## Memento BASE PHP : "never trust input user"

# structure du PHP

|  |  |  |
| --- | --- | --- |
| **Balise php** | <?php ... ?> | Balise entourant le code PHP |
| **Echapper caractère** | \"exemple\" | Pour échapper un caractère mètre un \ avant celui-ci |
| **Commentaire** | // ou /\* \*/ | Pour écrire un commentaire dans la balise php |
| **Inclure un fichier** | <?php include("mapage.php"); ?> | Inclure un fichier PHP |

## Les bases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Afficher texte** | echo "texte"; | | | Affiche du texte | |
| **variable**  **forcer conversion** | $mavar = 0;  $mavar = (int) $mavar | Donne à mavar la variable 0 (la var peut être un entier(int), decimaux(float) string,booléens ou NULL)  Force la conversion de $mavar en un entier | | | |
| **Concaténer une variable** | echo "Le visiteur a $age\_du\_visiteur ans"; | | | Avec les " " | |
| echo 'Le visiteur a ' . $age\_du\_visiteur . ' ans'; | | | Avec les ' ' (**préférer cette version** pour la lisibilité et rapidité d'exécution PHP) | |
| **Condition** | If (condition) {code}  else ou elseif | | | Si... // **option utiliser le ! pour tester « si n’est pas » // && AND = et || OR = ou**  "Sinon" ou "sinon si" (bien attacher elseif en un seul mot!) | |
| **Condition ternaire** | $age = 24;  $majeur = ($age >= 18) ? true : false; | | | Si l'age >= 18 donne la valeur true, sinon false.  Syntaxe : $var = (condition) ? code\_si\_vrai : code\_si\_faux; | |
| **Switch** | switch ($mavar){ case 0: echo "tu as zero"; break;  case x (...); default: echo "pas compris"; } | | | Test la variable $mavar si elle vaut 0 le message "tu as zero" (mettre un break)  Ainsi de suite (...); et un message default si aucune n'est vérifier (fermer } ) | |
| **Boucle while** | while (condition) {code} | | | Tant que la condition est ok exécute le code | |
| **Boucle for** | for ($var =0; $var < 0; $var++) {code} | | | Initialise la variable; condition; action sur la variable | |
| **Déclarer fonction** | function maFonction(parametre){code; return $uneVar} | | | | Déclare une fonction qui exécute le code et renvois $unevar |
| **Appel fonction** | $mavar = maFonction(); | | | Stock dans la variable $mavar le résultat de la fonction | |
| **Exemple de fonction** | | | | | |
| **Compte les caractères** | $longueur = strlen($phrase); | | | Compte le nombre de caractère dans $phrase et sotck dans $longueur | |
| **Rechercher et remplacer** | $ma\_variable = str\_replace('b', 'p', 'bim bam boum'); | | | 3 paramètres : 1-la chaine qu'ont remplace, 2-ce qu'ont met a la place et 3-la chaine dans laquel ont fait la recherche | |
| **Mélanger lettres** | $chaine = str\_shuffle($chaine); | | | Mélange les caractères de $chaine et remplace sa valeur | |
| **Ecrire en minuscule** | $chaine = strtolower($chaine); | | | strtoupper pour mettre en majuscule | |
| **Récupérer la date** | $annee = date('Y'); | | | Recupere ici l'année.(H=heure; i=minute,d=jour,m=mois,Y =année) | |
| **tableau** | $tableau = array(valeur1, valeur2, valeur3);  $tableau= array('nom' => 'machin', 'heure' => '10')  $tableau['prenom'] = 'machin'; | | | Stocke valeur dans le tableau. pour afficher le premier : $tableau[0];  Stocke dans un tableau des valeurs associé a un libeller  Stocke dans $tableau la valeur 'machin' avec 'prenom' comme libeller | |
| **Affiche tout tableau** | echo print\_r($tableau); | | | Affiche l'ensemble du tableau (utile pour debugger) | |
| **foreach** | foreach ( $tableau as $valeur)  foreach ( $tableau as $libelle => $valeur) | | (foreach = pour chaque élément) Affiche un à un les élément avec sa valeur  Idem avec tableaux associatif, on peux donc utiliser le libelle et valeur | | |
| **Recherche dans un tableau** | array\_key\_exists('nom', $coordonnees) | | | Si le libeller(=key) 'nom' est trouver dans le tableau $coordonnees (booléens) | |
| in\_array('valeur', $coordonnees) | | | Même chose mais pour la valeur au lieu du libeller(key) (type booléens) | |
| array\_search('fraise", $fruits); | | | Recherche si 'fraise' se trouve comme valeur du tableau $fruit, renvois la position (int) | |

