



Collège Technique **Aumôniers du Travail**
Enseignement de **promotion sociale**
185 Grand'rue 6000 Charleroi
Tel. 071.285.905

Projet de développement WEB

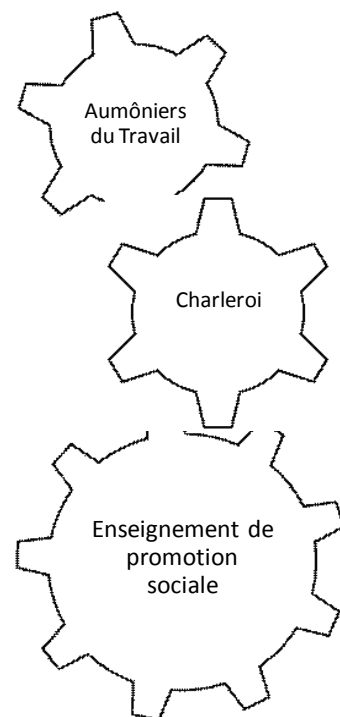
Code de l'unité de formation : **7534 30 U32 D1**

Partie(s) concernée(s) : **Exercices**

Domaine de formation :

- ***Bachelier en informatique de gestion***

Nom du chargé de cours : **Mr De Guglielmo**



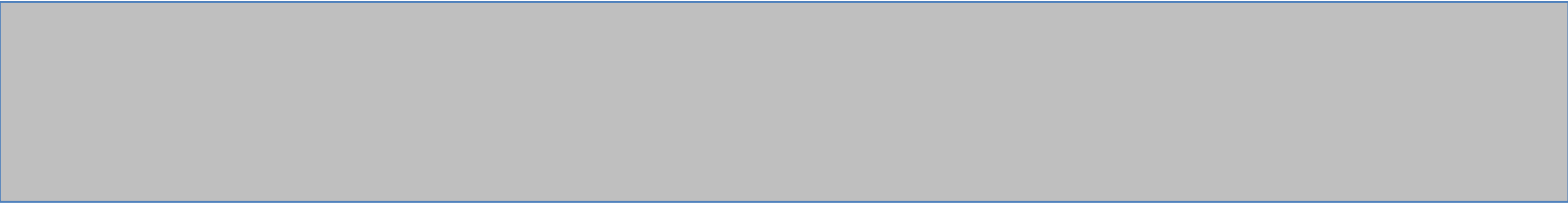


Table des matières

<u>Introduction</u>	3
<u>Serie 1 : Hello World</u>	4
<u>Serie 2 : Découpe de l'examen</u>	4
<u>Serie 3 : Les boucles</u>	4
<u>Exercice 1 :</u>	4
<u>Exercice 2 :</u>	4
<u>Exercice 3 :</u>	4
<u>Exercice 4 :</u>	5
<u>Série 4 : Les tableaux PHP</u>	6
<u>Exercice 1 :</u>	6
<u>Exercice 2 :</u>	6
<u>Exercice 3 :</u>	7
<u>Série 4 : Mise en place de Git</u>	8
<u>Série 5 : Formulaires et Session</u>	9
<u>Exercice 1</u>	9
<u>Exercice 2</u>	10
<u>Exercice 3</u>	10
<u>Exercice 4</u>	10
<u>Exercice 5</u>	10
<u>Série 6 : Les classes</u>	11
<u>Exercice 1</u>	11
<u>Exercice 2</u>	12
<u>Série 7 : Mise en place du MVC et connexion à la DB</u>	13
<u>Mise en place du MVC</u>	13
<u>Connexion à la DB</u>	14
<u>Série 8 : Mise en place de la structure pour afficher de l'information</u>	15
<u>Série 9 : Mise en place de l'accès DB</u>	17
<u>Série 10 : Recherches</u>	18
<u>Récapitulatif HTML5</u>	20

Introduction

En vue du développement de votre projet, vous allez avoir besoin de plusieurs outils :

- Wamp : Le serveur WEB.
- GIT : Le gestionnaire de sources.

Pour télécharger Wamp :

<http://www.wampserver.com/>

Créer un compte sur GitHub :

<https://github.com/>

Tous les exercices doivent être créés dans la racine de votre site

Chaque série correspond à une page web. Il vous sera précisé à chaque fois le nom de la page Web ainsi que son répertoire (vous comprendrez par la suite l'importance de cette arborescence).

Exemple → Série 1 doit correspondre à ceci :

projetweb2017/serie1/index.php

NB : Etc. Il vous est obligatoire que vos travaux soient synchronisés avec le Cloud (Git) de manière à ne **JAMAIS** rien perdre !

Serie 1 : Hello World

1 A la découverte de PHP, effectuez un echo de hello World

2 Créez une seconde page et utilisez la fonction require

Serie 2 : Découpe de l'examen

Reprenez l'examen d'HTML et effectuez une découpe selon le MVC.

