Web (Persistance)

Andrea G. B. Tettamanzi
Université de Nice Sophia Antipolis
Département Informatique
andrea.tettamanzi@unice.fr

CM - Séance 11

Introduction à JavaScript

Plan

- Dans cette séance
  - Éléments du langage JavaScript
  - JavaScript et HTML
  - Objets, prototypes et héritage
  - Objets standard et gestion des erreurs
- Dans la prochaine séance
  - Document Object Model
  - Ajax
  - Exemple d'encapsulation d'Ajax

Histoire

- Créé en 1995 par Brendan Eich pour Netscape
- En décembre 1995 Sun et Netscape annoncent sa sortie
- En mars 1996, moteur JavaScript dans NS Navigator 2.0
- Microsoft réagit en développant Jscript, en août dans IE
- En 1997, JavaScript devient un standard ECMA : ECMAScript
- Choix du nom
  - Des raisons de marketing
  - Complément à Java
  - Confusion auprès du publique

Vue d'ensemble du langage

- Orienté objet, mais basé sur les prototypes. Pas de classes
- Un programme est un regroupement d'objets communicants
- Objet = collection de propriétés
- Valeurs primitives : undefined | null | Boolean | Number | String
- Collection d'objets intégrés :
  - Objet global
  - Object, Function, Array, String, Boolean, Number, Math, Date, RegExp, JSON
  - Un certain nombre d'objets erreur : Error, EvalError, …
- Opérateurs : arithmétiques, relationnels, logiques, etc.
- Syntaxe intentionnellement similaire à Java, mais moins stricte

Types et Expressions

- JavaScript est un langage au typage faible et dynamique
- Le type d'une expression est déterminé pas son résultat
  - Nombres : entiers ou à virgule
  - Chaînes de caractères
  - Booléens (false, true)
  - « null » un type à une seule valeur = absence de données
- Expressions simples : formées par un seul élément
  - Littéral
  - Identifiant d'une variable
  - Le mot-clé « this »
- Expressions composées : avec opérateurs / fonctions / méthodes

Instructions

Le standard ECMAScript prévoit 15 types différents d'instructions :

- Block
- VariableStatement
- EmptyStatement
- ExpressionStatement
- IfStatement
- IterationStatement
- ContinueStatement
- BreakStatement
- ReturnStatement
- WithStatement
- LabelledStatement
- SwitchStatement
- ThrowStatement
- TryStatement
- DebuggerStatement

Structures de contrôle

- Les mêmes que Java – c'est fait exprès !

- Conditionnelles
  - if(*cond*) … [else …]
  - switch(*expr*) { case … : … break ; … default : … }
- Itératives
  - while(cond) …
  - do ... while(*cond*) ;
  - for(*instr* ; *cond* ; *instr*) …
- Définition de fonction (c'est une « lambda »)
  - function(*arguments*) { … }

JavaScript et HTML

- JavaScript interagit avec sa pages HTML via :
  - La balise <script>
  - Les fonctions qui font apparaître des boîtes-éclair
  - Les événements
  - Le modèle orienté objet de documents (DOM)
  - L'API de HTML5
    - Stockage local
    - Géolocalisation
    - Glisser-déposer
    - Web Sockets

La balise <script>

- Le code JavaScript est attaché à un document HTML avec la balise <script>, qui peut être placée n'importe où
- Il y a deux manières de faire cela (en HTML5) :
  - script « embarqué » (c'est-à-dire, le code fait partie du document HTML),
    <script> *code embarqué* </script>

    par exemple :
    <script>document.write("Hello World!")</script>
  - script externe (c'est-à-dire, le code est dans un fichier séparé),
    <script src="*URL du fichier* "> </script>
- En HTML 4, type obligatoire : <script type="text/javascript">

Les boîtes-éclair

- JavaScript peut faire apparaître trois types de boîte-éclair
- Boîte avertissement
  - alert(*message*)
  - Affiche une boîte-éclair avec le message et un bouton [OK]
- Boîte confirmation
  - confirm(*message*) → true | false
  - Affiche une boîte-éclair avec le message et [Cancel] [OK]
- Boîte invite
  - prompt(*message*, *texte_par_défaut*) → saisie utilisateur
  - Affiche le message, une case invite initialisée avec le texte par défaut dans laquelle l'utilisateur peut écrire et les boutons [Cancel] et [OK]

Les événements

- Les événements HTML sont des actions de l'utilisateur susceptibles de donner lieu à une interaction
- Par exemple : le clic de souris, les mouvements de la sourie, etc.
- On peut associer des fonctions JavaScript à des événements
- L'association d'un événement HTML à une fonction se fait par le biais des *gestionnaires d'événements*

- Exemple :

```
<button type="button" onclick="calc_input(1)">1</button>
```

- Chaque balise accepte un certain nombre d'événements

Objets

- Un objet JavaScript peut être pensé comme un ensemble de couples (propriété, valeur) que l'on peut représenter avec la notation intuitive

  propriété : valeur

- qui est aussi utilisé par la syntaxe du langage pour définir des objets littéraux
- Propriétés
  - Propres : apparaissent explicitement dans un objet
  - Héritées : ne font pas explicitement partie de l'objet, mais sont dérivées implicitement d'autres objets (dits prototypes)
- Une propriété peut prendre des valeurs de n'importe quel type
- Les objets JavaScript sont des tables de hachage

