Web dynamique

Techniques, outils, applications

(Partie D)

SOMMAIRE

1. Sessions avec PHP
2. Javascript
3. templates

Sessions avec PHP

Sessions: pourquoi?

- HTTP est sans état (stateless)
- Les sessions offrent un mécanisme pour créer un état au dessus de HTTP
  - Cela permet de réaliser des opérations au sein de transactions,
  - Cela veut dire permettre de retenir les valeurs de variables et de reconnaitrre un utilisateur d'une requete HTTP à l'autre
  - ce qui permet une interaction avec les bases de données !
- Enregistrement de Session:
  - Cookies
  - Réécriture d'URL

Sessions

- Les Sessions sont à considérer
  - comme un espace de travail qui est rattaché à un utilisateur donné
  - Dans lequel on peut stocker un ensemble de variables
  - Et qui reste persistant tant que l'utilisateur visite les pages de notre site sans fermer le navigateur
- Une session commence :
  - Avec une commande particulière (debut de session) dans la page php
- Et termine via :
  - une commande particulière (fin de session)
  - ou alors en cas d'inactivité, après une délai d'expiration

Gestion de l'etat: les sessions

- Chaque session possède un identifiant unique - qu'on appelle "session id"
- Chaque requête HTTP du client est accompagnée du "session id"
- Les "Session id" sont indiqués via cookie ou via URL
- A ces sessions correspondent des fichiers temporaires sur le serveur Web
  - De format interne ascii, afin de pouvoir partager des sessions avec perl

Cookies

- Correspond à de petits fichiers (<4Ko) envoyés par le serveur et stockés coté client (dans le navigateur)
  - Concrètement les cookies sont de simple fichiers texte contenant des chaines de la forme nom=valeur qui sont stockés dans un sous-repertoire d'un navigateur sur la machine client
  - Une URL est aussi associé au cookie ce qui permet au navigateur de déterminer s'il doit associer un cookie avec une requête vers un serveur
- À chaque visite de page dans un même site le client accompagne la requête HTTP du cookie
- Un cookie peut avoir une durée de vie instantanée ou très longue

PHP : Support des cookies

- Une page peut installer un cookie par appel à la fonction **SetCookie()**
  - **SetCookie("MonCookie", "valeur"); //détruit à la fermeture du navigateur**
  - **SetCookie("MonCookie", $valeur, time()+3600); /* expire dans une heure */**
- Toujours avant tout autre sortie sur la sortie standard (les cookies sont dans le HEADER http)
- Un cookie envoyé par le client sera récupéré sous la forme d'une variable dans la page php
  - **echo $MonCookie;** avec l'exemple ci-dessus
- Les cookies sont peu fiables car ils peuvent être nettoyés à tout moment sur les navigateurs (ceux-ci offrent une option pour cela)

- Exemple :

```
<?
$count++;
setCookie(“count”, $count);
... (a séparer en 2 dans un source PHP)
Echo “Bonjour, vous etes venu sur ce site $count fois!”;
?>
```

Réécriture d'URL

- Peut être utilisé quand les cookies ont été interdits sur le navigateur
- Le "Session id" est ajouté aux URLs dans la page retournée vers le navigateur
- Par ex:

```
<a href="carnet.php">
```

devient

```
<a href="carnet.php?PHPSESSID=F35AD32242DN43E">
```

sessions

- les sessions sont physiquement présents sur le serveur web sous forme de fichier texte.
- Ils sont associés au « session id » grâce à leur nom
- Ce mécanisme est beaucoup plus fiable que les cookies, c'est la raison pour laquelle il est utilisé en majorité

Fonctions de Session PHP

- Les principales:
  - session_start () : crée une session (ou restaure la session trouvée sur le serveur, via l'identifiant de session passé)
  - session_destroy (): détruit la session courante et toutes les données associées avec cette session
  - session_register (): Enregistre une variable dans une session
  - session_unregister (): Supprime une variable de la session
  - session_is_registered ():Vérifie si une variable est enregistrée dans la session
  - session_id (): identifiant courant de session