Rendez dynamique le TITLE de votre page en utilisant des variables. Chaque page doit avoir un TITLE différent.

Serie 3 : Les boucles

- Dupliquez votre répertoire série 2 vers série 3 ;
- Ajoutez une page3.php et ajoutez cette nouvelle page dans le menu ;
- Cette nouvelle page va pouvoir accueillir le résultat des exercices suivants.

Exercice 1 :

Ecrire une boucle qui permet d'afficher dix phrases :

Ceci est la ligne n°1
Ceci est la ligne n°2
Ceci est la ligne n°3
Ceci est la ligne n°4
Ceci est la ligne n°5
Ceci est la ligne n°6
Ceci est la ligne n°7
Ceci est la ligne n°8
Ceci est la ligne n°9
Ceci est la ligne n°10

Exercice 2 :

Ecrire une boucle qui crée la liste à puce suivante :

- Ceci est la ligne N°1
- Ceci est la ligne N°2
- Ceci est la ligne N°3
- Ceci est la ligne N°4
- Ceci est la ligne N°5
- Ceci est la ligne N°6
- Ceci est la ligne N°7
- Ceci est la ligne N°8
- Ceci est la ligne N°9
- Ceci est la ligne N°10

Exercice 3 :

Ecrire une boucle qui génère le tableau dynamique suivant (pas forcément avec le même look)

Table dynamique	
#	Libellé
1	Ceci est la ligne N°1
2	Ceci est la ligne N°2
3	Ceci est la ligne N°3
4	Ceci est la ligne N°4
5	Ceci est la ligne N°5
6	Ceci est la ligne N°6
7	Ceci est la ligne N°7
8	Ceci est la ligne N°8
9	Ceci est la ligne N°9
10	Ceci est la ligne N°10

Exercice 4 :

Ecrire le tableau dynamique suivant :

-Les lignes paires doivent apparaitre en rouge, les autres en bleu.

Table dynamique	
#	Libellé
1	Ceci est la ligne N°1
2	Ceci est la ligne N°2
3	Ceci est la ligne N°3
4	Ceci est la ligne N°4
5	Ceci est la ligne N°5
6	Ceci est la ligne N°6
7	Ceci est la ligne N°7
8	Ceci est la ligne N°8
9	Ceci est la ligne N°9
10	Ceci est la ligne N°10

Série 4 : Les tableaux PHP

Dupliquez la série 3.

Ajoutez un nouveau point de menu « Tableaux »

C'est dans cette nouvelle page que vous allez effectuer les exercices.

Exercice 1 :

Créez un tableau indicé 1 dimension PHP reprenant les jours de la semaine et affichez le dans un tableau HTML.

Jour
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche

Exercice 2 :

Créez un tableau associatif pour chaque jour de la semaine.

→ Dans les éléments 'lundi' jusqu' 'vendredi', je dois retrouver la valeur école, pour samedi et dimanche on retrouvera 'maison'.

Le résultat doit être affiché dans la page :

Jour	Ce que je fais
Lundi	Ecole
Mardi	Ecole
Mercredi	Ecole
Jeudi	Ecole
Vendredi	Ecole
Samedi	Maison
Dimanche	Maison

Le contenu de la colonne « Jour » est l'association, Ecole est la valeur stockée dans la zone.

Exercice 3 :

Créez un tableau indicé de 52 éléments (= 52 semaines), dans chaque élément, on retrouvera tous les jours de la semaine comme à l'exercice précédent.

Semaine	Jour	Ce que je fais
1	Lundi	Ecole
	Mardi	Ecole
	Mercredi	Ecole
	Jeudi	Ecole
	Vendredi	Ecole
	Samedi	Maison
	Dimanche	Maison
2	Lundi	Ecole
	Mardi	Ecole
	Mercredi	Ecole
	Jeudi	Ecole
	Vendredi	Ecole
	Samedi	Maison
	Dimanche	Maison

Série 4 : Mise en place de Git

Pour la mise en place de la suite du cours, nous allons utiliser le gestionnaire de codes sources GitHub.

Pour cela, nous avons besoin de créer un compte sur la plateforme :

➔ www.github.com

Nous aurons aussi besoin d'un logiciel jouant le rôle de client GitHub de manière à gérer les interactions :

➔ <https://desktop.github.com/>

Lexique de Git :

➔ <http://www.robusta.io/content/tutoriel/git/start-git.html>

Le tutoriel pour découvrir GitHub :

➔ <https://openclassrooms.com/courses/gerer-son-code-avec-git-et-github>

Série 5 : Formulaires et Session

Dupliquez la série 4.

