TP2 DBLL

JavaScript, DOM, AJAX

Partie 1 : JavaScript et Document Object Model (DOM)

Exercice 1 :

Créer **un projet web dynamique** nommé « **devineJavaScript** ».
Dans le web Content :

- Créer une page **index.html** contenant un bouton « Devine ».
- Créer un script javascript nommé « **javascript .js**».

En javascript :

• Math.random() retourne un nombre aléatoire entre 0 et 1
• Math.round(floatvalue) retourne la partie entière du nombre qu'on lui passe en paramètre.

Dans le fichier « **javascript .js**» :

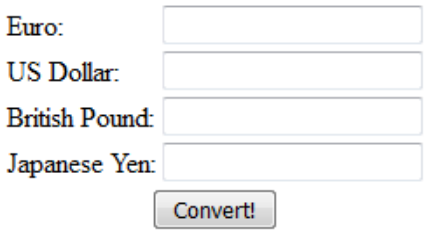
A. Ecrire la fonction aleatoire(n) qui retourne un entier aléatoirement entre 0 et n
B. Ecrire la fonction javascript devine() qui génère un entier aléatoire de 0 à 100 puis demande au visiteur de le deviner. A chacune de ses propositions, une indication « trop grand » ou « trop petit » est fournie à l'internaute. Une fois la valeur trouvée, un message le lui indique ainsi que le nombre d'essais.

Le bouton « Devine » fait appel à la fonction devine().

Exercice 2 :

Ecrire un Javascript permettant de convertir un montant donné en Euro en Dollars, pound et Yen, sachant que :

1 dollar = 1 euro *1.470
1 pound= 1 euro * 0.717
1 Yen = 1 euro * 165.192
La page html à afficher contient 4 champs texte et un bouton « Convert ».



Partie 2 : AJAX

Dans cette partie, un exemple simple est fourni afin de découvrir le principe d'utilisation et de fonctionnement d'AJAX.

Exercice 1 :

Vous allez développer une application web dynamique permettant d'afficher une chaine de caractère (par exemple : Bonjour le monde !) en utilisant les technologies JavaScript, DOM, Servlet, l'objet XMLHTTPRequest et JSP.

- Créer un projet web dynamique nommé « myAjaxApplication ».
- Dans le web Content, créer une page index.JSP qui affiche la phrase suivante : « ***Getting Started with AJAX using JAVA : Bonjour ! »***
- Dans le src, créer une classe java servlet nommée « HelloWorldServlet » qui permet de retourner une chaine de caractère de valeur « Bonjour le monde !».
- Dans le web Content, créer un script JavaScript nommé ajax.js dans lequel trois fonctions sont définis:
    - getXMLHttpRequest() permettant de créer un objet XMLHTTPRequest et tester le type de navigateur.

```
function getXMLHttpRequest() {
	var xmlHttpReq = false;
	// to create XMLHttpRequest object in non-Microsoft browsers
	if (window.XMLHttpRequest) {
		xmlHttpReq = new XMLHttpRequest();
	} else if (window.ActiveXObject) {
		try {
			// to create XMLHttpRequest object in later versions
			// of Internet Explorer
			xmlHttpReq = new ActiveXObject("Msxml2.XMLHTTP");
		} catch (exp1) {
			try {
				// to create XMLHttpRequest object in older versions
				// of Internet Explorer
				xmlHttpReq = new ActiveXObject("Microsoft.XMLHTTP");
			} catch (exp2) {
				xmlHttpReq = false;
			}
		}
	}
	return xmlHttpReq;
}
```

- La fonction getReadyStateHandler(xmlHttpRequest) : Retourne une fonction qui attend le changement d'état dans XMLHTTPRequest.

```
function getReadyStateHandler(xmlHttpRequest) {

	// an anonymous function returned
	// it listens to the XMLHttpRequest instance
	return function() {
		if (xmlHttpRequest.readyState == 4) {
			if (xmlHttpRequest.status == 200) {
				document.getElementById("hello").innerHTML =
xmlHttpRequest.responseText;
			} else {
				alert("HTTP error " + xmlHttpRequest.status + ": " +
xmlHttpRequest.statusText);
			}
		}
	};
}
```

- la fonction makeRequest() : permettant d'effectuer la requête vers le serveur.

```
function makeRequest() {
	var xmlHttpRequest = getXMLHttpRequest();
	xmlHttpRequest.onreadystatechange = getReadyStateHandler(xmlHttpRequest);
	xmlHttpRequest.open("GET", "helloWorld.do", true);
	xmlHttpRequest.send(null);
}
```

Exercice 2 : Afficher la date Actuelle

Développer une application web dynamique Ajax permettant d'afficher la date actuelle de serveur en faisant appel à ***java.util.Date***.

Exercice 3 : Compte rendu

Développer une application web dynamique Ajax (en utilisant JavasScript, XMLHTTPRequest, servlet, JSP) pour faire une calculatrice simple permettant d'effectuer l'addition, la soustraction, la multiplication et la division.

