SCI6373 Programmation documentaire

Cours 12

Été 2025

Plan (1/2)

- Contrôles pour les options - TP3
- Affichage par nouvel onglet (pop-up)
  - La fonction document.write()
- Désaccentuation
- Objets non typés
- JSON

Plan (2/2)

- Gestion d'événements HTML
- JS dans 2 autres environnements courants
- Outils de débogage en navigateur

- Ce plan pourrait nous occuper jusqu'à C13 inclusivement

Contrôles pour les options (1/3)

- Boîte à cocher :

```
<input type="checkbox" id="ex">
```

  - Propriété « checked » dit si la boîte est cochée (valeur booléenne) :
    - Ex.: optionEx = gid('ex').checked;

Contrôles pour les options (2/3)

- Boutons de radio :

```
<input type="radio" id="ch1" name="ex"
  checked="checked">
<input type="radio" id="ch2" name="ex">
```

  - Propriété « checked » dit si ce bouton est sélectionné (valeur booléenne) :
    - Ex.: optionExChoix1 = gid('ch1').checked;
  - Un seul des boutons à la fois est « checked »
  - Le choix *par défaut* est celui qui porte l'attribut checked="checked"

Contrôles pour les options (3/3)

- Boîte combinée (*Combo box*) pour le choix de la couleur

```
<select id="saison">
  <option value="spring">printemps</option>
  <option value="summer">été</option>
  <option value="fall" selected="selected">automne</option>
  <option value="winter">hiver</option>
</select>
```

Onglets "pop-up" (1/3)

- Principe :

```
maFen = open("");// chaîne vide
maFen.document.write(texteHTML1);
maFen.document.write(texteHTML2);
…
maFen.document.close();
```

- *texteHTMLn* est une chaîne qui contient une partie du document HTML à afficher (balisage inclus, y compris éléments html, head, title et body)

Onglets "pop-up" (2/3)

- Un document.write peut être *dans une boucle*
- Les "bouts" sont concaténés dans l'ordre des instructions document.write()
- L'écriture du document peut aussi se faire en une seule étape, exemple:

```
maFen = open("");   // chaîne vide
maFen.document.write(documentHTMLcomplet);
maFen.document.close();
```

Onglets "pop-up" (3/3)

- open() ouvre un *nouvel* onglet
- Pour fermer l'onglet:

  maFen.close(); *ou* close();
- close() ferme l'onglet où le script s'exécute
- maFen.close() ferme un onglet précédemment créé par un script
- Ex. 800, 810 : traduction de saison pop-up

Désaccentuation

- Contrairement aux changements de casse (majuscules / minuscules), la désaccentuation n'est pas une fonction primitive en JS
- Rarement requis, mais parfois justifié
- Il faut donc utiliser une FDP à cet effet
  - Voir implantation : `désac()` dans `bib-6373.js`

Lecture d'un fichier

- Lire un texte à partir d'un fichier - local ou par URL - plutôt que par copier-coller
  - Diverses méthodes possibles
  - Soumises à la politique "same origin"

Objets non typés (1/5)

- On commande une commode toute neuve faite sur mesure
- Avec exactement ce qu'on veut comme tiroirs : nom et contenu

  monObj = { *nom1* : *val1*, *nom2* : *val2*, … }

  - Tiroir *nom1* contient valeur *val1*
  - Tiroir *nom2* contient valeur *val2*
  - etc.

Objets non typés (2/5)

- Objets non typés :
  - {a:*valeur1*, b:*valeur2*, …} représente un objet avec une propriété "a" valant *valeur1*, "b" valant *valeur2*, etc.
  - Ex.:* ({a:123, b:'abc'}).a → 123
    ({a:123, b:'abc'}).b → 'abc'
  - { } crée donc un objet "vide"; ex.: a = { };
  - On peut aussi utiliser "new Object()", ex.: var a = new Object();

*Les parenthèses ne sont pas toujours nécessaires

Objets non typés (3/5)

- On peut consulter et ajouter des propriétés en utilisation l'indexation [ ]; ex.:

```
monObj = { };
monObj['abc'] = 33;
monObj['abc'] → 33
monObj.abc → 33
```

- Avec [ ], noms de propriétés arbitraires:

```
monObj['*%$'] = 'joie';
monObj['*%$'] → 'joie'
```


Objets non typés (4/5)

- Pour tester si une propriété existe dans un objet:

'*nomProp*' in *object*

Cette expression retourne "true" si la propriété '*nomProp*' existe dans l'objet *object*; elle retourne "false" si elle n'existe pas.

Exemple 820

Objets non typés (5/5)

- On peut itérer sur toutes les propriétés d'un objet avec une forme spéciale de boucle "for":

```
objet = navigator; liste = "";
for (nomProp in objet) {
   liste += '\nNom: ' + nomProp + '\nValeur: ' + objet[nomProp];
};
```

- "*p* in *objet*" fait en sorte que la variable *p* prend tour à tour le nom de chaque propriété de l'objet *objet*

En passant : `for (item of tableau)`

- Autre forme spéciale de boucle for pour parcourir un tableau, sans avoir à gérer soi-même un index et son incrémentation
- Convient si le traitement à faire des items ne dépend pas de l'index
- `item` prendra successivement comme valeur chacune des valeurs de `tableau`

JSON (1/2)

- JavaScript Object Notation
- Combinaison de la notation pour les constantes "tableau" et "objet non typé"
- Extrêmement populaire
  - Remplace graduellement le XML pour les *documents orientés données*
    - XML est encore prépondérant pour les documents *orientés information*

JSON (2/2)

- Exemples dans oXygen 830 et 840
- Exemples à la console :
  - .stringify()
  - .parse()

Gestion d’événements HTML (1/2)

- Permet aux scripts de réagir à différents événements d'interaction entre l'utilisateur et la page web
  - survol d'un élément avec la souris
  - clic sur un élément quelconque
  - appuie sur une touche de clavier
  - etc.

Gestion d’événements HTML (2/2)

- Exemple : dossier 815
- Liste d'événements HTML sur <https://www.w3schools.com/>

JS dans deux autres environnements courants

- JavaScript côté serveur
  - Dossier d'exemples 980
- JavaScript en système d'exploitation
  - Exemples (WSH, NodeJS) 990

Outils de débogage de Firefox*

- Ctrl-Maj-K (ou F12) – onglet "Débogueur"
- Points d'arrêt
- Pas-à-pas
- Expressions espionnes

*L'équivalent existe dans tous les navigateurs

API "canvas"

- Atelier à la console
  - Dossier d'exemples 970
- Pour arcs et cercles :
  - ctx.beginPath();
  - ctx.arc(95, 50, 40, 0, 2 * Math.PI);
  - ctx.stroke();