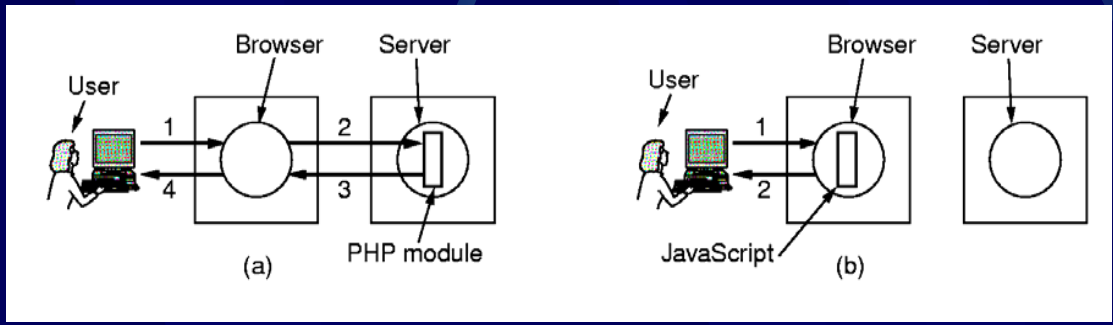
Documents web dynamiques

Documents web dynamiques

- Contenu Statique
  - Le client envoie une requête avec un nom de fichier
  - Le serveur répond en lui retournant le fichier
- Contenu dynamique :
  - Généré à la demande, plutôt que stocké sur disque
  - Génération faite
    - par le client (javascript)
    - ou par le serveur (php, asp, cgi)



Documents web dynamiques côté client

- But : Proposer un contenu actif et interagir directement avec l'utilisateur
  - Autrement que par un clic sur un lien qui fait quitter la page
  - Agrémenter la présentation : menus déroulants, etc
  - Vérification de saisies de formulaires
  - Intégrer des programmes : jeux, traitement de texte

- Scripts imbriqués dans des pages XHTML :
  Balise `<script> </script>`
- Exécution côté client

- Langage de script : javascript (le plus utilisé)
- Autres moyens : applet Java (Sun), contrôle ActiveX (Microsoft), Java, VBScript, Dynamic HTML, Flash …

Documents web dynamiques côté serveur

- But : Générer à la volée un contenu (XHTML) avant de l'envoyer au client. Par exemple :
  - Récupérer les données d'un formulaire
  - Envoyer une requête formulée à partir de ces données à un SGBD
  - Générer une page XHTML contenant la réponse
- Deux approches pour gérer ces pages interactives :
  - Utilisation de scripts CGI :
    - dans un formulaire, le bouton envoyer fait appel à une *URL désignant un fichier exécutable* (script)
    - Langages de script CGI : Perl, Python, C, C++, …
  - Scripts imbriqués dans des pages XHTML :
    - Le suffixe du fichier (.php, .asp) indique au serveur qu'il doit exécuter du code à l'intérieur de la page pour la créer dynamiquement
    - Technologies : PHP, ASP (Active Server Pages) de Microsoft qui utilise le VBScript, JSP (Java Server Pages) de Sun qui utilise Java.

Documents web dynamiques Côté Client

1. Javascript
2. Dom
3. DHTML & Dom API
4. AJAX

Exemple JavaScript

- Exemple 1 : insertion de javascript dans une page html

```
<html>
  <head>
   <script>
          var i=0;
   </script>
  </head>
<body>
  <script>
   document.write("La valeur de i est ",i,".")
  </script>
</body>
```

JavaScript: un survol…

- Langage interprété par le navigateur web (IE,Netscape,Firefox…)
  - Masquage du script pour les anciens navigateurs
- Permet d'intercepter les événements du clavier/souris/… et ainsi de pouvoir réagir aux actions de l'utilisateur
- Code
  - Commentaires : les conventions utilisées en langage C/C++
  - Entre les balises <script> </script> (éventuellement dans un fichier externe)

```
<script type="text/javascript"  src="script.js"> </script>
```