Exercice 1

Ajoutez une nouvelle page contenant un formulaire :

Les zones avec un « * » sont obligatoires

Nous contacter

Formulaire de contact

NOM

*Nom :

Nom, Prénom

EMAIL

*Email :

Email@monfournisseur.com

JESUIS

Je suis :

Particulier

MESSAGE

Notre message:

☐ Je veux recevoir la newsletter

NEWSLETTER

Envoyer

Valeurs de la liste déroulante :
formulaire :

Je suis :

Particulier

Professionnel

PART

PROF

Valeurs envoyées par le

Liste des valeurs envoyées

Nom	Valeur
"NOM"	"nom, prénom"
"EMAIL"	"roberto.deguglielmo@promsocatc.net"
"JESUIS"	"PROF"
"MESSAGE"	"Bon travail à tous!"
"NEWSLETTER"	"1"

Chargé de cours:
De Guglielmo Roberto

Votre formulaire doit pointer sur cette nouvelle page.

Lorsque votre page va recevoir le formulaire, elle va recharger les informations dans le formulaire.

De cette manière, vous pourrez recharger votre formulaire en cas d'erreur.

Exercice 2

Les tests suivants doivent être faits :

-La zone nom doit reprendre une virgule : XXX , XXX

-La zone Email doit reprendre un @ ainsi qu'un point.

Pour chaque erreur repérée, un libellé en rouge doit apparaître en dessous de la zone testée.

Exercice 3

Si le formulaire est correct, la zone NOM doit être enregistrée dans la session dans la zone « UTILISATEUR_NOM ». La zone « UTILISATEUR_OK » doit être = 1.

Exercice 4

Dans le Header :

-Si la zone « UTILISATEUR_OK » est affectée à 1 , affichez « Bienvenue » ainsi que le contenu de la zone « UTILISATEUR_NOM ».

Exercice 5

Ajoutez un bouton « se déconnecter » juste à côté du « Bienvenue NOM, Prénom ».

Ce bouton est un formulaire qui va pointer vers une nouvelle page php réceptrice de votre formulaire doit détruire la session et faire un require sur la première page de votre site.

Série 6 : Les classes

Comme à l'habitude, dupliquez le projet 5.

Exercice 1

Ajoutez une nouvelle page (avec son point de menu, etc...) dans laquelle vous allez placer votre combat.

Vous allez aussi créer une nouvelle classe PHP Personnage.

→ Pour rappel, 1 classe php = un fichier php !

Utilisez le tutoriel classroom pour faire combattre les deux personnages.

<https://openclassrooms.com/courses/programmez-en-orienté-objet-en-php/utiliser-la-classe>

Le résultat attendu est celui-là :

Que le combat commence !

Le personnage 1 a 50 de force, contrairement au personnage 2 qui a 100 de force.
Le personnage 1 a 2 d'expérience, contrairement au personnage 2 qui a 23 d'expérience.
Le personnage 1 a 105 de dégâts, contrairement au personnage 2 qui a 73 de dégâts.

Voici l'enchaînement des opérations :

```
mp
$perso1= new Personnage(50,5);
$perso2= new Personnage(100,23);
$perso1->frapper($perso2);
$perso1->gagnerExperience();
$perso2->frapper($perso1);
$perso2->gagnerExperience();
```

```
echo 'Le personnage 1 a ', $perso1->getforce(), ' de force, contrairement au personnage 2 qui a ', $perso2->getforce(), ' de force.<br />';
```

```
echo 'Le personnage 1 a ', $perso1->getexperience(), ' d\'expérience, contrairement au personnage 2 qui a ', $perso2->getexperience(), ' d\'expérience.<br />';
```

```
echo 'Le personnage 1 a ', $perso1->getdegats(), ' de dégâts, contrairement au personnage 2 qui a ', $perso2->getdegats(), ' de dégâts.<br />';
```

Exercice 2

Vous allez créer une nouvelle classe. Celle-ci va vous permettre de générer le code HTML de vos formulaires. Il s'agit d'une classe qui sera instanciée. Cela va vous permettre d'avoir différents objets formulaires.

Nous allons donc avoir besoin d'une classe « Form ». Lors de son instanciation, les paramètres suivant doivent être passés : name ; id ; méthode ; action.

Pour ajouter des éléments au formulaire, nous allons avoir besoin de fonctions telles que setText et setEmail donc le prototype est le suivant : label ;

Name ; id ; required ; placeholder ; value.

Pour disposer d'un bouton submit, vous allez créer une fonction setSubmit dont le prototype est : name ; value.

Une fonction getForm va vous permettre de récupérer le code HTML correspondant au formulaire ainsi généré.

Utilisez cette classe pour créer un nouveau formulaire contenant une zone nom et Email idem à la série 5.

