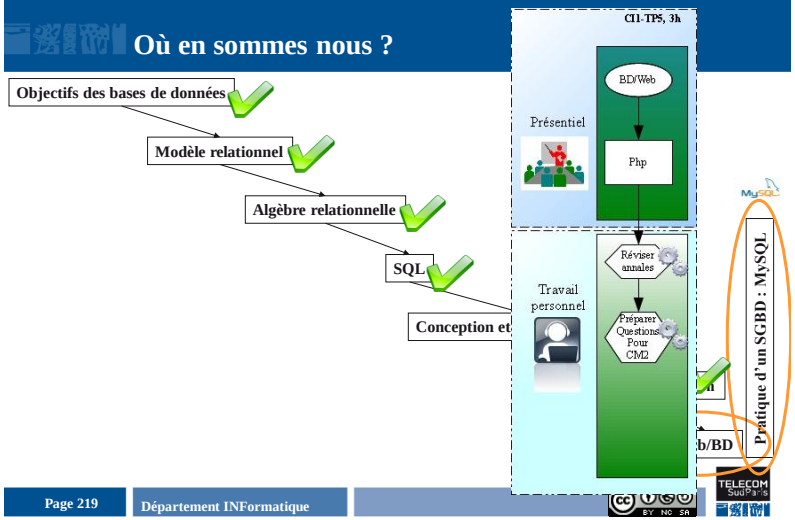
Applications Web et Bases de Données

Département INFormatique

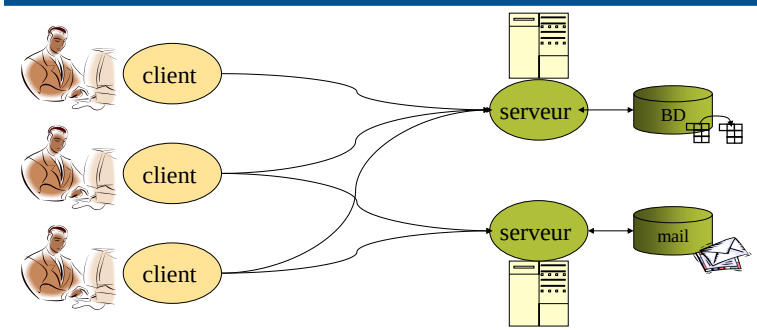
Où en sommes nous ?



Plan du document

- **Client/serveur** **slide 220**
  - Principe client/serveur
  - Critique du client/serveur en architecture à 2 niveaux
  - Architecture N niveaux
- **Web statique** **slide 226**
- **Interfacer les Bases de Données et le Web** **slide 228**
- **Conclusion** **slide 248**
- **Et moi que dois-je faire ?** **slide 254**
- **Annexes**
  - Annexe 1 : le Web statique slide 257
  - Annexe 2 : les formulaires Web slide 267
  - Annexe 3 : les CGIs slide 275
  - Annexe 4 : ADO DB slide 282

Principe client/serveur (1)



Client demande une "ressource" à un serveur ressource = fichier, donnée, traitement, ...

Principe client/serveur (2)

- **Application client/serveur = application dont l'exécution se fait sur plusieurs machines**
- **Exemples : serveur d'applications, de fichiers, de terminaux, de messagerie électronique**
- **Caractéristiques d'un serveur**
  - Passif (ou esclave) ;
  - A l'écoute (prêt à répondre aux requêtes envoyées par des clients) ;
  - Réaction (traitement) et réponse aux requêtes.
- **Caractéristiques d'un client**
  - Actif (ou maître) ;
  - Demandeur de services (envoi de requêtes au serveur) ;
  - En attente des réponses du serveur.
- **Client et serveur : même protocole de communication**
- **Serveur généralement capable de servir plusieurs clients simultanément**

Critique du client/serveur en architecture à 2 niveaux

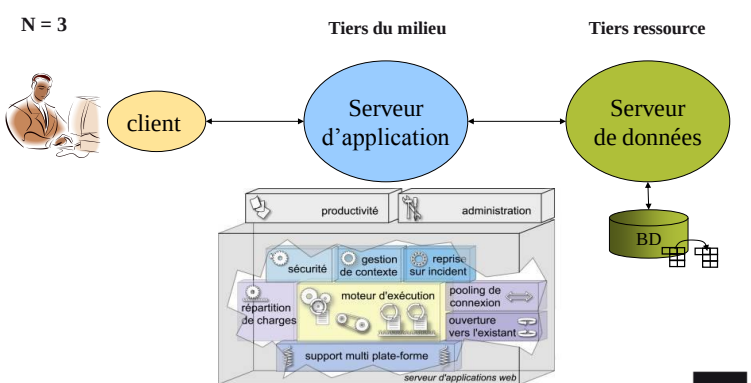
☺ Avantages

- ☺ Répartition des traitements
  - ☺Équilibrage de charge
- ☺ Administration :
  - ☺Clients simplifiés, « légers », peu administrés ;
  - ☺Serveur : Installation et de configurations de logiciels « unique »
- ☺ Ressources centralisées => référentiel
- ☺ Meilleure sécurité : nombre de points d'entrée pour l'accès aux données limité
- ☺ Technologies très matures