  - Associé à un événement

```
<balise evenement="Code Javascript" … />
```

  Evénement : clic de souris, passage de la souris au-dessus d'une zone, le changement d'une valeur, ...
- Variable,
- Types élémentaires,
- Types structurés (chaîne, tableau (multidimensionnel, associatif), …),

Javascript: un survol…

- Opérateurs : calcul (+ %), affectation (+= %=), incrémentation (++), comparaison (== != <), logiques(|| && !), manipulation de chaînes (+)
- Structures de contrôle : if [else], for, while, switch...case…break
- Fonctions :

```
function Nom_De_La_Fonction(argument1, argument2, ...) {
liste d'instructions
}
```

NE PAS HESITER A CONSULTER LA DOC POUR LA SYNTAXE…

- Déboguer un programme JavaScript
  - D'une façon générale : write ("message"), alert("message")
  - Avec la console JavaScript accessible sous :
    - IE : double-clic sur l'icône l'erreur (en bas à gauche)
    - Firefox :menu outil/console erreur
    - Netscape : menu outil/Développement web/console JavaScript
    - Opera :

Exemple avec une fonction

```
<script type="text/javascript">
<!--  permet de cacher le script pour les anciens
  navigateurs
// Un exemple de fonction
  function square(i)
  {
  document.write("The call passed ", i ," to the
  function.","<br/>")
  return i * i
  }
// une utilisation de cette fonction
  document.write("The function returned
  ",square(5),".")
-->
</script>
```

Gestion des événements

- Permet de réagir aux actions de l'utilisateur
- Un événement est associé à un élément HTML donné

  Exemple 3 : exécuter du code au lancement d'une page.

  Exemple 4 : interagir à l'événement clic sur un bouton.

  ```
  <input type="button" value="Cliquez ici !"
      onclick="window.open('http://www.info.iut.u-bordeaux1.fr')"/>
  ```
- Exemple d'événements :
  - onchange
  - onclick
  - onkeypress
  - onload
  - onmouseover
  - onsubmit
  - etc.
- Tous les événements ne sont pas disponibles sur tous les éléments HTML :
  - Onsubmit peut être intercepté uniquement par l'élément HTML form

DOM: Document Object Model

- Le navigateur crée des objets prédéfinis correspondant :
  - aux pages du navigateur,
  - à la page courante,
  - à l'état du navigateur,
  - etc.
- Une API (DOM) permet la gestion de ces objets
- Plusieurs versions de DOM (appelés niveaux)
  - DOM Niveau 1 pris en charge par tous les navigateurs (que nous traiterons)
  - DOM Niveau 2 pris en charge uniquement par Firefox et Netscape
- Les différents objets :
  - **navigator** : contient des informations sur le navigateur
  - **window** : l'objet où s'affiche la page,
  - **history** : liste de liens qui ont été visités précédemment
  - **document** : les propriétés du document (couleur d'arrière plan, titre, ...)
  - etc.

Objets du navigateur

- Les objets du navigateur sont classés hiérarchiquement
  - **window**
    - **navigator**
    - **screen**
    - **location**
    - **history**
    - **document**
      - **links[]**
      - **images[]**
      - **forms[]**
        - **button**
        - **submit**
        - **select…/…**
- Exemple d'utilisation :

```
document.write('Navigateur:'+navigator.appName
                +'Version:'+navigator.appVersion);
```

DOM API

- Le document a une structure par défaut qui permet d'accéder à ses composants
- DOM level 1 et maintenant (depuis 2000) DOM level 2
- essentiellement utilisé pour construire un arbre logique contenant les informations issues d'un document.
- Un exemple d'utilisation :
  - Pour définir un élément :

```
<div id = "Objet1">Un message fantôme</div>
```

  - Pour obtenir un élément particulier :

```
<div id = "Objet2"
onmouseover="(document.getElementById('Objet1')).style.visibility
   ='visible';"
onmouseout="(document.getElementById('Objet1')).style.visibility
   ='hidden';">
Celui qui est affiché
</div>
```