Nous contacter

Formulaire de contact

NOM

*Nom :
Nom, Prénom

EMAIL

*Email :
Email@monfournisseur.com

Liste des valeurs envoyées

Nom	Valeur
"NOM"	"nom, prénom"
"EMAIL"	"roberto.deguglielmo@promsocatc.net"

Série 7 : Mise en place du MVC et connexion à la DB

→ Dupliquez la série 6 en série 7.

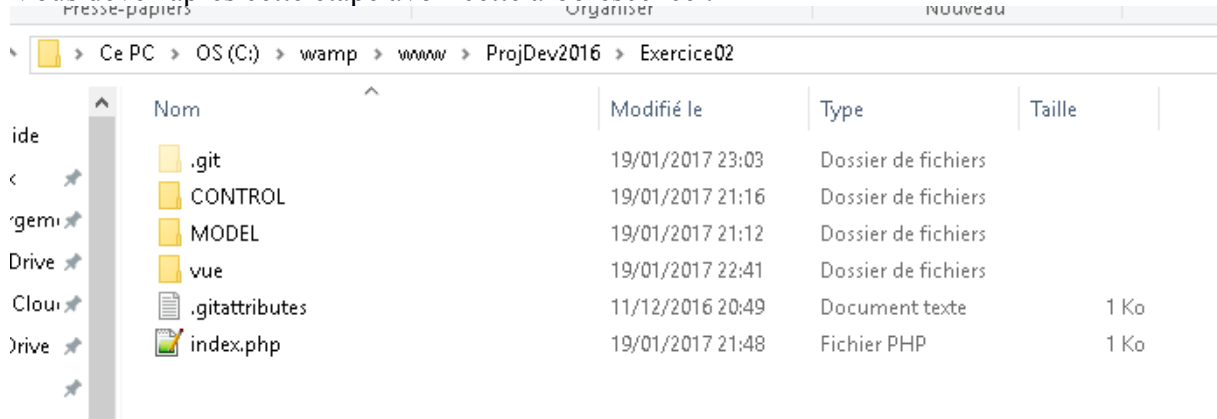
Mise en place du MVC

Étape 1 : Déplacer le contenu du répertoire dans un nouveau nommé « VUE »

Étape 2 : Créer les répertoires CONTROL et MODEL au même niveau que la vue

Étape 3 : Les page1.php et page2.php n'ont plus besoin de leurs include « Haut/up.php » et « bas/down.php »

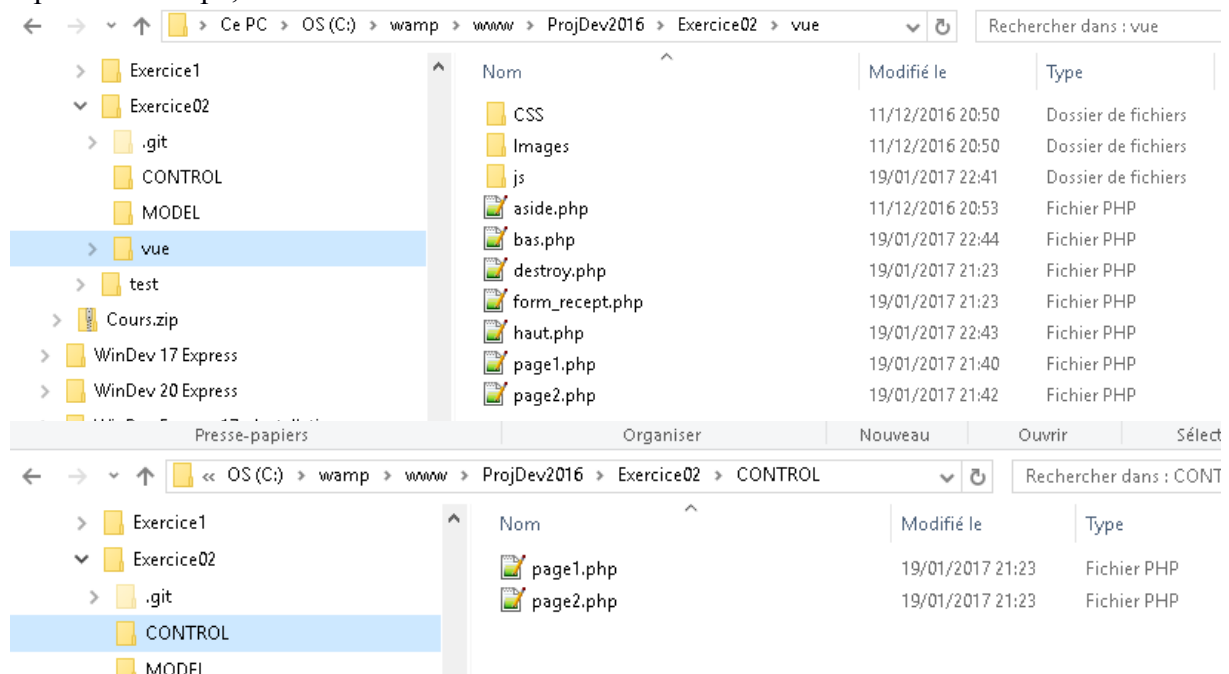
Vous devez après cette étape avoir cette arborescence :



