmes et à des scripts d'accéder et de mettre à jour dynamiquement le contenu de la représentation d'un document.
    - Les interfaces introduites sont *AbstractView* et *DocumentView*.
- **Events :**
  - Ajoute un système évènementiel générique pour les programmes et les scripts.
  - Introduction des concepts de flux d'évènements (event flow), de phases (capture, bubbling, cancellation).
  - Il inclue notamment les méthodes *addEventListener* and *handleEvent*.
    - Interfaces pour la gestion des évènements :
      - EventTarget, EventListener, Event, DocumentEvent, MouseEvent, MutationEvent, etc.
  - Cependant, il n'inclue pas d'interface pour la gestion du clavier

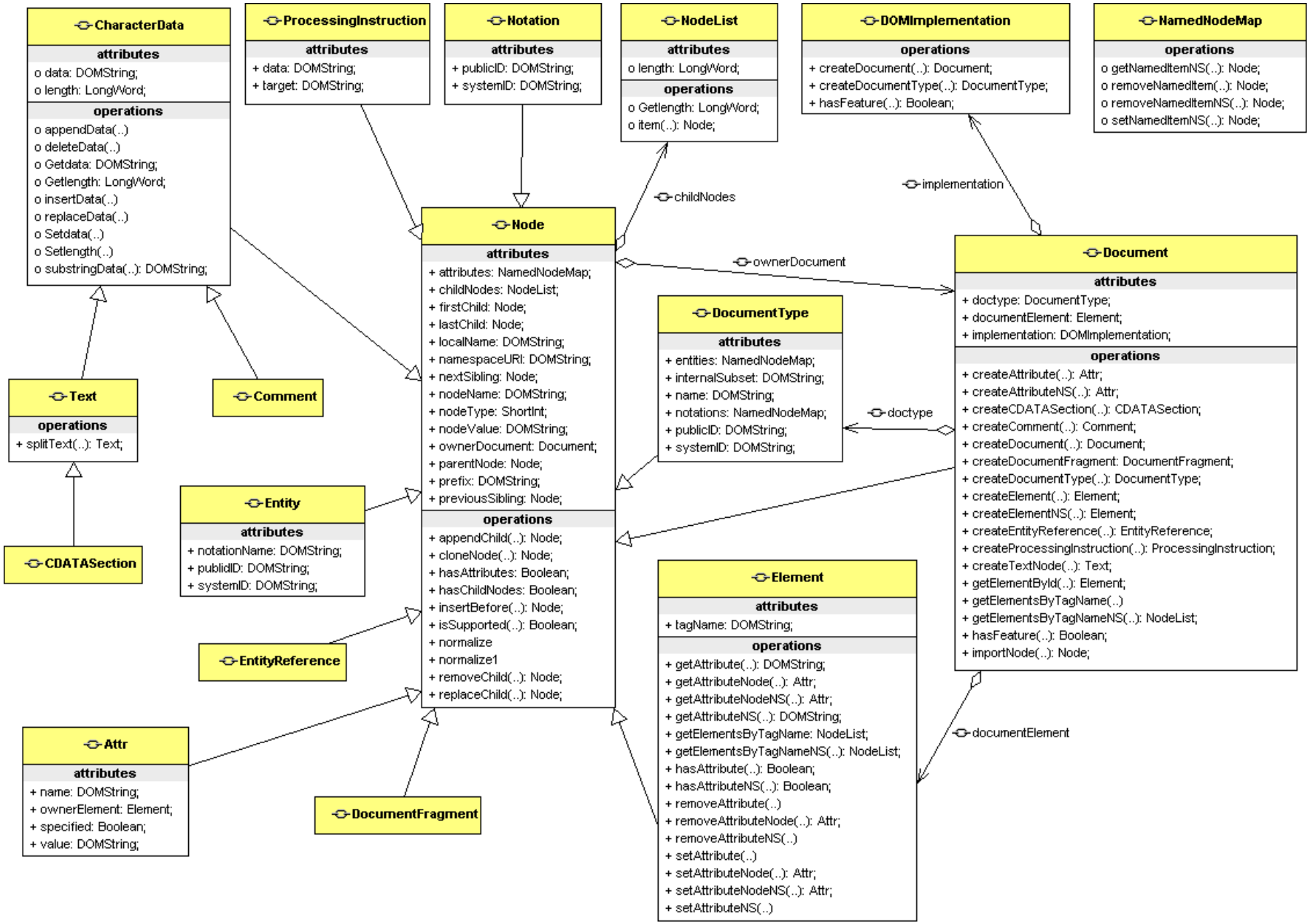
DOM Level 2 (2/2)

- **Style (CSS) :**
  - Mise à jour dynamique du style des éléments.

