**Une image contenant texte, clipart

Description générée automatiquementConcepteur Développeur d’Applications**

**Evaluation Rattrapage**

**PHP/HTML/CSS/JS : Application Web**

**Durée : 2 h**

**Documents autorisés : Oui**

**Internet autorisé : Oui**

**Consignes :**

Ce document est à compléter et à mettre à la racine de votre espace réseau dans le dossier **EVAL\_PHP\_Rattr\_[Prénom]**

L’évaluation porte sur **vos connaissances en matière de développement web**.

* Les réponses doivent être précises. Mettre des extraits courts de code (uniquement ceux qui répondent à la question), préciser le fichier et la méthode s’il y a lieu.
* La réponse 1.1 contient la réponse attendue pour exemple.

**Sujet :**

# Quels sont les points techniques qui permettent de passer de :

## La page connexion au catalogue (Exemple)

* L’actionConnexion dans PHP/CONTROLLER/ACTION traite les informations de connections, contrôle l’email et le mot de passe
* $uti =  UtilisateursManager::getList(null, ['adresseMail' => $\_POST['adresseMail']]);
* if ($uti != null) {
* if ($uti[0]->getMotDePasse() == crypte($\_POST['motDePasse']))
* La redirection avec le header
* header("location:index.php?page=Catalogue");
* La route dans Index.php charge la page catalogue avec
  + La présence de la route
* "Catalogue" => ["PHP/VIEW/LISTE/", "ListeCatalogue", "Catalogue", 1, false]
  + La méthode AfficherPage du fichier Outils.php dans PHP/Controller
* AfficherPage($routes[$page]);
  + Le fichier ListeCatalogue.php dans PHP/VIEW/LISTE

## La liste des articles au formulaire de modification

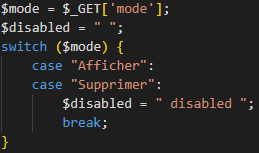
* Le lien dans la page ListeArticles.php fait passer la page FormArticles, le mode Modifier et l’ID de l’article à modifier
* La route dans Index.php charge la page de formulaire de modification avec
  + La présence de la route



* + La méthode AfficherPage du fichier Outils.php dans PHP/CONTROLLER



* Une fois sur la page FormArticles.php dans PHP/VIEW/FORM :
  + Le PHP détecte le mode passé par URL et donc ne désactive pas les champs



* + La présence d’une ID mène à la récupération de l’article concerné par la fonction findById du fichier ArticlesManager.Class.php dans PHP/MODEL/MANAGER



* + Cette fonction appel la fonction select du fichier DAO.Class.php dans le même dossier, en passant en paramètre que l’on veut toutes les informations possible sur l’élément, qu’il se trouve dans la table Articles et son ID.



* + La variable $elm et ensuite utilisé pour remplir les différents inputs et, du fait du mode, un bouton de validation est affiché pour valider le formulaire.



## Comment s’effectue le retour

* Le bouton de retour présent en bas de la page est un lien redirigeant sur la page ListeArticles
* La route dans Index.php charge la page avec
  + La présence de la route



* + La méthode AfficherPage du fichier Outils.php dans PHP/Controller



* + Le fichier ListeArticles.php dans PHP/VIEW/LISTE

# La liste des articles me fournit un IdTypeArticle, comment récupérer un texte ?

* On appel la fonction findById() de TypesArticlesManager.Class.php dans PHP/MODEL/MANAGER, en lui passant en paramètre l’IdTypeArticle de l’article en question
* Cette dernière nous retourne un objet de classe TypesArticles sur lequel on peut donc appeler la fonction ToString() présent dans le fichier TypesArticles.Class.php situé dans PHP/CONTROLLER/CLASSE



* Et ainsi récupérer son Libelle



# Dans le formulaire article

## Comment faire apparaitre l’image ?

* On récupère la valeur Photo associé à l’objet de la classe Articles et on l’utilise pour donner la source de l’image

## Comment transformer l’input TypeArticle pour avoir une liste déroulante ?

Utilisation de la fonction créerSelect()