## Transmettre des données avec l'URL (faire très attention avec cette méthode : sujet au modification user !!)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Passer variable url** | Ex : http://www.site.com/page.php?nom=Dupont&prenom=Jean | | Ce qui se situe âpres le ? sont les variable, séparer par des & | |
| **Crée lien avec paramètre** | <a href="bonjour.php?nom=Dupont&amp;prenom=Jean">Dis-moi bonjour !</a> | | | Le & remplacer par &amp; en html |
| **Récupérer les paramètres** | <p> Bonjour <?php echo $\_GET['prenom']; ?> ! </p> | | On récupère avec $\_GET['variable'] (pas dépasser 256 carac) | |
| **Conversion en int** | | $\_GET['age'] = (int) $\_GET['prenom'] ; | On s’assure du typage (faille xss) | |
| **Tester la présence d'un paramètre** | | isset($\_GET['prenom'] | Cette fonction test si une variable existe | |
| **DANGER : toujours faire vérification isset avec &&(quand n variable) et des conditions pour gérer les éventuel modification par user** | | | | |

## Transmettre des données avec les formulaires (faire très attention et toujours verifier les donnée : faille XSS)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **form en HTML :**  **input texte input zone**  **liste deroulante**  **''**  **Case à cocher**  **''**  **Bouton d'option**  **''**  **Champs cacher** | <form method="post" action="cible.php">  <input type="text" name="pseudo" value="M@teo21" />  <textarea name="message" rows="8" cols="45">  <select name="choix">  <option value="choix1" Choix 1</option> </select>  <input type="checkbox" name="case" id="case" />  <label for="case">Ma case à cocher</label>  <input type="radio" name="frites" value="oui" id="oui"/> <label for="oui">Oui</label>  <input type="radio" name="frites" value="non" id="non" /> <label for="non">Non</label>  <input type="hidden" name="pseudo" value="Mateo21" /> | | | | : Utilise la méthode post et envoie a cible.php (post pas plus sécurisé que get mais mieux = non limité a 256 caractères)  : Crée une liste déroulante avec le name ="choix"  : Définie les options  : Crée une case à cocher  : Avec son libeller  : les deux options doivent avoir le même name (pour ne pas qu'ont puissent choisir les deux)  : crée un champs cacher |
| **Récupérer texte**  **Zone de texte**  **Liste déroulante**  **Case à cocher**  **Bouton d'option**  **Champs cacher** | | <?php $\_POST[' pseudo ']; ?>  <?php $\_POST[' message ']; ?>  <?php $\_POST[' choix ']; ?>  <?php $\_POST[' case ']; ?>  <?php $\_POST[' frites ']; ?>  <?php $\_POST[' pseudo ']; ?> | | Récupère les donnée de l'input texte avec le name= "pseudo"  Récupère les donnée de l'input zone texte avec le name= " message "  Récupère la value du select avec le name= " choix "  Récupère la valeur "on" si name= " case " est cocher, sinon rien  Récupère la value du radio avec le name= " frites "  Récupère la value du champs name="pseudo" **(attention user peut voir)** | |
| **Echapper code html**  **Retirer les balises** | | | htmlspecialchars($\_POST['pseudo']); ?> !</p>  strip\_tags($\_POST['pseudo']); ?> !</p> | Echappe les balise HTML**TOUJOURS LE METTRE POUR DU TEXTE**  Retire les balises html ecrit dans le texte | |
| **Redirection** | | | header (Location: mapage.php'); | Redirige vers la page mapage.php | |