☹ Inconvénients

- ☹ Client spécifique au serveur
  - ☹Déploiement d'application
  - ☹Sécurité
  - ☹Applications propriétaires
- ☹ Technicité du serveur => Coût élevé
- ☹ Serveur = maillon faible
  - ☹ Panne serveur bloquante
  - ➢ Systèmes RAID
  - ☹ Passage à l'échelle
- ☹ Asymétrie de l'information au profit des serveurs. Pas de communication entre clients

Architectures à N niveaux



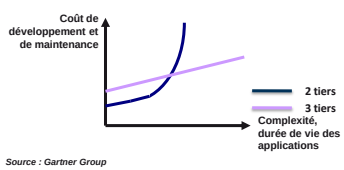
Critique de l'architecture à N niveaux

☺ Avantages

- ☺ Ceux du client/serveur +
- ☺ De l'indépendance !
  - ☺Le développeur ne s'attaque qu'à une seule couche !
  - ☺Maintenance facilitée
  - ☺Réutilisation accrue
- ☺ Répartition de charge
  - ☺Couche applicative = n services = x serveurs
- ☺ Sécurité des données accrues :
  - ☺ un contrôle réalisé par la couche applicative
  - ☺ Données d'accès à la BD ne sont pas chez le client
- ☺ Présentation adaptée au dispositif

☹ Inconvénients

- ☹ Technicité
  - ☹ Grande expertise de développement
  - ☹ Gestion de transactions distribuées
- ☹ Coût initial plus élevé



Source : Gartner Group

Plan du document

- Client/serveur
- **Web statique - voir annexe 1**
  - Internet et Web
  - Architecture Web statique
    - Client
    - Serveur
    - Protocole HTTP
    - Stockage des documents
    - Protection des documents
  - HTML : langage standard du Web
  - Petit bilan du Web statique
- Interfacer les Bases de Données et le Web
- Conclusion
- Et moi que dois-je faire ?

Petit bilan du Web (statique)

- **Avantages**
  - Client universel
  - Facilité d'emploi
  - Standards ouverts
  - Intégration des autres services Internet
  - Extensibilité du système
  - Faibles coûts logiciel et réseau
  - Utilisation au sein d'une entreprise (Intranet)
- **Mais le Web n'est pas conçu pour échanger des données, déployer des applications …**

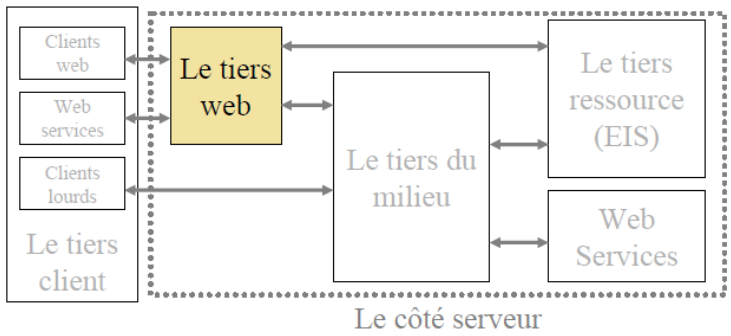
Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - Motivations
  - Architecture Web dynamique générale
  - Architecture 3-tiers Web/BD
  - Principes des passerelles
  - Formulaires Web
  - Passerelle CGI
  - Passerelle PHP
- Conclusion
- Et moi que dois-je faire ?

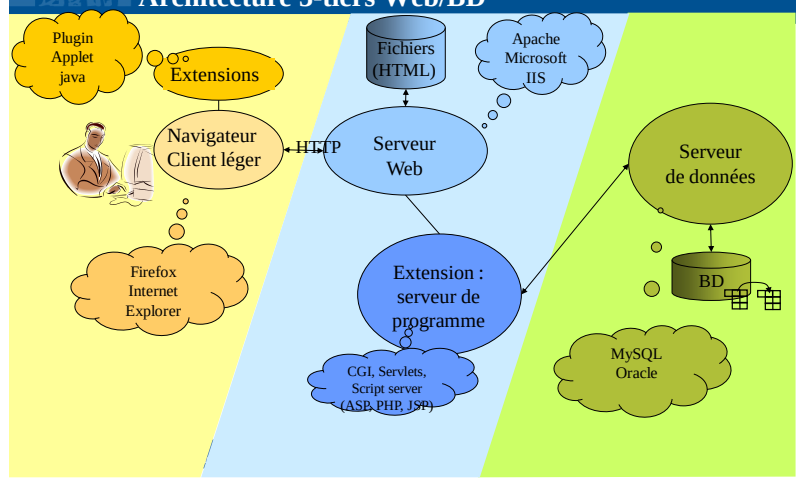
Motivations

- **Le Web, ce n'est pas qu'un ensemble de documents HTML statiques !**
- **Les programmes côté serveur permettent :**
  - de traiter des soumissions de formulaire ;
  - d'afficher de manière uniforme l'ensemble des pages d'un site ;
  - de proposer des applications interactives ;
  - de permettre à l'utilisateur d'ajouter ou modifier du contenu ;
  - etc.