## Quels fichiers, quelles méthodes entre en jeu ?

* On fait appel à la fonction creerSelect présente dans le fichier Outils.php dans le dossier PHP/CONTROLLER
  + On passe en paramètre l’ID de l’option qui devra être selectionnée,



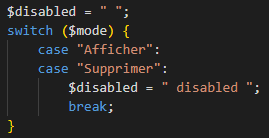
* + La table sur laquelle porte le select,



* + Les champs dont l’on veut voir apparaitre la valeur dans les Values des Options



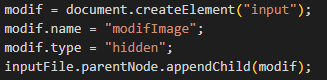
* + Et si l’on veut qu’il soit modifiable



Dans les cas de l’ID et du libelle, on fait appel à la fonction appelGet() situé elle aussi dans Outils.php dans PHP/CONTROLLER pour appeler le Getter voulu.

## Quels sont les modifications à apporter pour permettre le changement d’image ?

Le changement d’image est déjà effectif, mais il est géré par :

* L’utilisation d’un input de type File, permettant de choisir un fichier 
* Ce dernier est surveillé par le script FormArticles.js présent dans le dossier JS. Ce script :
  + Vérifie qu’un input de type File est bien présent et si oui, lui ajoute un EventListener appelant la fonction choixImage() présente à la suite
  + Cette fonction vérifie dans l’ordre :
    - Que l’input contient quelque-chose et
    - Qu’il s’agisse bien d’une image
  + Si tout est bon, elle crée un objet de type FileReader à partir de l’image
  + Prépare le changement de la source de l’image sur la page, et
  + Crée un input de type hidden ayant pour nom modifImage
  + La présence de ce dernier sera détectée lors du passage dans la page ActionArticles dans PHP/CONTROLLER/ACTION, entrainant la suppression de l’ancienne image, 
  + Et l’enregistrement de la nouvelle dans le dossier IMG sous un nom unique grâce à la fonction chargerImage du même fichier

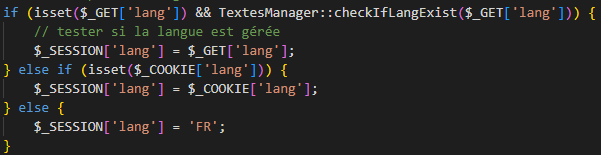


# Dans les classes, au niveau des setters, que veut dire ?int ?

Cela signifie que l’on veut que ces setters n’acceptent que des valeurs de type int ou des valeurs null.

# Traduction : quels sont les points techniques qui entrent en jeu ? (A tous les niveaux)

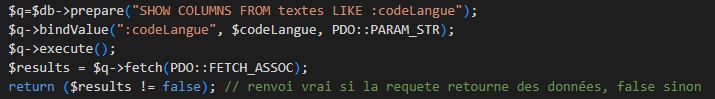
* Création d’une table dans la base de données associant un codeTexte à des valeurs différentes en fonction de la langue
* La gestion de la langue dans Index.php de la racine :
  + Vérifie si l’on vient de demander une langue et que cette dernière est « disponible », si un cookie contenant cette information existe déjà, ou est mis à FR par défaut



* + Puis création d’un cookie pour garder le choix en mémoire pour un prochain retour sur le site (<10h)



* + La confirmation qu’une langue est disponible se fait par l’appel de la fonction checkIfLangExist() du fichier TextesManager.Class.php dans PHP/MODEL/MANAGER, qui vérifie si la table Textes contient une colonne correspondante à la langue voulue et renvois true ou false en conséquence.

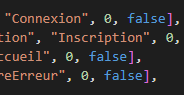


# Dans quel(s) contexte(s) sert le fichier DAO.Class.php

Pour des raisons de sécurité, il sert de fichier unique pour agir sur la base de données. Toutes les requêtes de récupération, de modification et de suppression de donnée des différentes classes passe par ce fichier.

# Utilisateurs : A quel endroit intervenir pour forcer le rôle et le mot de passe ?

Dans le fichier Index.php de la racine, les routes possèdent un entier à la 4ème position qui détermine le rôle minimal pour pouvoir accéder à la page en question.