Accès aux variables de requêtes

- Il a eu une évolution dans ces accès de variables.
- Il existe actuellement plusieurs manières d'accéder à des variables issues d'une requête HTTP (GET ou POST)
- À savoir:
  - $nom directement
  - $nom=urldecode($HTTP_GET_VARS['nom']);
  - $nom=urldecode($HTTP_POST_VARS['nom']);
  - $nom=$_GET["nom"];
  - $nom=$_POST["nom"];
  - $nom=$_REQUEST["nom"]; // fonctionne quelque soit la méthode (GET ou POST)
- Ces variantes ci-dessus sont dites « superglobales »
- La méthode 1 ($nom) n'est possible que si register_globals=on dans le fichier de configuration de l'interpréteur PHP (php.ini) - Pour des raisons de sécurité, elle n'est pas conseillée...

Variables spéciales

- $PHP_SELF: nom du script lui-même
  - Exemple dans un formulaire généré :
  - <form method=post action = $PHP_SELF >
- $_SERVER Variables serveur:
  - $_SERVER['PHP_SELF'] equivalent à $PHP_SELF
- $_COOKIE tableau des cookies

Exemple de Session Simple

```
<?session_start(); ?>
<html>
<head><title>Test de Session</title></head>
<body>
<?
   session_register("scount");
   $scount++;
   $pcount++;
?>
Compteur de Page: <? echo $pcount; ?><br>
Compteur de Session: <? echo $scount; ?><br>
</body>
</html>
```

Exemple de Session Simple V2

```
<?session_start(); ?>
<html>
<head><title>Test de Session</title></head>
<body>
<?
   Sscount=$_POST["scount"];

   $scount++;
   $pcount++;

   // Enregistre dans la session la cle et la valeur
   $_SESSION['scount'] = $scount;

?>
Compteur de Page: <? echo $pcount; ?><br>
Compteur de Session: <? echo $scount; ?><br>
</body>
</html>
```

Javascript

Javascript

- Dérivé de Java (donc de C)
  - Instructions communes avec Java/C++
  - Commentaires idem // et /* ... */
  - ';' conseillé mais non obligatoire
  - Fait la différence entre les majuscules et les minuscules
- Insertion
  - dans l'entête du document HTML en général
  - N'importe où dans le corps du document HTML
- Utilisé surtout pour:
  - la vérification des données saisies dans un formulaire HTML juste avant l'envoi
  - Mettre de l'animation dans les pages web
- Modèle événementiel pour réagir aux évènements déclenchés par l'utilisateur (click de souris, etc)
  - OnSubmit
  - OnClick
  - OnMouseOver
  - ...

Javascript avec HTML

- Intégré dans un page HTML : <SCRIPT>

le return peut **bloquer** ou pas l'envoi vers le serveur

Autre accès:
document.forms[0].elements[0].value

```
<Head>
<script language="JavaScript">
<!-- //masquer pour les vieux navigateurs

function ValidationForm()
{
   if (document.form1.telephone.value.length < 10)
   {
       alert("les numéros de tel. ont 10 chiffres");
       return false;
   }
   else return true;
}
//-->
</script>
</head>
<body>
<form name="form1" action="/cgi-bin/prog" method="POST" OnSubmit="return
    ValidationForm();">
<input type=text name="telephone">
<input type=submit VALUE="Envoyer">
</form>
</body>
```

Javascript et HTML

- Inclusion de fichier externe javascript :

```
<HTML>
<HEAD>
<TITLE>javascript</TITLE>
 <SCRIPT Language="Javascript"
   Src=“essai.js">
</SCRIPT>
</HEAD>
...
```

- Le fichier externe contient le code directement (sans balises script)

Javascript et HTML

- Intégration + interaction avec le document:

```
<HTML>
<HEAD>
<SCRIPT>
<!--
var b = 10;
//-->
</SCRIPT>
</HEAD>
<BODY>
Nous sommes dans le corps de la page.<p>
<SCRIPT Language="javascript">
<!--
   document.write("b"= +b);
//-->
</SCRIPT>
<P>Fin de la page.
</BODY>
</HTML>
```