Dynamic HTML (DHTML)

Ensemble de technologies associées pour fournir des pages plus interactives :

- HTML ou XHTML pour présenter le document ;
- CSS pour définir un style et un positionnement aux objets ;
- Le modèle objet de document (DOM), facilitant leur manipulation ;
- Un langage de script (généralement Javascript), pour définir des actions associées aux événements utilisateur ;

ATTENTION : problème de la compatibilité

- l'interprétation peut être différente d'un navigateur à l'autre,

=> Gros travail nécessaire pour s'assurer que l'effet sera visible sur la majorité des configurations :

- en écrivant un script pour chaque navigateur ;
- en créant (si possible) un script fonctionnant sur les différents navigateurs.

Un petit exemple

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
.ST1 { font-family: Blackadder ITC; color: #008000; font-size: 16pt;
   font-style:italic ;background-color:aqua}
.ST2 { font-size: 15pt; color: #FF00FF;}
</style>
<title>Page d'accueil</title></head>
<body>
<h1>Créer et manipuler des objets</h1>
<div id = "Objet1" class = "ST2" >Un message fantôme</div>
<div id = "Objet2" class = "ST1"
onmouseover = "(document.getElementById('Objet1')).style.visibility =
   'visible';"
onmouseout = "(document.getElementById('Objet1')).style.visibility =
   'hidden';">Celui qui est affiché<br /></div>
</body></html>
```

DEMO

Ajax : Asynchronous Javascript and XML

Principes :

- échanges de données entre client et serveur (HTTP),
- transparents pour l'utilisateur,

But : rafraichissement de certaines parties de la page uniquement.

Avantages :

- interface utilisateur plus conviviale et réactive ;
- moins de données échangées.

(=>Web 2.0)

Ajax : Asynchronous Javascript and XML

Ensemble de technologies associées pour fournir des pages plus interactives :

- Asynchrone : envoi de requêtes HTTP sans attente de la réponse ;
- Javascript : appel distant fait par le navigateur
- XML : contenant des données formatées en XML.

A ces technologies ci-dessus, rajoutons :

- l'objet JavaScript XmlHttpRequest pour l'échange HTTP
- le modèle Document Object Model (DOM+DOM API) pour la gestion d'une partie de la page uniquement ;

Une instance de XmlHTTPRequest

```
var xhr = null;
if (window.XMLHttpRequest) {// Firefox...
  xhr = new XMLHttpRequest();
  }
else {
  if (window.ActiveXObject) {
    try { // Tente de charger l'objet pour IE avant IE7
      xhr = new ActiveXObject("Msxml2.XMLHTTP");
      }
    catch (e) {
      try { // Une autre version de IE…
        xhr = new ActiveXObject("Microsoft.XMLHTTP");
       }
      catch (e) {
         window.alert(« Ne prend pas en charge l'objet XMLHTTPRequest.");
         } // try-catch
      } // try-catch
    }// if
  } // else
```

Un premier exemple Ajax

- Code client :

```
<html><head><title>Premier exemple Ajax sans intérêt...</title>
<script type='text/JavaScript'>
function CreerObjetXhr(){var xhr = null;…/…return xhr;}//CreerObjetXhr
function VasChercherLeTexte (){
  var xhr = CreerObjetXhr()
  // On définit ce qu'on va faire quand on aura la réponse
  xhr.onreadystatechange = function(){
    if(xhr.readyState == 4 && xhr.status == 200){//on a tout reçu & réponse OK
      alert(xhr.responseText);//xhr.responseText réponse retournée format 'texte'
     }
    }
  xhr.open("GET","Texte.php",true);// méthode:GET - url:Texte.php - ASynchrone
  xhr.send(null);//envoie la requête au serveur sans paramètre (null)
  }//VasChercherLeTexte
</script></head>
<body><input type='button' value='Vas chercher le texte !'
    onclick='VasChercherLeTexte()'></body></html>
```

- Code php:

```
<?php echo "Voici le texte envoyé par la page Texte.php.";?>
```