Nom	Modifié le	Type	Taille
.git	19/01/2017 23:03	Dossier de fichiers	
CONTROL	19/01/2017 21:16	Dossier de fichiers	
MODEL	19/01/2017 21:12	Dossier de fichiers	
vue	19/01/2017 22:41	Dossier de fichiers	
.gitattributes	11/12/2016 20:49	Document texte	1 Ko
index.php	19/01/2017 21:48	Fichier PHP	1 Ko

Étape 4 : Dans vue, créer les fichiers page1.php et page2.php. Ces pages se chargeront de faire les includes nécessaires des fichiers se trouvant dans la vue.

Après cette étape, vous devez avoir l'arborescence suivante :



Nom	Modifié le	Type
CSS	11/12/2016 20:50	Dossier de fichiers
Images	11/12/2016 20:50	Dossier de fichiers
js	19/01/2017 22:41	Dossier de fichiers
aside.php	11/12/2016 20:53	Fichier PHP
bas.php	19/01/2017 22:44	Fichier PHP
destroy.php	19/01/2017 21:23	Fichier PHP
form_recept.php	19/01/2017 21:23	Fichier PHP
haut.php	19/01/2017 22:43	Fichier PHP
page1.php	19/01/2017 21:40	Fichier PHP
page2.php	19/01/2017 21:42	Fichier PHP

Nom	Modifié le	Type
page1.php	19/01/2017 21:23	Fichier PHP
page2.php	19/01/2017 21:23	Fichier PHP

Étape 6 : Du fait du changement d'arborescence, tous les liens doivent être corrigés :

- Le menu : Les liens doivent maintenant pointer vers le contrôleur (Exemple: « ../control/page1.php »)
- Les liens vers les images, et CSS doivent être « ../vue/images/image.jpg »

Connexion à la DB

Vous disposez d'une archive ZIP reprenant différents fichiers. Il s'agit du Relational Mapping qui est à intégrer dans votre projet.

Vous retrouverez donc les fichiers php nécessaires pour faire fonctionner un MVC, mais aussi le répertoire MODEL permettant d'accéder à la base de données.

Faites dérouler votre script SQL sur votre phpMyAdmin, configurez les users et mot de passe dans Model.php et faites fonctionner l'exemple.

Une fois l'exemple fonctionnel, regardez à l'intégrer dans votre projet série 7 (Dans la dernière page de votre menu).

Série 8 : Mise en place de la structure pour afficher de l'information

Vous allez dupliquer la série 7.

L'objectif de cette série est de structurer votre code pour standardiser la manière dont vous allez utiliser votre contrôleur pour accéder à l'information (grâce au modèle) et l'afficher (grâce à la vue).

Exemple :

Pour un fichier donné « TABLETEST », vous allez donc avoir une classe dans le modèle (jusque-là rien ne change). L'objectif est d'afficher le contenu de cette table dans une page.

Pour cela, nous allons créer un fichier « TABLETEST_TAB.php » dans le contrôleur ainsi que dans la vue.

Dans le contrôleur, nous allons avoir ceci « TABLETEST_TAB.php » :

```
<?php
    require_once '../control/core.php' ;
?>

<?php
    $Tabletest=Model::load("tabletest");
    $Tabletest->read();
    require '../vue/TABLETEST_TAB.php' ;
?>
```

Et dans la vue « TABLETEST_TAB.php »:

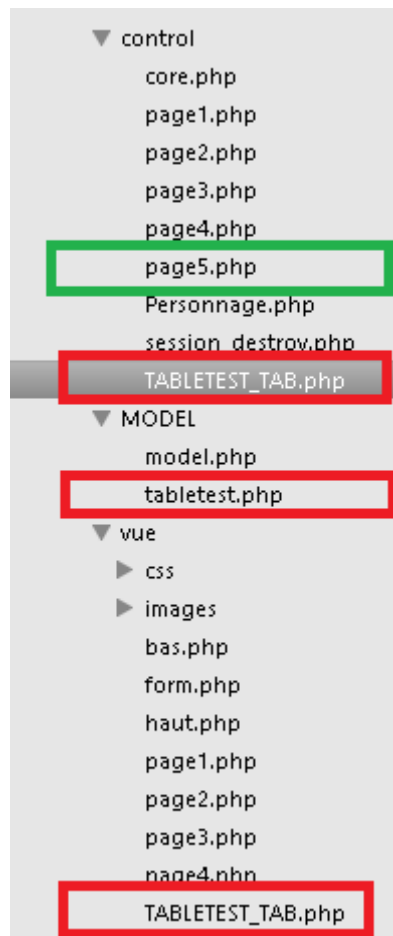
```
1 <?php
2 foreach($Tabletest->data as $k){
3     echo $k->NOM .', ' . $k->PRENOM . '<br>';
4 }
5 ?>
```

De cette manière, il vous suffira d'ajouter un require de votre /control/TABLETEST_TAB.php pour disposer de l'affichage du contenu de votre table.