Javascript: les types simples

- Déclaration de variable: mot clé var
  - var mavar;
- Types :
  - Nombres
    - Mavar = 10;
    - Mavar = 3.1415;
  - Booleens
    - Mavar = true || false;
  - Chaines (caractères " et ‘ utilisables)
    - Mavar = "une chaine";
- Type non figé
  - Le type des variables est déterminé lors d’une affectation
  - une variable peut changer de type en cours de route
    - var mavar="hello";
    - mavar = 10;
    - Mavar = true;

Javascript: les tableaux

- Création
  - var tab=new array(10);
- Utilisation
  - Tab[1]=10;
- Taille
  - tab.length
- Création dynamique
  - On peut créer de nouveau éléments en indexant au dessus de la taille actuelle du tableau

Javascript: Structures de contrôle

- Comme en C
  - If
  - For
  - While
  - Continue / break
- Fonctions : mot clé « function »
  - Devient une procédure si return absent

```
function test(i){
    var b=0;
    while(i>0){
        b+=i;
        i--;
    }
    return b;
}
```

Javascript: fonctions de chaînes

- Principalement (même syntaxe que Java)
  - length()
  - charAt(value)
  - indexOf(string)
  - substring(value1,value2)
  - toLowerCase()
  - toUpperCase()
- Utilisables par « pointage » (comme java)
  - var Mavar = « une chaine de caractères »;
  - Mavar.length()
  - Mavar.charAt(4) : retourne ‘c’
  - "Hello".charAt(1) : retourne ‘e’
  - Etc..

Javascript et HTML

- Evènements
  - Une liste d'évènements est disponible pour chaque balise HTML (format : « on... »)
  - On leur associe du code entre double quote
- Exemple :

```
...
<A href= "url"
onMouseOver="alert('la souris vient de passer dessus');">
un lien hypertexte</A>
...
```

- Si on ajoute « return false; » l'événement est annulé (à utiliser par exemple avec onSubmit)
- **On voit au passage la fonction alert(), utile pour débugger par exemple**

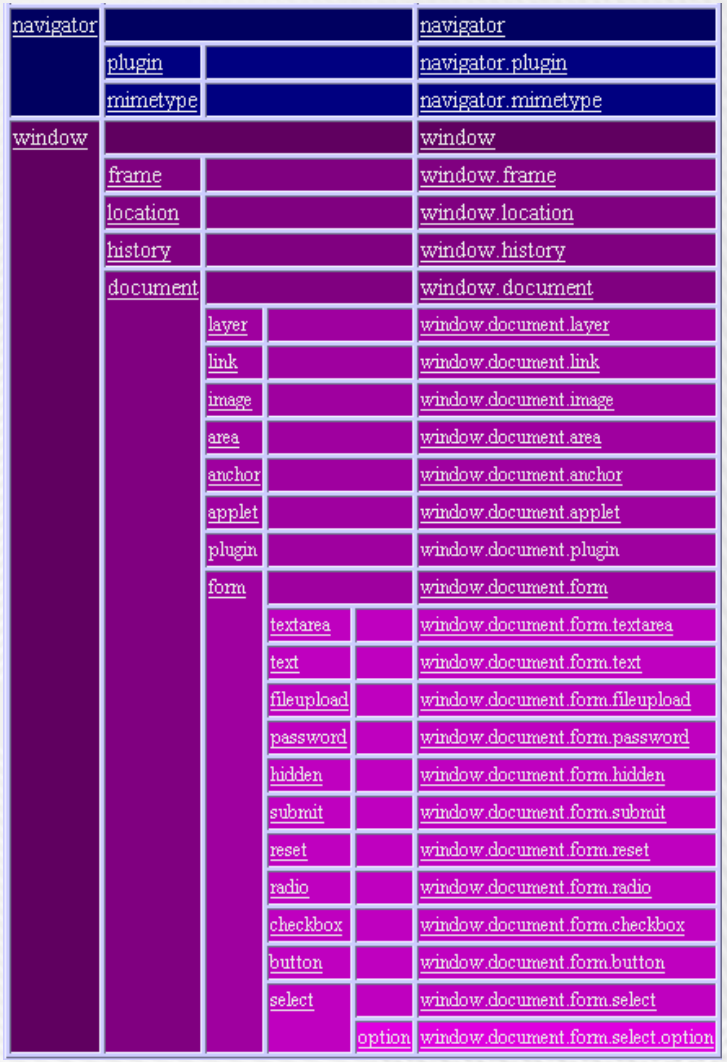
Javascript et HTML

Principaux évènements :

