

PHP - Partie Pratique

Support de cours IUT de Calais

Kevin GUERRIER

Préambule

Lien vers le support de cours

<https://bit.ly/2024-iut-php-cm-kguerrier>

Lien vers le support de pratique

<https://bit.ly/2024-iut-php-tds-kguerrier>

Ce support est fortement inspiré de la documentation officielle disponible sur

<https://www.php.net/manual/fr>

Votre Environnement de Développement avec VSCode

De nombreuses extensions peuvent vous accompagner dans vos développements sous VSCode, et elles pourront varier selon les préférences de chacun.

Cependant, je vous propose de trouver des extensions qui vont vous permettre :

- de vous assister dans la complétion de code
- de vous faciliter la lecture avec des colorations syntaxiques (voir personnalisables)
- de vous permettre d'aller directement vers les définitions des ressources externes (fonctions, classes, variables, ...)

LE sujet brûlant : les extensions d'Intelligence Artificielle !

L'un des aspects les plus critiques à prendre en compte lors de l'utilisation d'outils d'IA est la confidentialité des données. Les systèmes d'IA fonctionnent non seulement sur la base d'une grande quantité de données externes pour fonctionner efficacement mais ont également besoin des portions de code que vous êtes en train de développer, et cela peut poser des risques significatifs si ces données contiennent des informations sensibles ou confidentielles. Il est donc essentiel d'être toujours conscient du fonctionnement de ces outils et de s'assurer de ne les utiliser que sur des portions de codes non critiques.

Il est évident que des extensions comme [GitHub Copilot](#) peuvent être des atouts importants lors de vos développements ! Pour celles et ceux qui ne la connaissent pas encore, GitHub Copilot est un assistant de codage innovant alimenté par l'IA qui est développé par GitHub et OpenAI. Il utilise des modèles d'apprentissage automatique formés sur une grande quantité de code pour fournir des suggestions et des compléments de code intelligents.

Cependant, vous êtes ici dans une démarche d'apprentissage et les exercices qui vous sont demandés ont pour but de VOUS apprendre à coder. C'est grâce à cet apprentissage et au développement de votre expertise en tant que développeur que vous pourrez alors regarder les codes générés par copilot avec un regard critique.

Aussi, merci de ne pas utiliser ces extensions lors de vos différents travaux ;-)

A vous de trouver des extensions qui VOUS conviennent

VSCode permet la recherche et l'installation d'une extension directement depuis son interface, par le menu à gauche (ou le raccourci CTRL + Shift + X) qui vous permet de parcourir une librairie complète à partir de mots clés et d'installer facilement une nouvelle extension.

Bien sûr, vous pouvez de la même manière choisir de la désactiver / désinstaller. Et vous pourrez également avoir accès aux interfaces de configuration qui vous permettront de personnaliser ces extensions.

Globalement, peu importe le langage, il est important pour vous d'être complètement à l'aise avec votre station de travail : que ce soit concernant les services qui fonctionnent sur votre machine, sur leurs configurations et monitoring, mais également sur votre environnement de développement. Donc trouvez les extensions qui VOUS correspondent, soyez à l'aise avec les raccourcis claviers (et personnalisez-les si besoin) pour vous faire gagner du temps.

Organisation du Code

Il est recommandé de structurer son code PHP en utilisant des fichiers séparés pour la logique, les fonctions réutilisables, et la présentation (HTML). Par exemple :

- index.php : Point d'entrée principal.
- inc/header.php : En-tête HTML.
- inc/footer.php : Pied de page HTML.
- conf/config.php : Fichier de configuration.
- lib/functions.php : Ensemble de fonctions utilitaires.

Cela vous permettra de mieux comprendre, organiser, transmettre et reprendre les projets sur lesquels vous travaillez. Pensez cependant à protéger également la lecture des sous-répertoires pour éviter que le serveur web ne puisse transmettre la liste des fichiers présents. Une bonne pratique est d'y placer un fichier index.php vide ou qui redirige vers une page connue et sécurisée.

Qualité du Code et Respect des Standards

Comme dans tous les langages, la qualité du code que vous écrivez est primordiale, et PHP ne fait pas exception !

Aussi, il est important de toujours garder en tête les notions de lisibilité, de clarté, de simplicité et surtout de rester cohérent tout au long de vos développements !

De plus, vous verrez que lorsque votre code est de qualité, alors les phases de tests et de débogage sont plus aisées, les phases de maintenance et d'évolution se font également plus facilement, tout comme le travail collaboratif ! Et pour cela, respecter une norme de codage commune facilite énormément les choses.

Pour vous accompagner sur ce sujet, des éléments normés ont été établis : les PSRs, pour PHP Standards Recommendations. Elles sont des normes établies pour promouvoir des pratiques de codage cohérentes au sein de la communauté PHP. Elles couvrent divers aspects du développement PHP, de la structure du code aux modalités d'autoloading.