## Variable superglobales (comme $\_POST, $\_GET)

|  |  |
| --- | --- |
| $\_SERVER['REMOTE\_ADDR'] | Elle nous donne l'adresse IP du client qui a demandé à voir la page |
| $\_SESSION | variables de session. Ce sont des variables qui restent stockées sur le serveur le temps de la présence d'user |
| $\_COOKIE | contient les valeurs des cookies enregistrés sur l'ordinateur du visiteur |
| $\_FILES | elle contient la liste des fichiers qui ont été envoyés via le formulaire précédent |

## Session (écrire session\_start() avant le <!doctype>)

|  |  |  |  |
| --- | --- | --- | --- |
| **Démarre système session** | session\_start() | Vous devez appeler cette fonction au tout début de chacune des pages où vous avez besoin des variables de session | |
| **Ferme la session** | session\_destroy() | Automatiquement appeler âpres plusieurs minute (timeout) // utile pour une "déconnexion" | |
| **Créer variable session** | $\_SESSION['nom'] = valeur; | | Crée une variable de session nom avec comme valeur valeur (a déclarer âpres session\_start() ) |

## Cookies (placer setcookies avant <!doctype>) (utilisé le sécurisé !!) (tjrs vérifier avec isset)

|  |  |  |  |
| --- | --- | --- | --- |
| **Crée un cookie** | setcookie('nom', 'valeur', time() + 365\*24\*3600); | Crée un cookie avec le nom nom la valeur valeur et expirant dans 1 an(365jourx24hx3600s) | |
| **Sécurisé** | setcookie('pseudo', 'M@teo21', time() + 365\*24\*3600, null, null, false, true); | | Le dernier true active httpOnly (= +securisé) |
| **Utilisé cookies** | $\_COOKIE['pseudo'] | | Récupère le cookies avec la valeur de pseudo |

## Lire et écrire dans un fichier (il faut les droit chmod en 777 pour que PHP est l'accès !)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ouvrir un fichier** | $monfichier = fopen('lenom.txt', 'r+'); | | | Ouvrir en R+= lecture et écriture / r= lecture seul / a=écriture seul /a+ = lecture et écriture (a=le crée) |
| **Fermer un fichier** | fclose($monfichier); | | | Ferme le fichier contenu dans la variable, ne pas oublier ! |
| **Lire caractère par caractère** | | | $caractere = fgec($monfichier); | On stock le premier caractère dans la variable $caractere |
| **Lire ligne par ligne** | | | $ligne = fgets($monfichier); | On stock la première ligne dans la variable $ligne |
| **Ecrire sur le fichier** | | fputs($monfichier, 'Texte à écrire'); | | Ecrit la ligne 'Texte à écrire' dans le fichier $monfichier |
| **Se placer dans le fichier** | | fseek($monfichier, 0); | | Se place au tout début du fichier $monfichier (si on a ouvert avec a ou a+ le curseur est a la fin du fichier) |