Evénement	description	principaux tags
onBlur	perd le focus	<INPUT> <SELECT> <TEXTAREA>
onClick	click avec la souris	<INPUT> <SELECT> <A>
onChange	la valeur a changé	<INPUT> <SELECT> <TEXTAREA>
onFocus	prend le focus	<INPUT> <SELECT> <TEXTAREA>
onLoad	le document HTML est complètement chargé	<BODY>
onMouseOut	la souris sort de la zone de l'objet	<A ... >
onMouseOver	la souris passe sur la zone de l'objet	<A ... >
onSubmit	envoie d'un formulaire vers le serveur	<FORM>

Document Object Model

- A connaître si on veut maîtriser javascript.
- Permet de retrouver des éléments du navigateur
- Spécifications complètes à l'origine par Netscape



Exemples pratiques

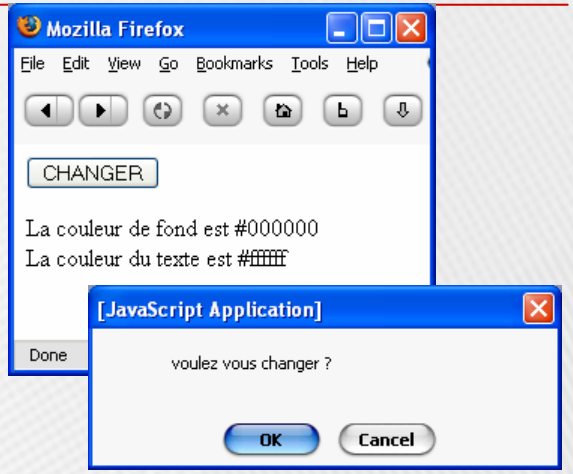
- Fonction confirm(): Permet de poser une question coté client

```
<HTML>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!-- masquer pour les vieux navigateurs
function changer(Message) {
        if ( confirm (Message)) {
                document.TEST.Bete.value=
                document.TEST.Bete.value=="MODIFIER"?
                                "CHANGER":"MODIFIER";
                var old= document.fgColor;
                document.fgColor=document.bgColor;
                document.bgColor=old;
        }
}
// fin masquer pour les vieux navigateurs -->

</SCRIPT>
<form name="TEST">
<input type=button name="Bete" value="CHANGER" onclick="changer('voulez
        vous changer ?');" ></form>

<SCRIPT LANGUAGE="JavaScript">
document.write("La couleur de fond est ", document.fgColor,
"<BR>La couleur du texte est ",document.bgColor);

</SCRIPT>
</BODY>
</HTML>
```



Exemples pratiques

- Affichage coté client de sa propre adresse

```
<HTML>
<BODY>Affichage coté client de sa propre addresse.<br>
<SCRIPT LANGUAGE="JavaScript">
myAddress=java.net.InetAddress.getLocalHost();
host=myAddress.getHostName();
ip=myAddress.getHostAddress();
document.write("Votre Host est: "+host);
document.write("<BR>Votre adresse IP est : "+ip);
</script>
</BODY>
</HTML>
```

Exemples pratiques

- Determiner le type de navigateur

```
<HTML>
<HEAD>
<script language="JavaScript">
<!--
  browserName = navigator.appName;
  browserVer = navigator.appVersion;
  browserVer = browserVer.substring(0,browserVer.indexOf(".") + 3);
  browserAgent = navigator.userAgent;
//-->
</script>
</HEAD>
<BODY>
<H2>Determiner le type de navigateur</H2>

<p><b>Approche 1: via appName et appVersion</b><BR>
<script>
document.write("Vous utilisez <b>" + browserName + " v" + browserVer + "</b>");
</script>

<p><b>Approche 2: via le userAgent</b><BR>
<script>
document.write("Vous utilisez <b>" + browserAgent + "</b>");
</script>

</BODY>
</HTML>
```