Méthodes

- Une propriété peut avoir une fonction comme valeur
- On parle alors d'une méthode
- Lorsqu'une méthode d'un objet est appelée, la variable spéciale « this » passe à référencer l'objet
- De ce fait, le code contenu dans la méthode peut accéder aux propriétés de l'objet à travers de la variable « this »

Création d'objets

- Un objet peut être créé de plusieurs façons, parmi lesquelles
  - Syntaxiquement, à l'aide d'une notation littérale :

    obj = { x : 5, y : 10 } ;
  - Par des constructeurs
    - Un constructeur, en JavaScript, est une fonction qui est appelée avec l'opérateur « new »

      new f(…) ;
    - Renvoie un nouvel objet moulé sur l'objet f.prototype

Prototypes et héritage

- Chaque objet a un lien interne vers son prototype
- Un prototype est un objet et peut avoir un prototype à son tour
- On parle alors de chaîne de prototypes d'un objet
- Étant donné un objet o, la valeur o.x d'une propriété x est déterminée comme suit :
  - si x est une propriété propre de o, o.x renvoie la valeur de la propriété x en o ;
  - sinon, la chaîne des prototypes de o est parcourue jusqu'au premier prototype p où x est définie ; dans ce cas, o.x renvoie la valeur de p.x ;
  - si, en parcourant la chaîne des prototypes, l'objet « null » est atteint, o.x renvoie « undefined », la valeur indéfinie.

Objets intégrés

- Un certain nombre d'objets intégrés sont disponibles lors qu'un programme JavaScript est exécuté.
- L'objet global fait directement partie de l'environnement lexical du programme
  - Il contient les variables et le fonctions « globales »
- Les autres objets intégrés sont accessibles comme propriétés initiales de l'objet global
- Beaucoup des objets intégrés sont des fonctions
  - Ils peuvent être appelés avec des arguments
  - Certains sont, en outre, des constructeurs
- Les noms de ces objets s'inspirent lourdement aux noms des classes de la plate-forme Java et de leurs méthodes et attributs

L'objet « Object »

- Permet de créer un objet de conversion pour une valeur donnée
- Par défaut, le prototype des objets créés est « null »
- Parmi ses méthodes on trouve
  - create : crée un nouvel objet avec l'objet prototype et les propriétés spécifiées
  - keys : renvoie un tableau contenant les noms de toutes les propriétés énumérables de l'objet donné
  - getPrototypeOf(o) : renvoie le prototype de l'objet o passé comme argument

L'objet « Function »

- Toute function en JavaScript est en réalité un objet qui hérite le prototype de Function
- Comme constructeur Function crée des fonctions à run-time
- Parmi les méthodes des objets créés par Function on peut citer :
  - apply(o [, args]) : applique l'objet fonction comme s'il était une méthode de l'objet o passé comme argument ; les arguments de la fonction doivent être passés comme un objet Array
  - bind : crée une nouvelle fonction qui, lorsqu'elle est appelée, invoque cette fonction comme si c'était une méthode de la valeur fournie
  - call : la même chose que apply, mais avec les arguments passés un par un au lieu que comme un objet Array

L'objet « Array »

- Constructeur de tableaux
- Conteneurs de haut niveau, dont le comportement et les caractéristiques les rapprochent à des listes
- Un tableau peut être créé soit par une expression littérale,
  - [élément 0 , élément 1, . . ., élément n−1]
- soit par une invocation du constructeur Array :
  - new Array(élément 0 , élément 1 , . . ., élément n−1)
  - new Array(n)
- La propriété length d'un tableau contient toujours sa taille

L'objet « String »

- Constructeur de chaînes de caractères
- La propriété length d'une chaîne de caractères contient sa taille
- Des chaînes de caractères peuvent être comparées en utilisant les opérateurs de comparaison <, <=, ==, >=, >, et !=
- L'opérateur de concaténation est +
- De nombreux méthodes utilitaires font partie de cet objet :
  - charAt, concat, contains, endsWith, indexOf, match, replace, search, slice, split, substr, trim, toLowerCase, to UpperCase, …

L'objet « Math »

- Un objet qui possède des propriétés et des méthodes qui mettent à disposition du programmeur des constantes et des fonctions mathématiques.
- Math n'est pas un constructeur
- Toutes les propriétés et méthodes de Math sont statiques

L'objet « Date »

- Crée des objets qui représentent des dates et des temps
- Permet de manipuler des dates et des temps
  - new Date()
  - new Date(u) : avec l'heure POSIX
  - new Date(chaîne)
  - new Date(A, M , J [, h, m, s, ms])
- L'objet Date fournit trois méthode d'utilité
  - now : renvoie l'heure POSIX actuelle
  - parse : analyse une date en format chaîne de caractères
  - UTC : prend les mêmes arguments de le forme la plus longue du constructeur et renvoie l'objet Date correspondant.

Gestion des erreurs

- Le mécanisme de gestion repose sur trois ingrédients :
  - un ou plusieurs routines de traitement d'exceptions (les handlers) ;
  - un mécanisme de signalement d'exceptions ;
  - un mécanisme qui permet d'associer les exceptions à leurs handlers.
- En JavaScript, le mécanisme d'association des exceptions à leurs handlers est fourni par la structure try … catch … finally
- Les routines de traitement d'exception sont contenues dans les clauses « catch » de cette construction
- Le signalement des exception se fait à l'aide de l'opérateur « throw » et d'objets créés par le constructeur Error.

Merci de votre attention