## créer, afficher et modifier des images (il faut que la bibliothèque GD soit activé)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Renvoyer image** | | header ("Content-type: image/png"); | | Au lieu de renvoyer une page html le script php renvois une image (png ou jpg ) | | |
| **Créer image** | | $image = imagecreate(200,50);  $image = imagecreatetruecolor(200, 150); | | Crée une nouvel image de 200px x 50px, $image est une ressource contenant l'image.  Crée une nouvelle image avec plus de choix de couleur (indispensable pour miniature!) | | |
| $image = imagecreatefromjpeg("couchersoleil.jpg"); | | Crée une image a partir d'une existante // Variante : imagecreatefrompng | | |
| **Afficher image**  **Dans le html** | | imagepng($image); **ou**  imajpeg($image);  <img src="image.php" /> | | Affiche l'image dans le script  Crée une balise img qui contiens l'image renvoyez par le script PHP | | |
| **Enregistrer image**  **Afficher html** | | imagepng($image, "images/monimage.png");  <img src="images/monimage.png" /> | | Enregistre l'image dans le dossier images (pas besoin du header dans le cas la) | | |
| **couleur** | | $orange = imagecolorallocate($image, 255, 128, 0); | Applique à $image la couleur orange (le premier appel imagecolorallocate est la couleur de fond | | | |
| **texte** | imagestring($image, $sizePolice, $x, $y, $texte, $couleur); | | | Prend en paramèt re la ressource, la taille police, la position, le texte et la couleur | | |
| **Dessiner : pixel**  **droite**  **ellipse**  **rectangle**  **polygone** | | ImageSetPixel ($image, $x, $y, $couleur); | | Dessine un pixel au coordonnée x,y | | |
| ImageLine ($image, $x1, $y1, $x2, $y2, $couleur); | | Dessine une droite entre deux point | | |
| ImageEllipse ($image, $x, $y, $largeur, $hauteur, $couleur); | | Dessiner une ellipse dont le centre est x,y | | |
| ImageRectangle ($image, $x1, $y1, $x2, $y2, $couleur); | | Dessine un rectangle | | |
| ImagePolygon ($image, $array\_points, $nombre\_de\_points, $couleur); | | | $array\_points contient toute les coordonnées des x point $nombre\_de\_points | |
| **transparence** | | imagecolortransparent($image, $couleur); | | Indique quel $couleur on veut rendre transparent (ne fonctionne qu'avec PNG) | | |
| **Renvois taille** | | imagesx($source) **OU** imagesy($source) | | Renvois la taille x ou y de l'image | | |
| **Fusion 2 images** | | imagecopymerge($destination, $source, $destination\_x, $destination\_y, 0, 0, $largeur\_source, $hauteur\_source, 60); | | | | 0,0=tout/60=% transparence |
| **miniature** | | imagecopyresampled($destination, $source, 0, 0, 0, 0, $largeur\_destination, $hauteur\_destination, $largeur\_source, $hauteur\_source); | | | | |

## Expression régulières - recherche avec preg\_match (PCRE plus rapide et performant que POSIX)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rechercher**  **Un mot**  **Min/maj**  **OU**  **Début chaine**  **Fin chaine** | preg\_match("#Votre REGEX #", "Ce dans quoi vous faites la recherche"))  if (preg\_match("#GUITARE#", "J'aime jouer de la guitare.")) (...)  if (preg\_match("#GUITARE#i", "J'aime jouer de la guitare.")) (...)  if (preg\_match("#guitare|piano|banjo#i", "J'aime jouer de la guitare.")) (...)  #^Bonjour#  #zéro$# | | | | | Fonction renvoyant un booléen : true si l'expression est trouvé  Recherche le mot guitare (prend en compte maj/minuscule !!) // false ici  IDEM mais le i âpres le # rend insensible a la casse // true ici  Le symboles | symbolise un OU // renvois true  Recherche "Bonjour" en début de chaine (avec le ^au début)  Recherche " zéro " en fin de chaine (avec le $a la fin) |
| **Classe simple**  **Intervalle**  **Pas dans intervalle** | | #gr[ioa]s#  #[A-Z0-9]#  #[^0-9]#  #[a-zéèàêâùïüë]#  #[a-z?+\*{}]# | Recherche soit gris, gros ou gras, les lettre a l'intérieur des [] comme un OU  Recherche la présence de majuscules ou de chiffres  Vérifie qu'il n'y est PAS de chiffre avec le symbole ^ au début de l'intervalle// si pas de chiffre = true si des chiffre=false  Recherche une lettre entre a et z, plus les lettres écrit âpres z (éèàêâùïüë)  Recherche une lettre entre a et z, plus (?+\*{} pas besoin d'échapper entre [] sauf # ] et -) | | | |
| **quantificateur** | #a?# // #bor?is#  #a+#  #a\*#  #Ay(ay)\*#  #a{3}# / #a{3,}# / #a{3,5}# | | | ? (point d'interrogation) : ce symbole indique que la lettre est facultative. **Elle peut y être 0 ou 1 fois**.// boris ou bois = true  + (signe plus) : la lettre est obligatoire. **Elle peut apparaître 1 ou plusieurs fois**.  \* (étoile) : la lettre est facultative. **Elle peut apparaître 0, 1 ou plusieurs fois**.  Cherche Ay suivis de 0, 1 ou plusieurs fois (ay)  Le(s) chiffre(s) entre accolades donne l'intervalle dans l'ordre respectif : 3fois, 3fois et +, entre 3 et 5 fois | | |
| **échapper** | #\.# #\?# #\\*# | | | Echappe les caractères . ? et \* // liste caractères spéciaux : # ! ^ $ ( ) [ ] { } ? + \* . \ | | | |
| **Raccourcis** | **\d** | | | | Indique un chiffre. Ça revient exactement à taper [0-9] | |
| **\D** | | | | Indique ce qui n'est PAS un chiffre. Ça revient à taper [^0-9] | |
| **\w** | | | | Indique un caractère alphanumérique ou un tiret de soulignement. Cela correspond à [a-zA-Z0-9\_] | |
| **\W** | | | | Indique ce qui n'est PAS un mot. Si vous avez suivi, ça revient à taper [^a-zA-Z0-9\_] | |
| **\t** | | | | Indique une tabulation | |
| **\n** | | | | Indique une nouvelle ligne | |
| **\r** | | | | Indique un retour chariot | |
| **\s** | | | | Indique un espace blanc | |
| **\S** | | | | Indique ce qui n'est PAS un espace blanc (\t \n \r) | |
| **.**  #[0-9]-.#s | | | | Indique n'importe quel caractère **sauf les entrées** (\n). Il autorise donc tout !  Pour faire en sorte que le point indique tout, même les entrées, vous devrez utiliser l'option « **s** » | |