Ressort:
“Vous utilisez Netscape v5.0”

Ressort:
“Vous utilisez Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.10) Gecko/20050716 Firefox/1.0.6”

Exemples pratiques

- Effet « bouton poussoir »
  - Utilisation de MouseOver et MouseOut
- Exemple typique:

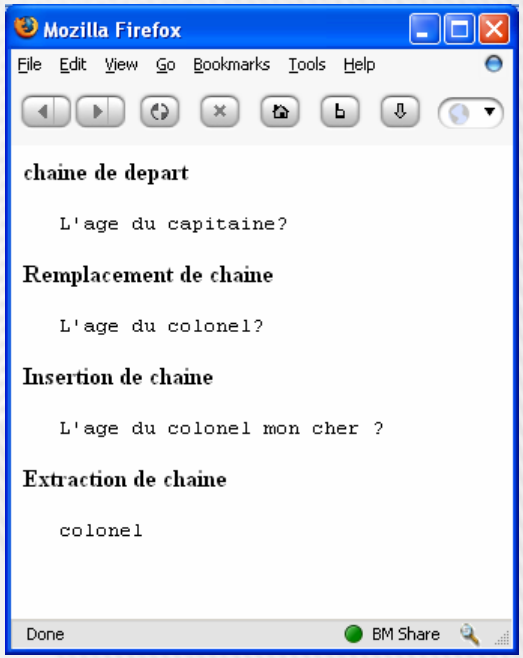
```
<HTML>
<BODY>
<A HREF="http://index.html" onMouseOver="img1.src='APPUYE.GIF'"
    onMouseOut="img1.src='NONAPPUYE.GIF' ">
<IMG NAME="img1" BORDER=0 SRC="NONAPPUYE.GIF"></A>
</BODY>
</HTML>
```

- Le bouton bascule lorsqu'on passe la souris dessus (même sans cliquer)

Manipulations de chaines

```
<HTML>
<BODY>
<p><b>chaine de depart</b>
<script>
 demoString = "L'age du capitaine?";
  document.write("<pre>   " + demoString + "</pre>");
</script>
<p><b>Remplacement de chaine</b>
<script>
  demoString = demoString.replace("capitaine","colonel");
  document.write("<pre>   " + demoString + "</pre>");
</script>
<p><b>Insertion de chaine</b>
<script>
  whereIS = demoString.indexOf("?");
  demoString = demoString.substring(0,whereIS)
  + " mon cher " + demoString.substring(whereIS);
  document.write("<pre>   " + demoString + "</pre>");
</script>

<p><b>Extraction de chaine</b>
<script>
  whereIS = demoString.indexOf("colonel");
  mot = demoString.substring(whereIS,whereIS+"colonel".length );
  document.write("<pre>   " + mot + "</pre>");
</script>
</font>
</BODY>
</HTML>
```



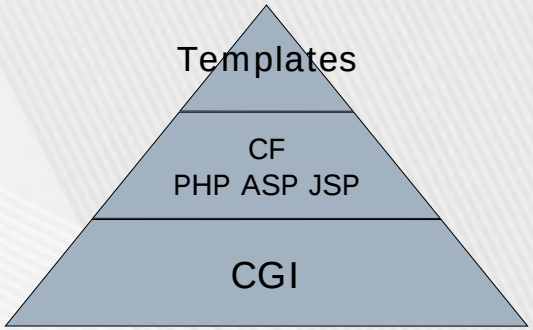
Templates

Séparer contenu et présentation

Petit rappels sur les problematiques

- Rappel
  - Le Web ne s'est vraiment développé qu'a partir du moment où il a proposé des contenus dynamiques
  - Un contenu dynamique est nécessairement connecté à une base de données (dans son sens large)
- les problématiques associées au web dynamique
  - Le maintien de sessions transactionnelles
  - la performance
  - la sécurité
  - la réutilisabilité du code
  - Séparation logique / Cosmétique
- A retenir
  - CGI est le dénominateur commun à tous les serveurs
  - Les API serveurs ont permis d'améliorer les performances de CGI
  - Les scripts serveurs s'appuyant sur CGI proposent des facilités (sessions, mixage codes tiers serveur/client)
  - Les Templates permettent la séparation du contenu et de la présentation