- **Traversal & Range :**
  - DOM Traversal propose de nouveaux moyens pour accéder au contenu de l'arbre DOM.
  - Le DOM Range permet la création, l'insertion, la modification et la suppression d'une plage de contenu dans un document. La sélection est spécifiée par tous les contenus compris entre deux points.
    - Les interfaces principales ajoutées sont : NodeIterator et TreeWalker
    - Les interfaces définies par le DOM Range sont utilisées essentiellement pour gérer précisément la sélection de texte dans un document.

- **HTML :**
  - permet aux programmes et aux scripts d'accéder et de mettre à jour dynamiquement le contenu et la structure des documents HTM. Il étend les interfaces définies dans le DOM HTML Level 1 en utilisant les possibilités du DOM2 Core.
  - Il introduit la propriété *contentDocument* pour accéder au document contenu dans une frame.

Diagrammes de classes DOM Core Level 2



DOM Views

- Les vues du DOM sont les représentations mémoires sur laquelle s'appliquent les méthodes de modification dynamique.
- Deux interfaces ont été définies par le W3C :
  - `AbstractView` : interface de haut niveau
  - `DocumentView`
- L'objet `document` accessible en javascript correspond au `DocumentView` du navigateur.
- `DocumentView` intègre les règles CSS et les Frames.
- Certaines extensions permettent d'observer directement le `DocumentView` du navigateur
  - *Voir DOM Inspector.*

DOM Traversal (1/2)

- NodeIterator
  - Créé une collection de nœuds DOM sélectionnés selon un filtre passé en paramètre.
  - Méthodes du NodeIterator :
    - `nextNode()`
    - `previousNode()`

- TreeWalker
  - Objet dédié au parcours d'un arbre DOM sur lequel on applique un filtre.
  - Contrairement au NodeIterator, le TreeWalker respecte la hiérarchie des nœuds.
  - Méthodes du TreeWalker :
    - `parentNode();`
    - `firstChild();`
    - `lastChild();`
    - `previousSibling();`
    - `nextSibling();`
    - `previousNode();`
    - `nextNode();`

- http://www.w3.org/TR/DOM-Level-2-Traversal-Range/traversal.html

DOM Traversal (2/2) : Exemples

- TreeWalker

```
1.  var treeWalker = document.createTreeWalker(
2.    document.body,
3.    NodeFilter.SHOW_ELEMENT,
4.    { acceptNode: function(node) { return NodeFilter.FILTER_ACCEPT; } },
5.    false
6.  );
7.  var nodeList = new Array();
8.  var currentNode;
9.  while (currentNode =  treeWalker.nextNode()) {
10.   nodeList.push(currentNode);
11. }
```

- NodeIterator

```
1.  var nodeIterator = document.createNodeIterator(
2.    document.body,
3.    NodeFilter.SHOW_ELEMENT,
4.    { acceptNode: function(node) { return NodeFilter.FILTER_ACCEPT; } },
5.    false
6.  );
7.  var nodeList = new Array();
8.  var currentNode;
9.  while (currentNode = nodeIterator.nextNode()) {
10.   nodeList.push(currentNode);
11. }
```

DOM Level 3, quelques goodies…

- **Core :**
  - Extension des fonctionnalités du DOM Core 1 et 2.
    - Les nouvelles méthodes et propriétés inclues : *adoptNode(), strictErrorChecking, textContent,...*
- **Load & Save :**
  - Permet le chargement dynamique d'un document XML dans un DOM et la sérialisation du DOM dans un document XML.
- **Validation :**
  - Permet de contrôler dynamiquement la validité XML du document.
- **Events :**
  - Extension du DOM2 Events.
  - Ajoute la gestion du clavier aux évènements du DOM2.
- **Xpath :**
  - Ajoute des méthodes simple pour l'accès au DOM en utilisant Xpath.