Elles ont été établies par PHP Framework Interop Group (PHP FIG) et sont pleinement détaillées sur le site : <https://www.php-fig.org/>

La liste complète des PSRs est disponible sur <https://www.php-fig.org/psr/> et est découpée en plusieurs parties

PSR-1 : Basic Coding Standard

Objectif

PSR-1 comprend ce qui doit être considéré comme les éléments de codage standard qui sont nécessaires pour garantir un niveau élevé d'interopérabilité technique entre le code PHP partagé.

Quelques principes clés

- Les fichiers DOIVENT utiliser uniquement les balises `<?php` et `<?='`
- Les fichiers DOIVENT utiliser uniquement UTF-8 sans BOM pour le code PHP
- Les fichiers DOIVENT soit déclarer des symboles (classes, fonctions, constantes, etc.) soit provoquer des effets secondaires (par exemple, générer une sortie, modifier les paramètres .ini, etc.) mais NE DOIVENT PAS faire les deux.
- Les espaces de noms et les classes DOIVENT suivre un PSR « à chargement automatique » : [~~PSR-0~~ (abandonné) , PSR-4].
- Les noms de classe DOIVENT être déclarés en StudlyCaps.
- Les constantes de classe DOIVENT être déclarées en majuscules avec des séparateurs de soulignement
- Les noms de méthodes DOIVENT être déclarés en camelCase.

```
<?php
// Classe en PascalCase
class MaClasse
{
    // Méthode en camelCase
    public function maMethode()
    {
        // Code
    }

    // Constante en UPPER_SNAKE_CASE
    const MA_CONSTANTE = 'valeur';
}
```

Tous les aspects présents sur : <https://www.php-fig.org/psr/psr-1/>

PSR-12 : Extended Coding Style

Objectif

Cette spécification étend, développe et remplace PSR-2, le guide de style de codage et exige le respect de PSR-1, la norme de codage de base.

Comme PSR-2, l'objectif de cette spécification est de réduire les difficultés de compréhensions lors de l'analyse de code provenant de différents auteurs. Elle le fait en énumérant un ensemble partagé de règles et d'attentes sur la manière de formater le code PHP.

Cette PSR cherche à fournir un ensemble de méthodes que les outils de style de codage peuvent mettre en œuvre, auxquelles les projets peuvent déclarer leur adhésion et auxquelles les développeurs peuvent facilement s'identifier entre différents projets.

Lorsque plusieurs auteurs collaborent sur plusieurs projets, il est utile de disposer d'un ensemble de directives à utiliser pour tous ces projets. Ainsi, l'intérêt de ce guide ne réside pas seulement dans les règles elles-mêmes, mais également dans le partage de ces règles.

La norme PSR-2 a été acceptée en 2012 et depuis, un certain nombre de modifications ont été apportées à PHP, ce qui a des répercussions sur les directives de style de codage. Bien que la norme PSR-2 soit très complète sur les fonctionnalités PHP qui existaient au moment de la rédaction, les nouvelles fonctionnalités sont très ouvertes à l'interprétation.

Cette norme PSR vise donc à clarifier le contenu de la norme PSR-2 dans un contexte plus moderne avec de nouvelles fonctionnalités disponibles, et à rendre les errata de la norme PSR-2 contraignants.

Quelques principes clés

- Le code DOIT suivre toutes les règles décrites dans le PSR-1 .
- Le terme « StudlyCaps » dans PSR-1 DOIT être interprété comme PascalCase où la première lettre de chaque mot est en majuscule, y compris la toute première lettre.
- Tous les fichiers PHP DOIVENT utiliser uniquement la fin de ligne Unix LF (saut de ligne).
- Tous les fichiers PHP DOIVENT se terminer par une ligne non vide, terminée par un seul LF.
- La balise de fermeture `?>` DOIT être omise des fichiers contenant uniquement PHP.
- Les lignes NE DOIVENT PAS dépasser 80 caractères ; les lignes plus longues DOIVENT être divisées en plusieurs lignes ultérieures ne dépassant pas 80 caractères chacune.
- Il NE DOIT PAS y avoir d'espace à la fin des lignes.
- Des lignes vides PEUVENT être ajoutées pour améliorer la lisibilité et pour indiquer des blocs de code associés, sauf lorsque cela est explicitement interdit.
- Il NE DOIT PAS y avoir plus d'une déclaration par ligne.
- Le code DOIT utiliser un retrait de 4 espaces pour chaque niveau de retrait et NE DOIT PAS utiliser de tabulations pour le retrait.
- L'accolade ouvrante de la classe DOIT être placée sur sa propre ligne ; l'accolade fermante de la classe DOIT être placée sur la ligne suivante après le corps.