Ajoutez un nouveau point de menu ainsi qu'une nouvelle page dans le contrôleur (ainsi que dans le point de menu et faites un require de votre nouvelle page.

Voilà à quoi cela doit ressembler au final :

Ma nouvelle page 5 qui est dans mon point de menu, ainsi que mes fichiers TABLETEST_TAB + tabletest.php



Série 9 : Mise en place de l'accès DB

Dupliquez la série 8 !

Sur base du document fourni en annexe (Gestion des sessions et authentification):

1. Nous allons devoir créer la table utilisateurs dans la base de données

```
CREATE TABLE `utilisateurs` (  
  `utilisateur` varchar(255) NOT NULL,  
  `code` varchar(255) DEFAULT NULL,  
  `nom` varchar(255) DEFAULT NULL,  
  `prenom` varchar(255) DEFAULT NULL,  
  `admin` int(11) DEFAULT NULL,  
  `actif` int(1) DEFAULT NULL,  
  PRIMARY KEY (`utilisateur`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

```
INSERT INTO `test`.`utilisateurs` (`utilisateur`, `code`, `nom`, `prenom`, `admin`,  
  `actif`) VALUES ('Administrateur', 'code', 'De Guglielmo', 'Roberto', '1', '1');
```

2. Ajouter un enregistrement pour l'utilisateur Administrateur
3. Créer le fichier utilisateurs dans le répertoire model.
4. Créer le fichier login.php dans le répertoire vue.
 - a. Il reprendra le formulaire nécessaire pour l'authentification
5. Créer le fichier login.php dans le répertoire contrôleur
 - a. C'est lui qui va recevoir le formulaire de login, il se chargera de vérifier l'authentification et ainsi de préparer la session de manière à débloquent l'accès au site.
6. Modifier core.php
 - a. Démarrer la session
 - b. Si l'url courante n'est pas *login.php et que la variable 'UTILISATEUR' de la session n'est pas garnie, redirection automatique vers control/login.php
7. Modifier vue/haut.php
 - a. Conditionner l'affichage du menu uniquement si la variable UTILISATEUR' de la session est pas garnie
8. Ajouter deux fonctions statiques à la classe Utilisateur pour se charger de la vérification de l'utilisateur, ainsi que de son niveau d'autorisation.

Série 10 : Recherches

Nous allons maintenant faire évoluer notre application de manière à pouvoir effectuer des recherches.

Pour cela, nous allons tout d'abord devoir faire évoluer notre classe **Model**

→ Elle doit permettre d'accueillir une clause **Where**. Pour cela, nous allons ajouter un paramètre \$where à la fonction READ.

```
public function read($fields=null){  
    if($fields==null){  
        $fields = '*';  
    }  
    if ($this->id== null){  
        $sql= 'SELECT '.$fields.' from '.$this->table ;  
    }  
    else{  
        $sql= 'SELECT '.$fields.' from '.$this->table .' where '.$this->PK.' = '.$this->id ;  
    }  
}
```

Il doit être ajouté une condition supplémentaire dans le cas où l'ID est null. Si le contenu de la variable \$where est différent de null et ' ', on ajoute la clause where ainsi que le contenu de la variable dans \$sql.

Il est conseillé à ce point d'effectuer un test pour vérifier que la classe fonctionne correctement. Pour cela, ajoutez un paramètre dans le read présent dans tablettest_tab.php de manière à demander « prenom = 'roberto' »