Architecture Web dynamique générale



Architecture 3-tiers Web/BD



Principe des passerelles

- **Déclencher un programme sur le serveur**
  - Passage d'information possible (valeurs d'un formulaire par exemple)
  - Programme renvoie une page HTML, retransmise au client par le serveur Web (comme une page statique ☺)
- **Technologies utilisées dans le tiers Web**
  - **CGI/fast CGI (peut être écrit en java, C, C++, Perl,….)**
  - Pages actives (scripting interprété dans des pages HTML)
    - Langage de script : **PHP (parmi les plus populaire, intégration à Apache facile)**, ASP (microsoft), JSP (nécessite un serveur d'application Java, par ex. Tomcat)
  - Java Servlets : véritables programmes Java; nécessite un serveur d'applications Java en plus d'Apache (Sun, gratuit voire libre)
  - Python

Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - …
  - Formulaires Web – voir annexe 2
    - Formulaires par l'exemple
    - Principes des formulaires HTML
    - Balise `<form>`
    - Champs de saisie
    - Autre utilisation de la balise `<input>`
    - Balise `<input>` - Exemples
  - Passerelle CGI
  - …
- Conclusion
- Et moi que dois-je faire ?

Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - Motivations
  - Architecture Web dynamique générale
  - Architecture 3-tiers Web/BD
  - Principes des passerelles
  - Formulaires Web
  - Passerelle CGI – voir annexe 3
  - Passerelle PHP
- Conclusion
- Et moi que dois-je faire ?

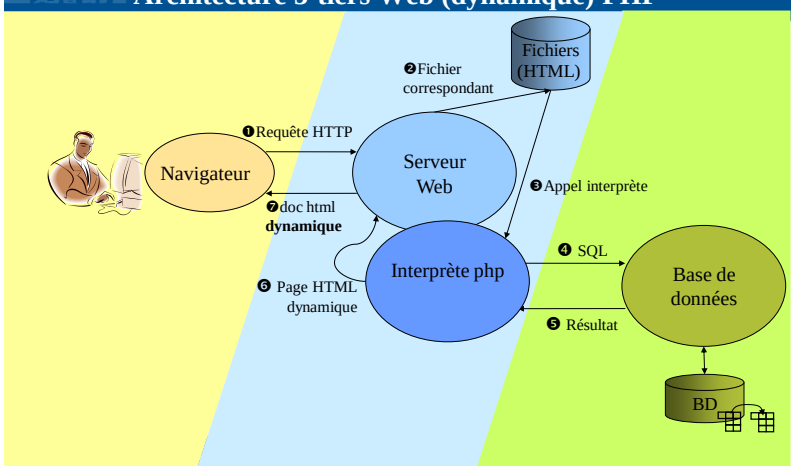
Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - …
  - **Passerelle PHP**
    - Principe des passerelles PHP
    - Architecture 3-tiers Web (dynamique) PHP
    - Langage PHP
    - PHP/PostgreSQL
- Conclusion
- Et moi que dois-je faire ?

Principe des passerelles PHP

- **Version OpenSource de la technologie page active**
- **Intégré à HTML**
- **Dédié à la production de pages HTML dynamiques**
- **Page active : programme s'exécutant côté serveur**
- **Généralise et améliore les CGI**
  - "Mixé" avec HTML
  - Valeurs des variables plus faciles à retrouver
  - Exécution au sein du serveur (Apache) => pas 1 processus par appel
- **Suppose l'installation d'un interpréteur associé au serveur HTTP**
- **Interface BD via une API spécifique ou une couche d'abstraction (ADODB par exemple)**
- **Ressources sur http://www-inf/cours/BD/PRIVATE/AI21-BD**

Architecture 3-tiers Web (dynamique) PHP



Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - …
  - **Passerelle PHP**
    - Principe des passerelles PHP
    - Architecture 3-tiers Web (dynamique) PHP
    - Langage PHP
      - PHP et HTML
      - Variable PHP et conditionnelle
      - Itérations
      - $_GET et $_POST
      - Script serveur PHP - Exemple de gestion de formulaire
    - PHP/PostgreSQL
- Conclusion
- Et moi que dois-je faire ?