- Les accolades ouvrantes DOIVENT être sur leur propre ligne et NE DOIVENT PAS être précédées ou suivies d'une ligne vide.
- Les accolades fermantes DOIVENT être sur leur propre ligne et NE DOIVENT PAS être précédées d'une ligne vide.
- Tous les mots-clés et types PHP réservés DOIVENT être en minuscules.
- Tous les nouveaux types et mots-clés ajoutés aux futures versions de PHP DOIVENT être en minuscules.
- La forme courte des mots-clés de type DOIT être utilisée, par exemple bool au lieu de boolean, int au lieu de integer, etc.

```
<?php
for ($i = 0; $i < 10; $i++) {
    // for body
}

foreach ($iterable as $key => $value) {
    // foreach body
}

while ($expr) {
    // structure body
}

if ($expr1) {
    // if body
} elseif ($expr2) {
    // elseif body
} else {
    // else body;
}

if (
    $expr1
    && $expr2
) {
    // if body
} elseif (
    $expr3
    && $expr4
) {
    // elseif body
}
```

```

switch ($expr) {
    case 0:
        echo 'First case, with a break';
        break;
    case 1:
        echo 'Second case, which falls through';
        // no break
    case 2:
    case 3:
    case 4:
        echo 'Third case, return instead of break';
        return;
    default:
        echo 'Default case';
        break;
}

try {
    // try body
} catch (FirstThrowableType $e) {
    // catch body
} catch (OtherThrowableType | AnotherThrowableType $e) {
    // catch body
} finally {
    // finally body
}

```

D'autres éléments relatifs à la construction des classes (en POO), à la définition des espaces de noms et à l'ensemble des structures de base du PHP (opérateurs, structures de contrôle, variables, ...), sont également précisés dans cette section.

Tous les aspects présents sur : <https://www.php-fig.org/psr/psr-12/>

PSR-4 : Autoloading Standard

Objectif

Ce PSR décrit une spécification pour le chargement automatique de classes à partir de chemins de fichiers.

Il est entièrement interopérable et peut être utilisé en complément de toute autre spécification de chargement automatique.

Ce PSR décrit également où placer les fichiers qui seront chargés automatiquement conformément à la spécification.

Quelques principes clés

- Structure des Namespaces :
 - Les namespaces doivent correspondre à la structure des répertoires.
- Chemin des Fichiers :
 - Le chemin du fichier doit correspondre au nom complet de la classe.
- Un nom de classe entièrement qualifié a la forme suivante :
 - `\<NamespaceName>(\<SubNamespaceNames>)*\<ClassName>`

Nom de classe entièrement qualifié	Préfixe de l'espace de noms	Répertoire de base	Chemin du fichier résultant
\Acme\Log\Writer\File_Writer	Acme\Log\Writer	./acme-log-writer/lib/	./acme-log-writer/lib/File_Writer.php
\Aura\Web\Réponse\Statut	Aura\Web	/chemin/vers/aura-web/src/	/chemin/vers/aura-web/src/Response/Status.php
\Symfony\Core\Request	Symfony\Noyau	./vendor/Symfony/Core/	./vendor/Symfony/Core/Request.php
\Zend\Acl	Zend	/usr/includes/Zend/	/usr/includes/Zend/Acl.php

Documentation et Commentaires

Les commentaires aident à comprendre le code et sont essentiels pour la documentation et la collaboration en équipe. Voici quelques recommandations :

Commenter les en-têtes en expliquant le but des fonctions, classes, et algorithmes complexes.

```
<?php
/**
 * Calcule la somme de deux nombres.
 *
 * @param int $a Premier nombre
 * @param int $b Deuxième nombre
 * @return int La somme des deux nombres
 */
function additionner($a, $b) {
    return $a + $b;
}
```

Vous pouvez documenter les Paramètres et les Retours en utilisant des annotations avec les directives `@param` et `@return`.

Des outils comme PHPDocumentor (<https://www.phpdoc.org/>) vous permettent alors de générer automatiquement des documentations complètes de vos projets en parcourant vos codes sources (sur le même principe que javadoc par exemple).

Gestion des Erreurs et Débogage

Affichage des erreurs

Lors de l'exécution d'un script en ligne de commande, il est essentiel de pouvoir voir les erreurs. Pour ce faire, PHP dispose de 2 moyens : La directive `error_reporting` qui fixe le niveau d'erreur. Ce paramètre est un entier défini dans la configuration générale de PHP (`php.ini`).

Vous pouvez ajuster cette configuration en modifiant la valeur de la constante, telles que décrites dans la section Constantes pré-définies (<https://www.php.net/manual/fr/errorfunc.constants.php>).