DOM Events 1/4

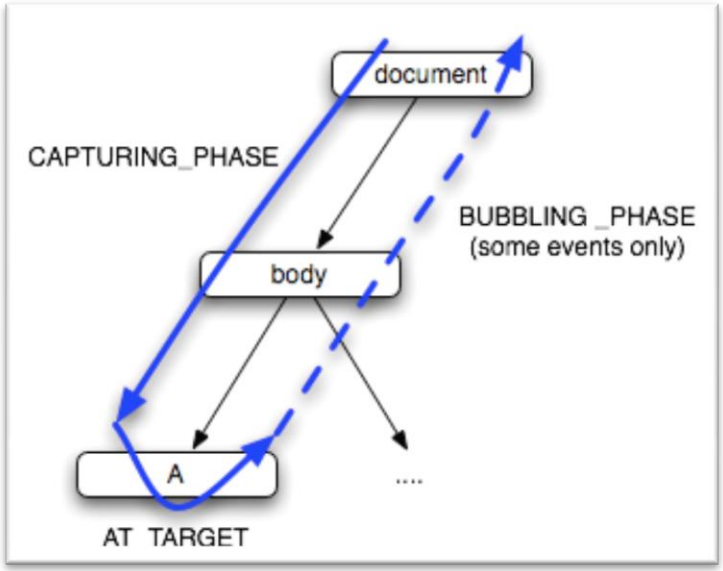
- Les évènements dans le DOM servent à gérer les interactions.
  - Evènements clavier/souris.
  - Gestion du focus
  - Création d'évènements personnalisés.
- Certains listeners d'évènements sont natifs dans HTML
  - Tous les évènements « onmouse… », « onkey… »
  - Listeners possibles sur la modification du DOM.
  - Les plus utilisés : click, dblclick, mouseover, mousemove, keypress, keyup, load…
    - http://en.wikipedia.org/wiki/DOM_events
    - https://developer.mozilla.org/en/DOM/event

DOM Events 2/4

- Propriétés du DOMEvent
  - Permet de connaître le contexte de l'évènement
    - Coordonnées de la souris (`event.clientX, event.clientY, event.screenX, event.screenY`)
    - Bouton de la souris enfoncé (`event.button`)
    - *modifiers* enfoncées pendant l'évènement clavier (`event.ctrlKey…`)
  - Selon le contexte, l'objet *event* est toujours passé en paramètre des méthodes de listener.
    - `<div onclick="alert(event.button)">…</div>`

DOM Events 3/4

- La propagation des évènements dans le DOM



- Trois phases :
  - **Capture** : l'évènement part de la racine (document) et descend l'arborescence jusqu'à sa cible.
  - **Target** : l'évènement a atteint sa cible
  - **Bubble** : l'évènement remonte jusqu'à la racine.

DOM Events 4/4

- Exemples :
  - S'abonner à un évènement

```
1. target.addEventListener(type, listener, useCapture)
```

  - Intercepter un évènement et arrêter sa propagation

```
1. event.preventDefault() ou event.stopPropagation()
```

  - Création d'un évènement personnalisé

```
1. var event = document.createEvent(type);
2. element.dispatchEvent(event);
```

Sérialisation XML et Parsing DOM

- Bien que les fonctions « Load & Save » soient dans la recommandation DOM Level 3, elles ne sont presque jamais implémentées.
  - Firefox propose deux objets non standards :
    - DOMParser

```
1. var str='<people>…</people>';
2. var parser = new DOMParser();
3. var dom = parser.parseFromString(str, "text/xml");
```

    - XMLSerializer

```
1. var serializer = new XMLSerializer();
2. var xml = serializer.serializeToString(document);
```

Ajax et Web 2.0

- AJAX : Asynchronous Javascript And XML
  - Chaque élément d'une page peut communiquer avec un serveur séparément du reste de la page.
  - Implémentations différentes selon les navigateurs
    - XmlHttpRequest dans ECMAScript
    - Msxml2.XMLHTTP dans IE
      - Voir http://fr.wikipedia.org/wiki/XMLHttpRequest
  - Nombreuses librairies JS qui simplifient la vie
    - Jquery
    - Prototype
    - ExtJS
    - Dojo
    - Qooxdoo…