PHP et HTML

- **Script PHP : document HTML (par exemple), dans lequel est**
- **incorporé du code PHP**
- **Code PHP à l'intérieur d'une pseudo-balise `<?php ... ?>`**
- **Exemple**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
...
<body>
   <h1>
      <?php echo 2+2; ?>
   </h1>
</body>
</html>
```


Variable PHP et conditionnelle

- **Affectation : `$var=valeur`**
  - Où `valeur` peut être
    - un nombre entier ou réel ;
    - une chaîne de caractères définie entre deux guillemets simples (') ou entre deux guillemets doubles (")
- **Conditionnelle**

```
if (C) { T } else { E }
```

  - Exemple

```
if ($sexe=='f') {
   echo 'Madame';
} else {
   echo 'Monsieur';
}
```


Itérations

- **for (I;C;P) { B } où :**
  - I : affectation permettant d'initialiser la variable itérée,
  - C : condition d'arrêt de la boucle,
  - P est le pas de la boucle,
  - B est le bloc d'instructions itéré
- **Exemples**

```
for( $i=0; $i<10; $i=$i+1 ) {
   echo 'Au secours!';
}
$fact=1;
for( $i=13; $i>0; $i=$i-1 ) {
          $fact = $fact*$i;
}
```

$_GET et $_POST

- **Paramètres HTTP peuvent être récupérés en PHP grâce au tableaux associatifs $_GET et $_POST**

```
<html>
<head><title>Supprimer un vin</title></head>
<body><h1>Supprimer un vin</h1>
<form action="delete.php" method="post">
<p>Identifiant du vin à supprimer :
<input name= "num" type ="text" size=4/></p>
<input value="Réinitialiser" type="reset" />
<input value="Supprimer" type="submit" />
</form>
</body>
</html>
```

delete.php

```
<?php
   ouverture de la connexion

   $query="DELETE FROM vins
           WHERE num=".$_POST['num'].";";
   echo "Requête suppression :".$query."</p>";

   traitement de la requête
?>
```


Script serveur PHP
Exemple de gestion de formulaire

Délimitation PHP

Écriture des valeurs de variables

```
<?php
echo "<h1 align=center>Premier programme PHP
  d'utilisation de formulaires</h1>";
echo "<br><br><ul>";
echo "<li>Nom = ".$_POST["Nom"];
echo "<li>Pr&eacute;nom = ".$_POST["Prenom"];
echo "<li>Ville = ".$_POST["Ville"];
echo "<li>D&eacute;partement =
  ".$_POST["Departement"];
echo "</ul>";
?>
```


Plan du document

- Client/serveur
- Web statique
- **Interfacer les Bases de Données et le Web**
  - …
  - **Passerelle PHP**
    - Principe des passerelles PHP
    - Architecture 3-tiers Web (dynamique) PHP
    - Langage PHP
    - PHP/PostgreSQL
      - Exemple de connexion et requête de mise à jour
      - Exemple de requête d'interrogation
      - Exemple avec une abstraction de BD : ADODB – voir annexe
- Conclusion
- Et moi que dois-je faire ?

Exemple de connexion et requête de mise à jour

```
$server='IPserveur';          // ex : 157.159.110.17
$database='NomDeLaBD';        // ex : tpconception
$user='SonLogin';             // ex : A00
$pass='SonPassword';          // ex : A00

$chaine_cx="host=$server dbname=$database user=$user password=$pass";
$dbconn=pg_connect($chaine_cx) or die ('Could not
  connect'.$chaine_cx.pg_last_error());
echo "<p>connexion etablie</p>";

$query= "insert into vins values (12,'Tavel', 2001, 12.1); ";
$resultat=pg_query($dbconn,$query);
if(!$resultat){
     echo pg_errormessage($dbconn);
} else {
     echo "Nombre de tuples insérés : " . pg_affected_rows($resultat);
}
```

Exemple de requête d'interrogation

