Asynchronous Javascript and XML (AJaX)

Introduction

— Une archive contenant les fichiers mentionnés dans cette fiche est à récupérer sur la liste de diffusion.

Exercice 1 – *Premier essai en AJaX : le BartViewer*

Question 1.1 : Récupérez le contenu du répertoire `bv`, modifiez le fichier `bart-viewer.html` afin que lorsque l'utilisateur clique sur le bouton *afficher*, le contenu du fichier dont l'adresse est donnée dans le champs de saisie soit affiché sur le tableau.
Notes :

— le code d'instanciation de l'objet *XHR* est fourni dans le fichier `ajax.js`.
— vous prendrez soin d'intercepter les exceptions susceptibles d'être déclenchées par l'objet xhr et de les afficher dans un `alert()`.

Question 1.2 : Testez votre application en local (`file://...`) avec le nom d'un fichier en local (`file://...`), par exemple `ajax.js` puis `bart-viewer.html`.
Question 1.3 : Testez maintenant avec l'URL d'un fichier pris sur votre serveur web (`http://localhost/~login/...`). Que se passe-t-il ? Comment s'appelle la politique à l'origine de cette erreur ?
Question 1.4 : Copiez votre répertoire `bv` sur votre serveur web refaites vos tests à partir de votre serveur (`http://localhost/~login/bv/bart-viewer.html`) et testez également des URL de ressources sur d'autres sites Web.

Exercice 2 – *Un formulaire dynamique en AJaX*

Le but de cet exercice est de créer un formulaire avec deux listes déroulantes liées : les choix possibles dans la deuxième liste dépendront du choix effectué dans la première. En l'occurrence, la première liste permettra de choisir une commune parmi celles de l'agglomération de Rouen qui disposent d'une station de métro et la deuxième permettra de choisir une station de métro parmi celles situées dans la commune choisie.

Les fichiers à récupérer sont dans le répertoire `Communes`.
Question 2.1 : Commencez par créer la page HTML contenant le formulaire. Pour la première liste, le contenu de chaque balise option sera le nom de la commune et la valeur de la variable (attribut value) sera un code à deux lettres indiqué par le tableau suivant :

Commune	Code
Le Grand-Quevilly	GQ
Le Petit-Quevilly	PQ
Rouen	RO
Saint-Étienne-du-Rouvray	SE
Sotteville-lès-Rouen	SO

La deuxième liste n'a initialement pas de contenu ; elle sera remplie plus tard par un script JavaScript qu'on va écrire et qui utilisera une requête HTTP. Par la suite, on travaillera dans un fichier JavaScript séparé.
Question 2.2 : Définissez une variable globale qui contiendra un objet XMLHttpRequest et la remplir en utilisant une fonction (voir le cours).
Question 2.3 : Écrivez une fonction qui envoie la requête HTTP. Cette fonction doit être appelée quand on choisit une commune dans le formulaire (événement *onchange* de la première liste déroulante). On utilisera la méthode `GET`, la page de traitement sera *ajax.php* (à télécharger dans le répertoire du TP) et il faudra passer

le code de la commune (à récupérer avec les fonctions DOM) dans l'URL. L'appel se fera en mode asynchrone. Écrire la fonction qui traite la réponse : en cas de succès (*readyState* égal à 4 et *status* égal à 200), on insère dans la deuxième liste déroulante le code HTML renvoyé par le serveur, ce qui nous permet d'obtenir le résultat voulu. C'est cette fonction qui sera appelée lors d'un changement d'état de la requête : il faut affecter, dans la fonction qui envoie la requête, le nom de la fonction qui traite la réponse à l'attribut *onreadystatechange* de l'objet XMLHttpRequest.

Question 2.4 : Reprenez les questions précédentes dans le cas où le résultat est envoyé au format XML par le script `ajax_xml.php` puis dans le cas où il est envoyé au format JSON par le script `ajax_json.php`.

Question 2.5 : Dans la fonction de traitement de la réponse, afficher un texte ou une image (le site `http://www.ajaxload.info` fournit des images de chargement comme on en trouve sur beaucoup de sites utilisant AJAX) quand la requête est en cours de traitement ; on masquera ce texte ou cette image une fois le traitement terminé. Pour voir quelque chose, il est vivement conseillé d'insérer dans le script PHP une ligne de code de la forme `sleep(n);` afin de le rendre inactif pendant n secondes.