```
<?php  
    $Tabletest=Model::load("tabletest");  
    $Tabletest->read()  
    require '../vue/TABLETEST_TAB.php' ;  
?>
```

Visualisez le point de menu de manière à vérifier que le contenu affiché correspond.

Maintenant que cela fonctionne, nous allons ajouter un formulaire dans cette même page.

(Veillez à utiliser l'objet form !). Le formulaire doit pointer vers la page5.php.

Ensuite, veillez à ajouter la gestion de la récupération du formulaire de manière à créer la String SQL et la passer à la fonction read() .

→ \$where.= " upper(prenom) like upper('%".\$VarPrenom."%')";

...Voilà, cela devrait fonctionner !

Série 11 : Exercices JQuery

Exercices

Ressources :

https://www.w3schools.com/jquery/jquery_events.asp

<https://openclassrooms.com/courses/un-site-web-dynamique-avec-jquery?status=published>

Sur base de la feuille Excel, effectuez l'exercice et consignez les résultats dans celle-ci.
Pour mener à bien l'exercice, vous devez aller à l'URL :

<http://dero-promsocatc.alwaysdata.net/jquery/>

Exercice à mettre en place dans le projet

Etape 1 : Créer le répertoire JS dans la vue. Ce répertoire va contenir tous nos scripts JS

Etape 2 : Créer une Init.js dans ce nouveau répertoire. Celui-ci reprendra la déclaration de nos bibliothèques ainsi qu'un include des JS dont nous allons avoir besoin.

Etape 3 : Ajouter le require d'init.js dans notre « bas/down.php »

-Lors d'un clic sur H2, je veux qu'en bas du ASIDE soit ajouté une ligne reprenant la classe et le contenu du titre.

-Lors d'un clic sur un paragraphe, emballer celui-ci pour qu'il soit en italique.

-Lors d'un clic sur une image :

- Ajouter une nouvelle <div> au body
- Effectuer un Fade sur les sections et aside
- Effectuer un fadein sur la div
- Dans la div, ajouter l'image
- Ajouter une zone cliquable permettant de supprimer cette div et annuler le Fade sur section + aside.

La vie à l'école

décès Abbé Vanderus



otre Président de PO a tiré sa révérence en ce début de mois de juillet. atif de Charleroi, l'Abbé Jacques Vanderus était un Aumônier du travail pass ollège Technique des ATC. Il cumula ainsi la gestion d'écoles organisant des es années, il es et supéri

Tableau

est Table

Titre 1	Titre 2	Titre 3	Titre 4	Titre 5
Cel 1	Cel 2	Cel 3	Cel 4	Cel 5
Ligne 1	Ligne 1	Ligne 1	Ligne 1	Ligne 1
Col 1	Col 2	Col 3	Col 4	Col 5
Cel 6	Cel 7	Cel 8	Cel 9	Cel 10
Ligne 2	Ligne 2	Ligne 2	Ligne 2	Ligne 2

Intégrer dans sa page un lien pour remonter en haut de page.

Vous allez intégrer dans votre code sur base des informations disponibles sur le site suivant.

<http://www.paulund.co.uk/how-to-create-an-animated-scroll-to-top-with-jquery>

→ Ajoutez-y le Logo ATC une fois l'intégration terminée

Enseignement de Promotion Sociale

[Accueil](#)
[Contact](#)
[Recherches](#)
[Employés](#)
[Clients](#)

Nos formations

favoriser la démarche d'insertion socio-professionnelle, cette formation poursuit le but de développer n qualifiante. En particulier, cette formation s'oriente vers la restauration de mobilier et les apprentissages

n oeuvre la diversité méthodologique des différentes fonctions de l'informaticien en réponse aux besoins





Point d'informations

Fin des inscriptions

20 octobre 2015 16:29

Les inscriptions pour l'année scolaire

Récapitulatif HTML5

BALISE		ATTRIBUT / DESCRIPTION	
<!-- Commentaire -->		Commentaire	
<!DOCTYPE html>		Déclaration du Doctype	
BALISE DE PREMIER NIVEAU – Code minimal d'une page web			
<html>		Indiquez au navigateur que le document est en HTML	
<head>		<body>	Corps de la page
BALISE D'EN-TÊTE – Entre les balises <head>			
<title>		Titre de la page, tres important pour le référencement	
<script>		Inserer script	<style> Inserer CSS
<noscript>		Message à afficher si le script n'est pas toléré	
<base />		URL par défaut <base href="http://41mag.fr/" target="_blank" />	
<link />	CSS	<link rel="stylesheet" type="text/css" href="#" />	
	PAGE	<link rel="start" href="index.html" />	
	RSS	<link rel="alternate" type="application/rss+xml" href="#" />	
	FAVICON	<link rel="shortcut icon" type="image/x-icon" href="#" />	
<meta />	TITLE		Titre de la page
	DESCRIPTION		Description de la page
	KEYWORDS		Mots-clés de la page
	ROBOTS		Restrictions pour les robots
	SYNDICATION-SOURCE		Indique l'URL d'origine