	CGI	API Serveurs	Scripts serveurs	Templates
sessions			X	
performance		X		
Sécurité			X	
réutilisabilité			X	X
Séparation logique / Cosmétique				X
Disponibilité serveur/OS	X		X	X



Le principe

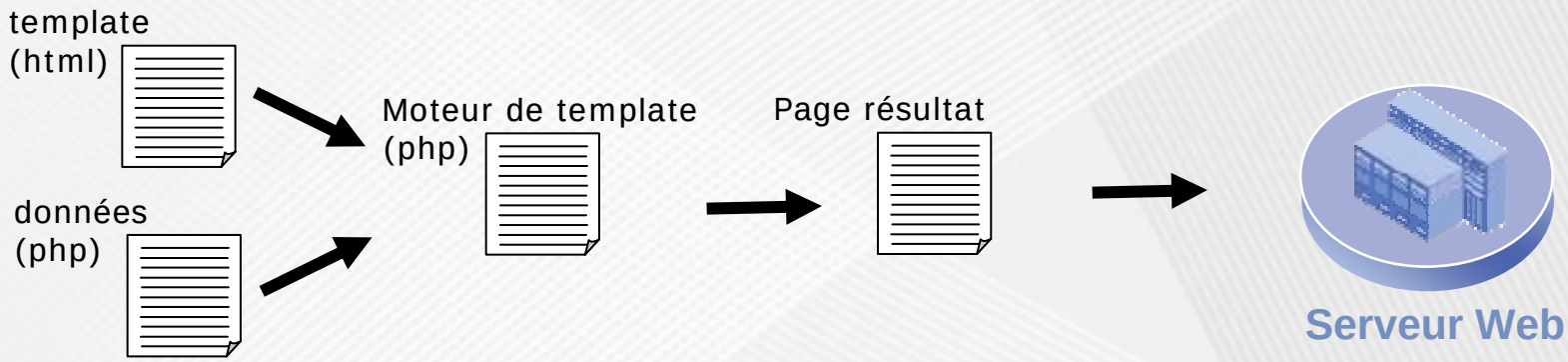
- Les moteurs de templates permettent la séparation du contenu (les informations issues de la base de données par exemple) et de la présentation dans des fichiers distincts
- On peut donc considérer qu'il s'agit d'un retour en arrière par rapport aux principes des scripts serveurs

- Les moteurs de templates resolvent 2 gros problemes:
  - Comment réussir cette separation
  - Comment séparer le code php/jsp/asp considéré comme "complexe" du code HTML

- Cela permet en théorie et en pratique, à des infographistes HTML de modifier l'apparence du site sans avoir besoin de regarder le moindre code PHP.