```
<?php
   ouverture de la connexion
   $query = "SELECT * FROM vins WHERE num= " .$_POST['num']. "; ";
   echo "Requete select :".$query."</p>";
   $resultat=pg_query($query) or die ('Echec requête'.pg_last_error());;
   if (!$resultat) {
      echo "Erreur postgres :".pg_last_error()."</p>";
   }else{
      echo "Interrogation réussie</p>";
      echo "<table border=1>" ;
      echo "<tr><th>num</th><th>cru</th><th>annee</th><th>degre</th></tr>";
      while ($ligne=pg_fetch_array($resultat, null, PGSQL_ASSOC)) {
         echo "<tr>";
         echo "<td>".$ligne['num']."</td>";
         echo "<td>".$ligne['cru']."</td>";
         echo "<td>".$ligne['annee']."</td>";
         echo "<td>".$ligne['degre']."</td>";
         echo "</tr>";
      }
      echo "</table>";
   }
?>
```


Conclusion PHP-BD

- **Plusieurs APIs possibles**
  - APIs natives :
    - efficaces
    - mais du codage à faire pour changer de SGBD
  - Interfaces abstraites :
    - moins efficace,
    - pas d'adaptation de code à faire,
    - pas de standard pour PHP
- **Beaucoup de ressemblances entre les différentes approches**

Plan du document

- Client/serveur
- Web statique
- Interfacer les Bases de Données et le Web
- **Conclusion**
  - Web/Internet comme environnement client/serveur
  - De site Web à application Web
  - Evolutions du Web
  - Limites du Web
- Et moi que dois-je faire ?

Web/Internet comme environnement client/serveur

- **Protocoles et langages standard (W3C), ouverts et bien diffusés (HTTP, HTML, navigateurs)**
- **Mais**
  - Faiblesses de l'IHM (Java côté client ?)
  - Plus une vision document que traitements (quid du CGI ?)
  - Un protocole sans état (quid de la gestion de session ?)

De site Web à application Web

- **Utilisation du Web comme infrastructure d'exécution des applications de l'entreprise**
- **De nouveaux besoins**
  - Bonne structuration des développements (que faut il modifier si le service communication change la charte graphique ?)
  - Interface avec les applications existantes (ne pas tout redévelopper)
  - Interface avec les SGBD
  - Interface avec d'autres sites (extranet)

Evolutions du Web

- HTML et **XML :** méta-langage permettant de définir de nouveaux langages de balisage (dont HTML), avec séparation nette structure logique et présentation
- De HTTP à **SOAP** : protocole de niveau fonctionnel RPC
- De site Web à **service Web** : possibilité d'accéder par programme à des contenus Web (WSDL, XML, UDDI)

Limites du Web

- **Performances**
  - Réseau Internet => augmenter le débit
  - Scripts CGI => FastCGI, scripts serveurs
- **Sécurité**
  - HTTP => version sécurisée S-HTTP
  - Niveau TCP/IP => solution SSL

Limites du Web (2)

- **Gestion des transactions**
  - Pas possible avec HTTP 1.0
    - Ne travaille pas en mode session
    - Une solution est d'utiliser des "cookies"
  - Version HTTP 1.1
    - Intègre la notion de connexion persistante (au niveau TCP)
- **Construction d'interfaces utilisateur**
  - Limitation de HTML dans sa version actuelle
  - Utilisation de Java

Plan du document

- Client/serveur
- Web statique
- Interfacer les Bases de Données et le Web
- Conclusion
- **Et moi que dois-je faire ?**

Et moi que dois-je faire ?

- **Relire les transparents**
- **Lire la documentation complémentaire :**
  - Cours de Télécom ParisTech : http://www.infres.enst.fr/~danzart/mysql/index.html
- **Comparer cette approche et ce que vous ferez en CSC4002 (JDBC) …**
- **Aller plus loin :**
  - ASR
  - ISI
  - DSI
- **Et finir les mots-fléchés !**

Remerciements

- **Nous remercions Pierre Senellart, Maître de Conférence à Télécom ParisTech pour nous avoir autorisé à alimenter quelques unes de nos diapositives à partir de ses cours**

- **Vous pouvez retrouver ses cours sur :**
  - http://pierre.senellart.com

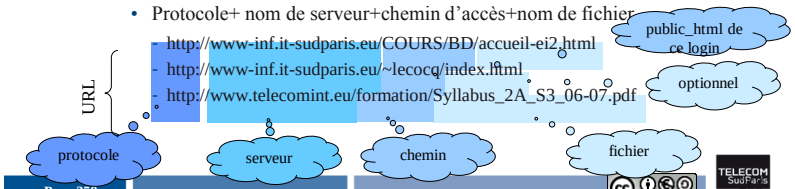
Annexe 1

Web statique

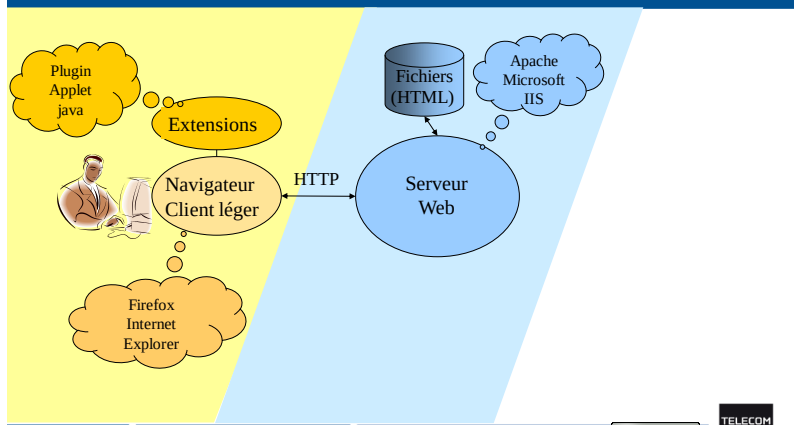
Internet et Web
Architecture Web statique
Client
Serveur
Protocole HTTP
Stockage des documents
Protection des documents
HTML : langage standard du Web

Internet et Web

- **Internet, c'est le réseau**
  - Basé sur le protocole TCP/IP (bas niveau)
  - Ordinateur connu par son adresse IP (157.159.0.0)
  - Association adresse IP ↔ nom (www.it-sudparis.eu)
- **Le Web, ce sont toutes les ressources fournies par les serveurs Web !**
  - Documents HTML
  - Documents multimédias
  - Protocole+ nom de serveur+chemin d'accès+nom de fichier
    - http://www-inf.it-sudparis.eu/COURS/BD/accueil-ei2.html
    - http://www-inf.it-sudparis.eu/~lecocq/index.html
    - http://www.telecomint.eu/formation/Syllabus_2A_S3_06-07.pdf



Architecture Web statique



Client Web

- **Logiciel de consultation hypertexte**
  - interprète et affiche le texte HTML
  - affiche le texte plat sans interprétation
  - visualise les images fixes gif, jpeg et x-bitmap
- **Contient des plug-in ou des visualiseurs externes spécialisés pour divers formats**
  - son, image animée, Postscript, ..

Serveur Web

- **Fonctions assurées**
  - Gère les connexions des clients Web
  - Assure la protection des documents statiques (HTML, images, ..)
  - Vérifie la validité des requêtes et les droits des clients
  - Exécute les requêtes des clients
    - Renvoyer un document statique
    - Exécuter un programme externe (architecture dynamique)

Protocole HTTP

- **HTTP est un protocole simple (GET, EXEC, HEAD, ...)**
- **HTTP est un protocole sans session**
  - La nouvelle version HTTP 1.1 permet de maintenir une connexion persistante
- **HTTP est un protocole peu sécurisé**
  - Évolution avec S-HTTP (Secure) du NCSA ou SSL (Secure Socket Layer) de Netscape

Stockage des documents Web

- **Système de fichiers de la machine serveur**
- **Arborescence de fichiers**
  - Par défaut, à partir du serveur W3 avec un répertoire racine fixé par l'administrateur
  - À partir d'un compte utilisateur avec le répertoire racine ~/public_html avec droits de lecture)
- **Un fichier de l'espace de stockage W3 doit être en lecture pour tous**

Protection des documents Web

- **Protections standard du système d'exploitation**
- **Protections propres au système Web pour accéder un document**
  - Accès avec un nom utilisateur et un mot de passe
  - Accès à partir d'un domaine IP
  - Accès depuis une machine particulière

HTML : langage standard du Web

- **Langage de présentation déterminée par le navigateur**
- **Langage à balises <balise> …. </fin de balise>**
- **Pour la navigation (hyper-documents)**
  - Ancres donnant accès à d'autres documents, d'autres parties (hypertexte)
  - Cible locale ou distante => documents répartis
- **Aspects dynamiques : scripts, formulaires**

Exemple - Extraits de http://www-inf.it-sudparis.eu/COURS/BD/

Bienvenue sur le tutoriel de bases de données relationnelles de l'. . . INT

Ce tutoriel présente les concepts fondamentaux des bases de données relationnelles en les appuyant par des exemples, des exercices, des travaux pratiques sous Oracle. Nous vous proposons plusieurs modes d'interaction :

- Approche "fondamentale" : vous pouvez suivre le cours chapitre par chapitre à la manière d'un livre. Ces cours sont illustrés par quelques exercices d'application;