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Exemple** | | | #^[a-z0-9.\_-]+@[a-z0-9.\_-]{2,}\.[a-z]{2,4}$#  #^0[1-68]([-. ]?[0-9]{2}){4}$# | | Recherche une expression de type @ mail // ex nom.prenom@mail.fr renvois true  Recherche une expression de type numéro de téléphone fr // ex : 0X XX.XX-XXXX renvois true | | | | |
| **MySQL est compatible avec les regex mais seulement en POSIX : les diff : pas de délimiteur ## ni d'option; pas de raccourcis type /d sauf le . (n'importe quel caractères)** | | | | | | | | | |
| **Exemple** | | | SELECT nom FROM visiteurs WHERE ip REGEXP '^84\.254(\.[0-9] {1,3}) {2}$' | | | | | Recherche adresse IP commençant par 84.254. | |
| **Capture et remplacement avec preg\_replace /!\ les parenthèses sont capturantes !** | | | | | | | | | |
| **Remplacement**  **=> exemple**  **Cas** $0 | | | | $texte = preg\_replace('#\[b\](.+)\[/b\]#i', '<strong>$1</strong>', $texte);  **/!\** Ce texte est [b]important[/b], il faut me [b]comprendre[/b] !  #(anti)co(?:nsti)(tu(tion)nelle)ment# | | | Les parenthèses crée des variables : $1, $2 etc/ [b] sera remplacer par <strong>  Renverra : Ce texte est **important, il faut me comprendre** ! (voir U)  $0 contient toute la chaine dans les parenthèses | | |
| **parenthèses** | | | | #(anti)co(?:nsti)(tu(tion)nelle)ment# | | (?:) est une parenthèses standard (comme dans preg\_match) et non capturante | | | |
| **U** | $texte = preg\_replace('#\[i\](.+)\[/i\]#isU', '<em>$1</em>', $texte); | | | | **Le U de isU** permet de s'arrêter au premier [/i] rencontré / le s pour retour ligne / i pour maj/min | | | | |
| **Exemple** | | $texte = preg\_replace('#http://[a-z0-9.\_/-]+#i', '<a href="$0">$0</a>', $texte);  $texte = preg\_replace('#\[color=(red|green|blue|yellow|purple|olive)\](.+)\[/color\]#isU', '<span style="color:$1">$2</span>', $texte); | | | | | | | Rend tout les liens http://... cliquable  [ crée une balise pour colorer le texte dans une des couleurs qui  [se situe entre les () |