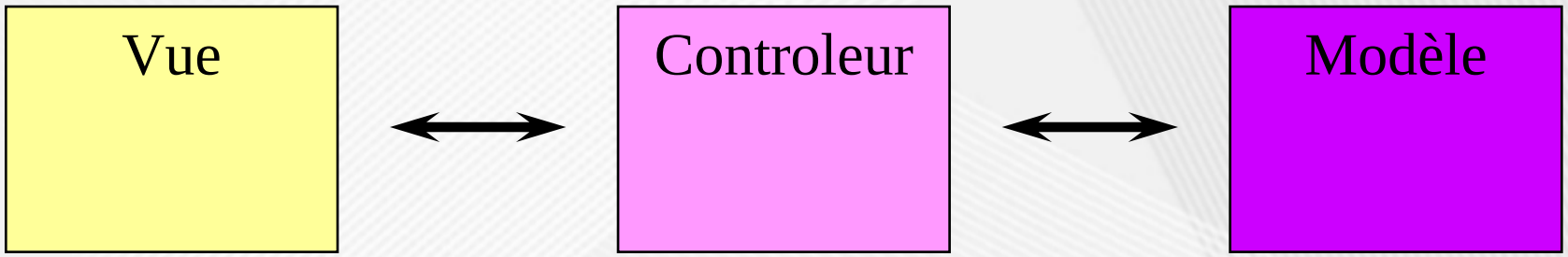
Le principe

- En general, cela s'organise comme suit:
  - Une page PHP en charge du contenu (ou données ou business logic)
  - Un fichier HTML de "mise en forme" (template)
  - Le moteur de template lui-même (sur un ou plusieurs fichiers)



Le principe MVC

- L'objectif est de s'orienter vers le modèle MVC
- Il est prouvé que ce modèle assure une meilleure maintenant dans un développement
  - Appliqué partout, par exemple Windows (MFC et Visual Studio), Java (JFC, JSF…)
- Principe:
  - Séparation Vue (=interface) et Modèle (=appli info)
  - Éventuellement est placé un Contrôleur(validation des informations de la vue) entre les deux (rarement fait)
- Bénéfices:
  - Plusieurs vues possibles pour un même modèle (exemple : 2 fenêtres Word sur le même document, une vue XML ou HTML d'une base de données)
  - Des personnes différentes peuvent travailler en parallèle sans se marcher sur les pied



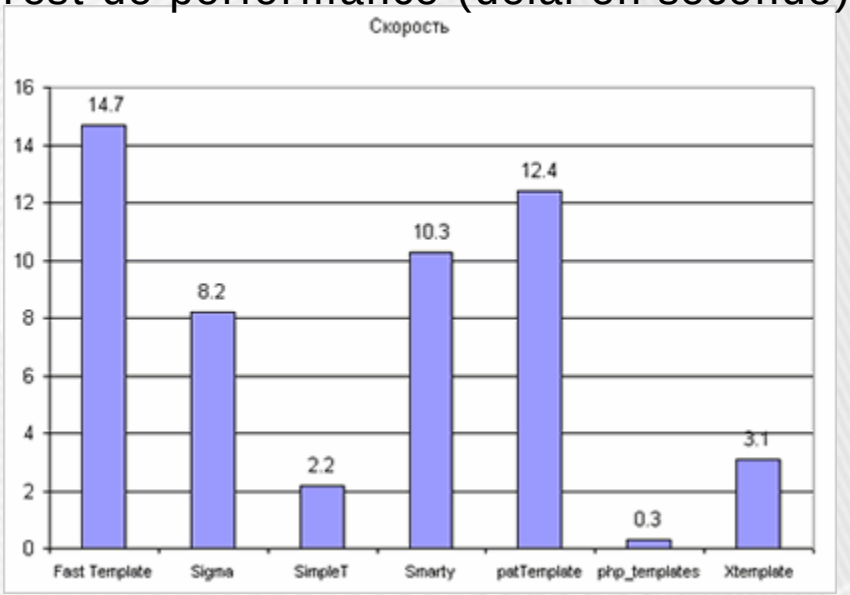
Offre actuelle

- Existe de multiples déclinaisons dans toutes les technologies cités dans ce cours
  - CGI
  - scripts serveur: JSP, ASP, PHP, Cold Fusion
  - Cold Fusion est une exception car on peut considérer que la séparation est intégré dans le langage CFML
  - Servlets Java
- Exemples:
  - Java: struts, JSF (Java Server Faces)
  - PHP: PHPtemplates, smarty, sigma
  - Perl: dTemplate

Focus sur PHP

- Principaux moteurs pour PHP
  - Smarty
  - Fast Template
  - Sigma
  - php_Templates
  - XTemplate
  - patTemplate
  - SimpleT
- S'appuient tous sur un modèle objet
  - Objets en PHP non couvert dans ce cours

Test de performance (délai en seconde)



Source: http://wiki.essentialmind.com/NetArch/tools/templates?v=45d