	ORIGINAL-SOURCE		Indique que c'est l'original
	NOTRANSLATE		Ne sera pas traduit
	HTTP-EQUIV CHARSET		Jeux de caractères
	HTTP-EQUIV REFRESH		Raffraichir la page
HTTP-EQUIV PRAGMA		Définit le cache du navig.	
BALISE D'ARCHITECTURE			
<header>		Définit l'en-tête d'une section ou d'une page	
<footer>		Définit le bas d'une section ou d'une page	
<hgroup>		Définit les informations d'en-tete d'une section ou d'une page	
<details>		Définit les détails d'un élément	
<summary>		Définit l'en-tête des details d'un élément	
<menu>		Définit un menu en forme de liste	
<section>		Définit une section	
<article>		Définit un article	
<aside>		Définit un élément latéral	
<nav>		Définit un groupe de liens de navigations	
<iframe>		Introduis une page html dans une frame	
<div>		Calque ou section	
		Section de type inline	
BALISE DE STRUCTURATION DE DE TEXTE			
<h1> à <h6>		Créer un titre	
<p>		Paragraphe	<a> Lien
		Mise en exergue	 Texte en gras
		Italique en exergue	<i> italique
<mark>		Marqueur de texte	<small> Rétrécis le texte
<sub>		Mise en indice	<sup> Mise en Exposant
<adress>		Définit une adresse	<cite> Citation
<abbr>		Abréviation	<dfn> Définition
		Texte supprimé	<ins> Texte ajouté
<time>		Date / Horaire	<meter> Mesure
<code>		Portion de code	<pre> Texte préformaté
 		Saut de ligne	<wbr> Empl pr saut 2 ligne
<blockquote>		Longue citation	<q> Courte citation
<samp>		Echantillon	<var> Variable
<kbd>		Raccourci clavier	<bdo> Sens du texte

BALISE	ATTRIBUT / DESCRIPTION			
BALISE DE LISTE				
	Liste non-ordonné		Liste ordonné	
	Élément de liste	<dl>	Liste de définition	
<dt>	Terme à définir	<dd>	Définition du terme	
BALISE DE TABLEAU				
<table>	Tableau	<caption>	Titre du tableau	
<thead>	En-tête du tableau	<tbody>	Corps du tableau	
<tr>	Ligne du tableau	<th>	Cellule d'en-tête	
<td>	Cellule du tableau	<tfoot>	Bas du tableau	
<col>	Colonne du tableau	<colgroup>	Groupe de colonne	
BALISE DE FORMULAIRE				
<form>	Formulaire	<fieldset>	Regroupe plusieurs élémnts du formulaire	
<legend>	Titre d'un groupe	<label>	Titre d'un élément	
<datalist>	Liste déroulante	<select>	Liste selectionnable	
<option>	Élément d'une liste	<optgroup>	Grp d'élmnets d'une list	
<textarea>	Zone de texte	<keygen>	Génération d'une clé	
<button>	Bouton cliquable	<command>	Bouton de commande	
<output>	Définit un type de sortie			
<input />	button	file	radio	text
	checkbox	hidden	range	time
	color	image	reset	url
	date	month	search	
	datetime	number	submit	
	email	password	tel	
BALISE MULTIMEDIA				
<area>	Zone cliquable à l'interieur d'une image			
<audio>	Contenu audio	<canvas>	Graphique	
	Image ou photo	<progress>	Progression	
<figure>	Groupe d'element multimédia	<figcaption>	Légende du groupe d'élément	
<video>	Vidéo	<source>	Source du media	
<map>	Carte / image	<param>	Parametre d'objet	
<embed />	Contenu extérieur	<object>	Objet du cont. ext	
AUTRE				
<rp>	Annotation pr le nav	<rt>	Explication ruby	
<ruby>	Annotation ruby	<hr />	Barre horizontale	

ATTRIBUT STANDAR			
Accesskey	Raccourci clavier	itemprop	Utilisé pr un gp d'élément
class	Attribut une classe	context-menu	Menu contextuel d'élément
lang	Langage d'un élément	spellcheck	Correction automatique
data-	Définit un attribut	style	Applique un style
dir	Direction du texte	subject	Définit l'élément correspondt
draggable	Élément déplaçable	tabindex	Définit l'ordre d'un tablo
hidden	Élément caché	title	Titre de l'élément
id	Nomme un élément	contenteditable	Élément éditable
item	Utilisé pour un groupe d'élément		

EVENEMENT			
Pour la balise <body>	Pour formulaire	Pour la souris	Pour les medias
Onafterprint, onbeforeprint, onafterload, onblur, onerror, onfocus, onhaschange, onload, onmessage, onoffline, ononline, onpagehide, onpageshow, onpopstate, onredo, onresize, onstorage, onundo, onunload	Onblur, onchange, oncontextmenu, onfocus, onformchange, onforminput, oninput, oninvalid, oninvalide, onsubmit, onselect	Onclick, ondblclick, ondrag, ondragend, ondragenter, ondragleave, ondragover, ondragstart, ondrop, onmousedown, onmouseup, onmousemove, onmouseout, onmouseover, onscroll	Onabort, oncanplay, oncanplaythrough, ondurationchange, onemptied, onended, onerror, onloadeddata, onloadstart, onpause, onplay, onplaying, onprogress, onseeked, onsuspend, onwaiting, onvolumechange