Exemple : Requête AJAX

```
1.  if(window.XMLHttpRequest) {
2.      var xhr=new XMLHttpRequest();
3.      if(xhr!=null){
4.              xhr.onreadystatechange=callback;
5.              xhr.open("GET",
6.                      "http://localhost:9998/tag/all",
7.                      true);
8.              xhr.send(null);
9.      }
10. }
11. function callback(){
12.     if(xhr.readyState==4) {  // readystate : 4="loaded"
13.             if(xhr.status==200){ // status HTTP 200 = OK
14.                     alert(xhr.responseText);
15.             }
16.             else{…}
17.     }
18. }
```

XPath

- Langage de requête dans un arbre XML
- Standard W3C
  - Syntaxe
    - Voir le tutoriel : http://jerome.developpez.com/xmlxsl/xpath/
  - Attention aux namespaces !!
- Evaluer une expression XPath en Javascript :

```
var xpathResult = document.evaluate(
      xpathExpression,
      contextNode,
      namespaceResolver,
      resultType,
      result
);
```

Pointeurs utiles

Tutoriels	DOM	http://www.javascriptkit.com/javatutors/
	DOM	http://www.toutjavascript.com/savoir/
	Parsing XML	https://developer.mozilla.org/en/Parsing_and_serializing_XML
	RegExp	http://www.commentcamarche.net/contents/javascript/jsregexp.php3
	RegExp	http://www.toutjavascript.com/savoir/savoir22.php3
	Xpath	http://jerome.developpez.com/xmlxsl/xpath/
	POO en Javascript	http://t-templier.developpez.com/tutoriel/javascript/javascript-poo1/
	POO en Javascript	http://t-templier.developpez.com/tutoriel/javascript/javascript-poo2/
	POO en Javascript	http://t-templier.developpez.com/tutoriel/javascript/javascript-poo3/
Référence		
	CSS	http://www.w3schools.com/CSS/CSS_reference.asp
	Javascript	http://www.toutjavascript.com/reference/
	Gecko DOM Reference	https://developer.mozilla.org/en/Gecko_DOM_Reference
Outils		
	Console Javascript Firefox	chrome://global/content/console.xul
	Firebug : extension Firefox	https://addons.mozilla.org/fr/firefox/addon/1843
	Evaluateur XPath	http://b-cage.net/code/web/xpath-evaluator.html

ANNEXES

Goodies et bonnes pratiques en développement web

Séparation fond/forme

- XHTML : Mauvais standard ?
  - Mauvaise granularité…
  - Tendance à tout mélanger
    - `<style>#warning{ background-color:red;}</style>`
    - ```
      <script type='text/javascript'>
            document.write('Hello world!');
      </script>
      ```
    - `<div style='background-color:red'>…</div>`
  - Les bonnes pratiques
    - Externaliser les scripts dans des fichiers JS et utiliser des objets
    - Externaliser les CSS et utiliser @import
    - Utiliser les classes et id HTML

      `<div id='warning'>…</div>` et
      `#warning{ background-color:red;}`


      ```

RegExp en Javascript

- Javascript implémente les expressions rationnelles
  - L'objet RegExp :

    ```
    var reg = new RegExp(String motif[, String type]);
    ```

    - Méthodes de l'objet RegExp :

      compile() (Modifie le motif d'une expression régulière)
      exec() (Retourne la première sous-chaîne correspondant au motif)
      test() (Teste l'expression régulière sur une chaîne)
  - Autres méthodes de String basées sur des RegExp :
    - Match
    - Split
    - Replace

POO en Javascript

- POO par prototype
- Cycle de vie des objets lié à la page HTML
- Les « objets » sont des tableaux associatifs

```
var obj = new Object(); ou var obj = {};
obj["attribut"] = "valeur1";
obj["methode"]  = function(param1, param2){ … };
alert(obj.attribut);
obj.methode("v1", "valeur2");
```