```
<HTML>
<HEAD> <TITLE>Bienvenue</TITLE> </HEAD>
<BODY TEXT="#000080" LINK="#0000ff" VLINK="#008080" BGCOLOR="#deedfa">
<H2>Bienvenue sur le tutoriel de bases de donn&eacute;es relationnelles de l'<a href="http://www.int-evry.fr"><IMG
    SRC="icones/logoMinus.gif" BORDER=0></a> </H2>
<P><IMG SRC="icones/barre.gif" WIDTH=601 HEIGHT=3></P> <P ALIGN="JUSTIFY">Ce tutoriel pr&eacute;sente les
    concepts fondamentaux des bases de donn&eacute;es relationnelles en les appuyant par des exemples, des exercices, des
    travaux pratiques sous Oracle. Nous vous proposons plusieurs modes d'interaction :</P> <UL> <P ALIGN="JUSTIFY">
<LI> Approche "fondamentale" : vous pouvez suivre le <A HREF="BD_REL/SUPPORT/poly_ToC.html">cours</A> chapitre
    par chapitre &agrave; la mani&egrave;re d'un livre. Ces cours sont illustr&eacute;s par quelques <A
    HREF="BD_REL/EXOS/liste.html">exercices d'application</A>
;</LI></P> ></P> ….
</BODY>
</HTML>
```


Annexe 2

Les formulaires Web

Formulaires par l'exemple
Principes des formulaires HTML
Balise <form>
Champs de saisie
Autre utilisation de la balise <input>
Balise <input> - Exemples

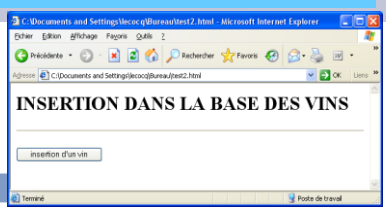
Formulaire par l'exemple : HTML pur

```
<html>
<body>
<h1>INSERTION DANS LA BASE DES VINS</h1><hr>
<form name="insvin"
      action="http://www-inf.it-sudparis.eu/cgi-bin/bd/prive/multi2.cgi"
    target="result" method="POST">
<input type="hidden" name="uid" value="eleve/eleve@TANNA">
<input type="hidden" name="mode" value="INS">
<input type="hidden" name="table" value="VINS">
<input type="hidden" name="sqlstatement" value="">
<input type="submit" value="insertion d'un vin">
</form>
</body>
</html>
```

Nom (complet) du CGI

Paramètres cachés

Bouton. Appel du CGI



Formulaire par l'exemple : HTML avec javascript