Ce rôle minimal est ensuite géré par la fonction afficherPage du fichier Outils.PHP dans PHP/CONTROLLER où un rôle de niveau 0 est attribué à un utilisateur non logué,



S’il est logué, son rôle est à 1 du fait de l’enregistrement lors de l’inscription (ActionInscription.php dans PHP/CONTROLLER/ACTION)



Ou de la création « interne » (ActionUtilisateurs.php dans PHP/CONTROLLER/ACTION)

# Panier Utilisateur : Quelles sont les actions à mener pour récupérer les lignes du panier du client ?

* On fait appel à la fonction getList() de PaniersManager.Class.php dans PHP/MODEL/MANAGER, avec en paramètre l’idClient pour récupérer le panier du client voulu



* Cela entraine l’appel de la fonction select du fichier DAO.Class.php dans PHP/MODEL/MANAGER.
* A partir du panier, on récupère la liste de toutes les lignes associées

# Comment gérer les listes vides ? A quel(s) niveau(x) ?

Dans la fonction select de DAO.Class.php dans PHP/MODEL/MANAGER, on pourrait tester la longueur du tableau contenant les résultats et retourner false si elle vaut 0. Il suffira alors de prévoir l’obtention d’un retour valant false lors des appels des fonctions qui l’appellent (qui devrait déjà être en place du fait que l’on renvois false aussi si la requête présente un problème.)

# Comment fixer le footer en bas de la page et permettre de voir toute la page ?

Si on ne veut plus que le footer soit tout le temps affiché, on peut retirer le « position : fixed ; » dans le fichier style.css dans le dossier CSS



# Présentation sous forme de Grid :

## Où intervenir, comment le paramétrer ?

Le style portant sur la mise en forme de Grid se trouve dans le fichier grids.css du dossier CSS. On ajoutera les classes voulues dans les différentes balises.

## Comment obtenir une case qui fusionne 2 cases adjacentes ?

Utiliser la classe « grid-columns-span-2 »

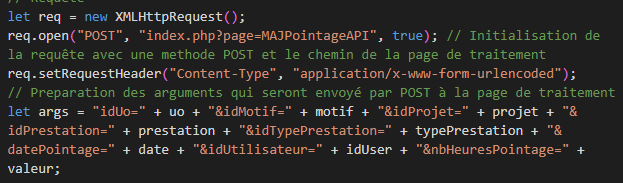
# Gestion des erreurs : que faut il faire pour mettre en place une gestion d’erreur ? A quel endroit ?

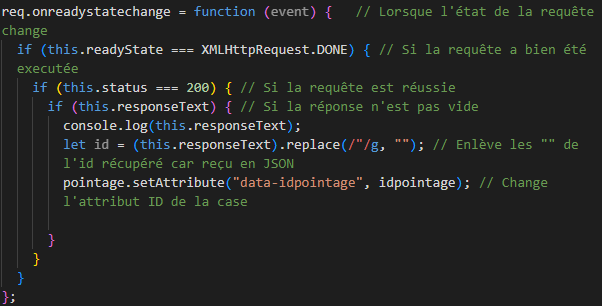
Utiliser des Try…Catch lors d’appel de fonctions, Afficher des messages clairs pour aider au débuggage.

# API : (Les extraits de code ne seront pas issus du projet, mais des projets réalisés en cours)

## Quels sont les différentes parties d’une API (nommer les parties et les langages dans lesquels ces parties peuvent être écrites ?

* Une partie en javascript qui crée une requête à l’intention de l’API



* La partie action qui reçoit la requête, la traite (en PHP ou C#) et renvois une réponse éventuelle en JSON
* Retour sur la partie JS où, après réception de la réponse, traitement des données éventuellement reçues en conséquence.

## Où mettre l’adresse du serveur d’API ?

Dans la création de la requête dans le JS, le deuxième paramètre de la fonction open().

## Comment passer des paramètres à l’API ?

Il faut les faire passer lors de l’appel de la fonction send() sur la requête.

## Comment passer d’une API asynchrone à une API synchrone ?

Il faut changer le troisième paramètre de true à false.