## Programmation orienté objet : les droits d'accès pour les Objet **Toutes les variables d'une classe doivent toujours être privées ou protégées.**

|  |  |
| --- | --- |
| public | tout le monde peut accéder à l'élément ; |
| private | personne (à part la classe elle-même) n'a le droit d'accéder à l'élément ; |
| protected | identique à private, sauf qu'un élément ayant ce droit d'accès dans une classe mère sera accessible aussi dans les classes filles. |

## Programmation orienté objet (il faut crée la class dans un fichier spécifique ex : Maclass.class.php et pas mettre ?> a la fin)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Créer classe**  **Variable**  **Fonction get**  **Fonction set**  **Envoyer un**  **e-mail** | class MaClasse {  private $pseudo;  public function getPseudo(){return $this->pseudo;}  public function setPseudo($newPseudo){$this->pseudo = $newPseudo}  public function envoyerEMail($titre, $message) {  mail($this->email, $titre, $message); }  } | | | | | Crée une class MaClasse **(commence par une majuscule !)**  Crée une variable $pseudo **(les variable tjrs en privée!)**  Crée une fonction (get), getPseudo() qui renvois pseudo  Crée une fonction (set), qui prend un argument et qui modifie le pseudo  Crée une fonction pour envoyez un mail  mail() est une fonction php qui permet d'envoyer un mail avec l'@ $this->email |
| **inclure** | include\_once('MesClass.class.php'); | Inclus le fichier contenant les class ( include\_once vérifie qu'il n'y soit pas déjà (ce qui provoquerais une erreur)) | | | | |
| **Crée un objet** | $membre = new MaClasse ();  $membre->setPseudo('monpseudo'); | | | | | Crée un objet a partir de la class MaClasse  Définie le pseudo (utilise la fonction de l'objet $membre hérité de la classe) |
| **constructeur** | public function \_\_construct($idMembre){ code} | | Un constructeur( \_\_construct) permet de crée un objet qui prend des paramètre (ex cherche dans la bdd) | | | |
| **destructeur** | public function \_\_destruct() { echo 'Cet objet va être détruit !'; }  unset($membre); | | | | Fonction appeler automatiquement par PHP pour détruire l'objet en fin de page ou il est  Détruit un objet manuellement | |
| **héritage** | class Admin extends Membre {fonctionalité suplémentaire} | | | Crée une nouvelle class fille Admin a partir de la class mère Membre avec + de fonctionnalités | | |

## Architecture mvc - modèle vue contrôleur

|  |  |
| --- | --- |
| C:\Users\Root\Desktop\382127.png | **Modèle** : cette partie gère les données de votre site. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc les requêtes SQL.  Parfois, les données ne sont pas stockées dans une base de données. C'est plus rare, mais on peut être amené à aller chercher des données dans des fichiers. Dans ce cas, le rôle du modèle est de faire les opérations d'ouverture, de lecture et d'écriture de fichiers (fonctions fopen, fgets, etc.).  **Vue** : cette partie se concentre sur l'affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple la liste des messages des forums.  **Contrôleur** : cette partie gère la logique du code qui prend des décisions. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès). |
| C:\Users\Root\Desktop\382129.png | Il faut tout d'abord retenir que le contrôleur est le chef d'orchestre : c'est lui qui reçoit la requête du visiteur et qui contacte d'autres fichiers (le modèle et la vue) pour échanger des informations avec eux.  Le fichier du contrôleur demande les données au modèle sans se soucier de la façon dont celui-ci va les récupérer. Par exemple : « Donne-moi la liste des 30 derniers messages du forum no 5 ». Le modèle traduit cette demande en une requête SQL, récupère les informations et les renvoie au contrôleur.  Une fois les données récupérées, le contrôleur les transmet à la vue qui se chargera d'afficher la liste des messages.  Le contrôleur sert seulement à faire la jonction entre le modèle et la vue finalement, non ?  Dans les cas les plus simples, ce sera probablement le cas. Mais comme je vous le disais, le rôle du contrôleur ne se limite pas à cela : s'il y a des calculs ou des vérifications d'autorisations à faire, des images à miniaturiser, c'est lui qui s'en chargera. |