```
<html>
<body>
<form name= "interroVin"
action="http://www-inf.it-sudparis.eu/cgi-bin/bd/prive/multi2.cgi"
method="get">
<input type="hidden" name="uid" value="eleve/eleve@TANNA">
<input type="hidden" name="sqlstatement"
 value="select * from vins where cru=">
Donner un cru : <input name="vcru" value="">
<p>
<input type="button" value="lancer"
 onClick="interroVin.sqlstatement.value+=interroVin.vcru.value;
          interroVin.submit();">
</form>
</body>
</html>
```

Champs de saisie = variable

Bouton Concaténation chaîne Appel du CGI

Principes des formulaires HTML

- **Un formulaire autorise l'interactivité entre le client et le serveur**
- **En HTML : uniquement l'interface de formulaire**
- **Essentiel du travail fait par le script qui traite la soumission du formulaire**
- **Un formulaire contient**
  - Des zones de saisie avec des boutons, des listes de choix, ..
  - Un attribut action qui définit l'URL d'un programme exécutable sur le serveur
  - Un attribut `method` qui définit le mode de transfert des données vers le programme

Balise <form>

- **Un formulaire HTML est placé à l'intérieur d'une balise `<form>`**
- **Attributs :**
  - `name` nom du formulaire
  - `action` URL du script auquel sera soumis le formulaire.
  - `method` Méthode HTTP, valant soit "`get`" soit "`post`"
  - `target (optionnel) : nom de fenêtre résultat`
  - `enctype` Encodage HTTP. Peut valoir :
    - "`application/x-www-form-urlencoded`" (valeur par défaut)
    - "`multipart/form-data`"
- **Exemples**

```
<form action="action.php" method="get">
    <div><input type="submit"></div>
</form>
<form name="f1"  action=http://www-inf.it-
  sudparis.eu/prog.cgi method="get"   target="resultat
</form>
```


Champs de saisie

- **Balise `<input>`, utilisation très vaste dans les formulaires**
- **Attribut type :**
  - `type` détermine le type (texte, mot de passe, liste, etc.) du champ
  - `value` (facultatif) permet de préciser la valeur par défaut
  - `name` (nom du paramètre de la requête HTTP) obligatoire (sauf pour les types "`reset`" et "`submit`")
    - permet de préciser au serveur à quelle saisie on fait référence

Autre utilisation de la balise `<input>`

- **`type = "hidden"`**
  - Pour cacher des champs au client mais leur contenu est envoyé avec le formulaire
  - À utiliser avec précaution car cela peut être à l'origine de problèmes de sécurité assez graves : le client peut éditer la page à la main pour changer la valeur de ces champs !
- **`type = "reset"`**
  - Bouton pour réinitialiser le formulaire en affectant aux différents champs leur valeur par défaut
  - Attribut value change le texte du bouton correspondant
- **`type ="submit"`**
  - Pour soumettre le formulaire : le client envoie le contenu du formulaire à l'adresse précisée par l'attribut action de la balise `<form>`
  - Attribut `value` permet de changer le texte du bouton correspondant

Balise <input> - Exemples

```
<input type="text" name="Commentaire« >
<input type="text" name="prenom" value="Jordy"
  maxlength="50">

<input type="hidden" name="nom"  value="10">
<input type="hidden" name="monnaie_utilisee"
  value="EUR">
<input type="hidden" name="customerCB" value="c2415-
  345-8563">

<input type="reset" value="Tout effacer">

<input type="submit" value="Envoyer">
```


Annexe 3

Les CGIs

Passerelle CGI

- **Définit le moyen de passer des informations du client vers le programme CGI et retour**
- **Reçoit un seul argument de type chaîne de caractères**
  - se décompose en couples attributs-valeurs
  - avec deux modes de transmission GET et POST
- **Décode la chaîne pour extraire les paramètres**
- **Renvoie son résultat sur sa sortie standard en ayant préalablement déclaré le type (content-type: text/html par exemple)**
- **Langages utilisés : Perl souvent, mais on peut choisir n'importe lequel**

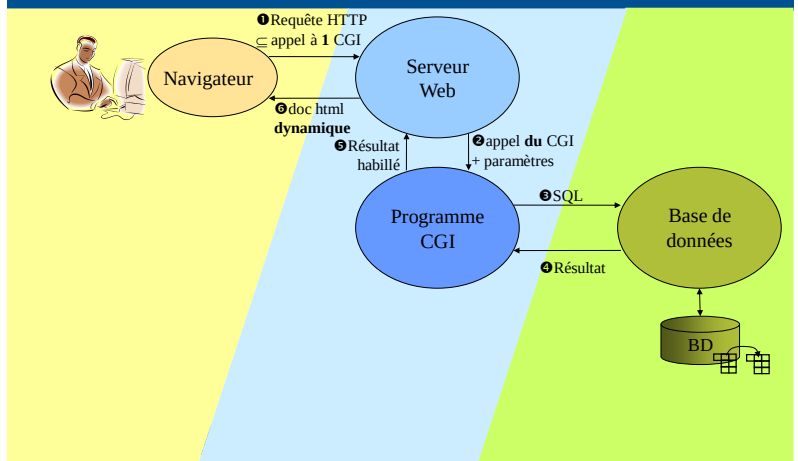
Exemple de CGI en Perl