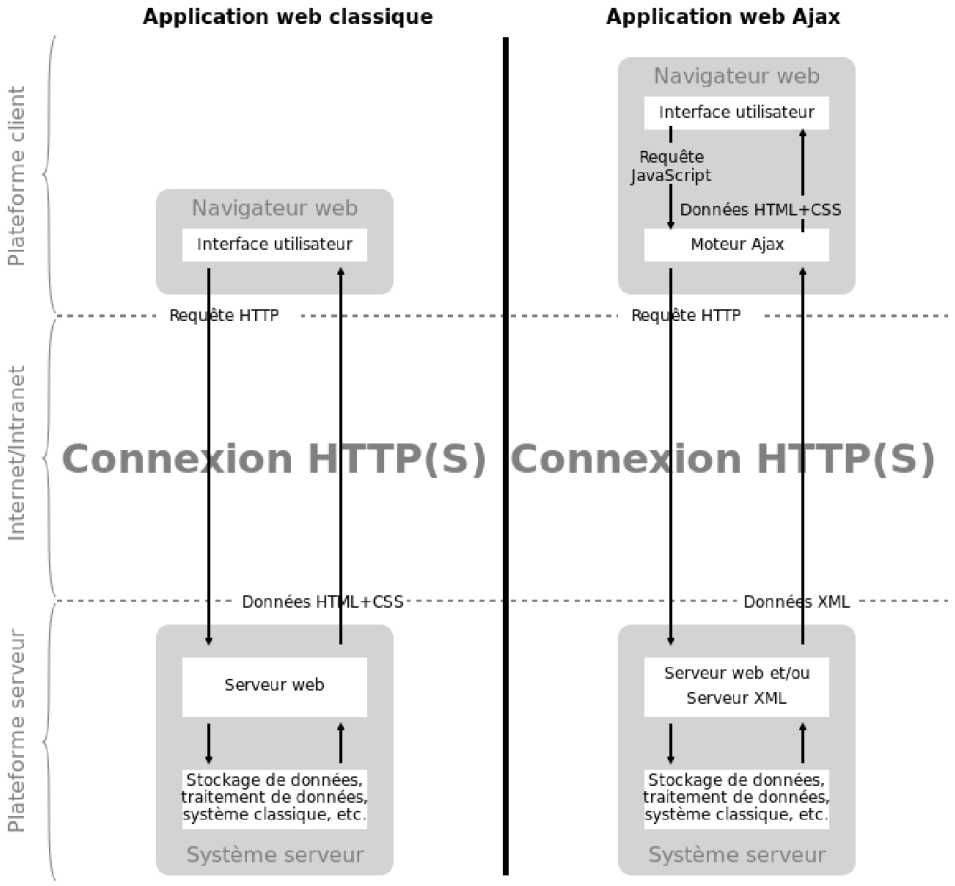
Annexe :

Partie 2 :

Pour plus d'information « http://www.xul.fr/xml-ajax.html#ajax-definition »

Principe et Comparaison avec les applications web traditionnelles

Dans une application Web, la méthode classique de dialogue entre un navigateur et un serveur est la suivante : lors de chaque manipulation faite par l'utilisateur, le navigateur envoie une requête contenant une référence à une page Web, puis le serveur Web effectue des calculs, et envoie le résultat sous forme d'une page Web à destination du navigateur. Celui-ci affichera alors la page qu'il vient de recevoir. Chaque manipulation entraîne la transmission et l'affichage d'une nouvelle page. L'utilisateur doit attendre l'arrivée de la réponse pour effectuer d'autres manipulations.



En utilisant Ajax, le dialogue entre le navigateur et le serveur se déroule la plupart du temps de la

manière suivante : un programme écrit en langage de programmation JavaScript, incorporé dans une page web, est exécuté par le navigateur. Celui-ci envoie en arrière-plan des demandes au serveur Web, puis modifie le contenu de la page actuellement affichée par le navigateur Web en fonction du résultat reçu du serveur, évitant ainsi la transmission et l'affichage d'une nouvelle page complète.

Pourquoi utiliser Ajax?

Ajax permet de modifier partiellement la page affichée par le navigateur pour la mettre à jour sans avoir à recharger la page entière.
Par exemple le contenu d'un champ de formulaire peut être changé, sans avoir à recharger la page avec le titre, les images, le menu, etc.

Ajax est une technique qui fait usage des éléments suivants:

- HTML pour l'interface.
- CSS (Cascading Style-Sheet) pour la présentation de la page.
- JavaScript (EcmaScript) pour les traitements locaux, et DOM (Document Object Model) qui accède aux éléments de la page ou du formulaire ou aux éléments d'un fichier XML chargé sur le serveur.
- L'objet XMLHttpRequest lit des données ou fichiers sur le serveur de façon asynchrone.
- PHP ou un autre langage de scripts peut être utilisé coté serveur.

Le terme "Asynchronous", asynchrone en français, signifie que l'exécution de JavaScript continue sans attendre la réponse du serveur qui sera traitée quand elle arrivera. Tandis qu'en mode synchrone, le navigateur serait gelé en attendant la réponse du serveur.

L'objet XMLHttpRequest

Permet d'interagir avec le serveur, grâce à ses méthodes et ses attributs.

Les attributs :

readyState	le code d'état passe successivement de 0 à 4 qui signifie "prêt".
status	200 est ok 404 si la page n'est pas trouvée.
responseText	contient les données chargées dans une chaîne de caractères.
responseXml	contient les données chargées sous forme XML, les méthodes de DOM servent à les extraire.
onreadystatechange	propriété activée par un évènement de changement d'état. On lui assigne une fonction.

Les méthodes :

open(mode, url, boolean)	mode: type de requête, GET ou POST url: l'endroit ou trouver les données, un fichier avec son chemin sur le disque. boolean: true (asynchrone) / false (synchrone). en option on peut ajouter un login et un mot de passe.
send("chaine")	null pour une commande GET.