```
#!/usr/bin/perl -w
use CGI qw (:standard); // écriture sur sortie standard
$ior = param('IOR');
print header;
print "<HTML><HEAD>\n"; // écrit du HTML
print "<TITLE>R&eacute;sultat d&eacute;codage IOR</TITLE></HEAD>\n";
print "<H4>IOR d&eacute;cod&eacute;</I></H4><BR>";
print "<HR WIDTH=80%><BR>\n";
open(FILE, "-|") || exec "/inf/QOS_CORBA/Olivier/bin/iordump", "-f",
   $ior;
// écriture de valeur de variable
while ($LIG=<FILE>) {
    print "$LIG<br>\n";
}
print "<HR WIDTH=80%><BR>\n";
close(FILE);
print "</BODY>\n";
print "</HTML>\n";
```

Rôle du programme CGI/BD

- **Construction d'une page complètement dynamique en fonction du contenu de la BD**
  - Page HTML ou formulaire (plus dynamique)
- **Trois grandes fonctions**
  - Décodage des paramètres venant de la requête HTTP pour en faire des requêtes SQL
  - Exécution du programme SQL
  - Habillage du résultat en HTML (écrit complètement avec des PRINT)
  - Écriture sur la sortie standard

CGI/BD

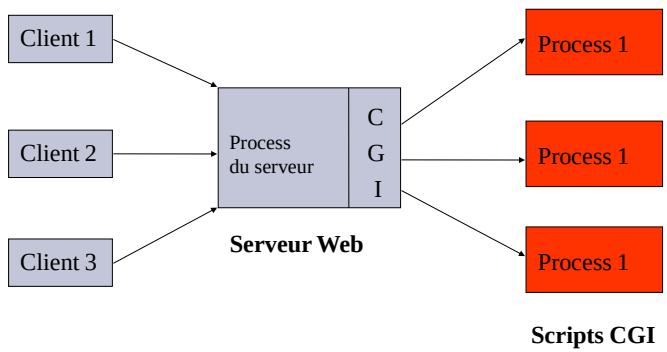


Critique des CGI

☹ **Inconvénients**
- ☹ Beaucoup de code à écrire (un CGI par requête!): peut être amélioré par des solutions génériques
- ☹ Faible intégration avec HTML
- ☹ Session ?
- ☹ Pas très efficace

☺ **Avantages**
- ☺ Simple et portable
- ☺ Utilisation de fait

WWW - Problème du CGI



Annexe 4

Ado db

Exemple avec une abstraction de BD : ADODB

```
<?php
  include('init-ado.php');
  $connection = &ADONewConnection('mysql');
  $connection->Connect(dbnode, dbuser, dbpasswd, dbinstance);
  $connection->debug=true;
  $rs=$connection->Execute('select num, cru from vins');
  rs2html($rs);
?>
```


Quelques fonctions ADODB-PHP

- **Connexion à l'utilisateur 'user1' identifié par le mot de passe 'passwd' sur la base mySQL 'essai' de la machine 'calcite'**
  `$connection->Connect ('calcite', 'user1', 'passwd', 'essai')`
- **Passage en mode debug**
  `$connection->debug=true`
- **Exécution d'une requête SQL (sans variables libres), résultat dans un resultset**
  `$rs=$connection->Execute('select num, cru from vins')`
- **Construction d'une chaîne représentant une instruction insert et exécution**
  ```
  $res=$connection->Execute("insert into vins(num, cru, degre, annee)
    values (".$num. ", '" .$cru. "', " .$degre. ", " .$annee. ")")
  ```
- **Message d'erreur produit par la dernière requête SQL**
  `$connection->ErrorMsg()`
- **Nombre d'attributs dans le resultset**
  `$rs->FieldCount()`
- **Nombre de nuplets produits par la dernière requête SQL**
  `$rs->RecordCount()`

Quelque fonctions ADODB-PHP (2)

- **Objet (avec trois champs) décrivant l'attribut de numéro $i du resultset**
  `$ff = $rs->FetchField($i)`
  - Nom de l'attribut : `$ff->name`
  - Type de l'attribut : `$ff->type`
  - Taille maximale de l'attribut : `$ff->max-length`
- **Accès au nuplet courant -> dans la variable $arr et passe au nuplet suivant**
  `$arr = $rs->FetchRow()`
- **Passage en mode extraction des valeurs via le nom des attributs**
  `$ADODB_FETCH_MODE = ADODB_FETCH_ASSOC`
- **Retour de la valeur de l'attribut 'CRU' du nuplet courant**
  `$rs->Fields('CRU'):`