À noter qu'une modification du fichier de configuration globale de PHP affectera l'ensemble des scripts exécutés. Vous pouvez cependant utiliser les fonctions `error_reporting` et `display_errors` pour ajuster ce comportement uniquement durant l'exécution d'un script spécifique.

La valeur par défaut est `E_ALL`.

```
<?php
// Désactiver le rapport d'erreurs
error_reporting(0);

// Rapporte les erreurs d'exécution de script
error_reporting(E_ERROR | E_WARNING | E_PARSE);
// Rapporter les E_NOTICE peut vous aider à améliorer vos scripts
// (variables non initialisées, variables mal orthographiées..)
error_reporting(E_ERROR | E_WARNING | E_PARSE | E_NOTICE);
// Rapporte toutes les erreurs à part les E_NOTICE
// C'est la configuration par défaut de php.ini
error_reporting(E_ALL & ~E_NOTICE);
// Rapporte toutes les erreurs PHP
error_reporting(E_ALL);
// Rapporte toutes les erreurs PHP mais moins lisible
error_reporting(-1);

// Même chose que error_reporting(E_ALL);
ini_set('error_reporting', E_ALL);
```

Et une fois le niveau d'erreur correctement défini, assurez-vous que le rapport d'erreurs est activé en mettant la directive `display_errors` à 1 :

```
<?php
...
ini_set('display_errors', 1);
...
```

TD0 : Installation de PHP

Objectifs :

- Installer PHP sur votre environnement de travail.
- Configurer PHP pour une utilisation en ligne de commande.
- Configurer PHP comme module web avec Apache.

Pour commencer, vous installerez sur votre station de travail un environnement PHP fonctionnel en 8.2 et vous devrez pouvoir accéder à un prompt PHP depuis une invite de commande et avoir un serveur web avec module php présent sur <http://localhost/r301devweb> qui pointera vers un répertoire r301-devWeb présent sur votre machine

Vous êtes libre des choix de votre environnement pour y arriver dès lors où vous maîtriser votre configuration.

Dans le cas où vous ne savez pas quel choix faire pour votre installation sous windows, je vous invite à privilégier WAMP.

TD1 : Prise en Main du PHP en Ligne de Commande

Objectifs :

- Comprendre la syntaxe de base de PHP.
- Manipuler des variables et des structures de contrôle.
- Créer des scripts PHP simples en ligne de commande.

Archive des fichiers : <https://bit.ly/2024-iut-php-td1-kguerrier>

Exercice 1 : Un classique Hello World en PHP

Créez un script PHP qui affiche "Hello, World!" lorsqu'il est exécuté en ligne de commande.

Nom du fichier : 1-hello_world.php

Instructions :

Affichez la phrase avec les variables suivie d'un retour à la ligne.

Le script doit être exécutable en ligne de commande

Exercice 2 : Introduction de Variables

Créez un script PHP qui utilise des variables pour stocker votre prénom, votre nom, et votre âge. Le script doit afficher une phrase du type : "Je m'appelle [prénom] [nom] et j'ai [âge] ans."

Nom du fichier : 2a-concatenation.php

Instructions :

Déclarez des variables pour le prénom, le nom et l'âge.

Affichez la phrase avec les variables suivie d'un retour à la ligne.

Le script doit être exécutable en ligne de commande.

Vous multipliez l'affichage de cette phrase de manière à utiliser :

- des concaténation avec guillemets doubles,
- des concaténation avec guillemets simples,
- aucune concaténation,
- l'ajustement d'une variable à l'affichage (augmenter l'âge 1),

Créez un script PHP qui demande à l'utilisateur de saisir un nombre positif, puis de saisir un pourcentage de remise et affiche le résultat arrondi à 2 décimales.

Nom du fichier : 2b-manipulation.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir un nombre positif.

Validez la saisie

Effectuez le calcul et affichez le résultat.

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir un nombre entier et affiche sa valeur absolue.

Nom du fichier : 2c-absolu.php

Instructions :

- Affichez un prompt à l'utilisateur pour lui demander de saisir un nombre entier.
- Validez la saisie
- Définissez sa valeur absolue et affichez le résultat.
- Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir un nombre positif, et affiche un prix remisé et arrondi à 2 décimales, en appliquant 5% par tranche de 100 €, et en ayant une remise maximale de 40%.

Nom du fichier : 2d-remise_max.php

Instructions :

- Affichez un prompt à l'utilisateur pour lui demander de saisir un nombre positif.
- Validez la saisie
- Effectuez le calcul et affichez le résultat.
- Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir un mot, et affiche le nombre de caractères qui constitue ce mot (attention aux accents ^_^)

Nom du fichier : 2e-taille_mot_mb.php

Instructions :

- Affichez un prompt à l'utilisateur pour lui demander de saisir un mot.
- Validez la saisie
- Effectuez le calcul et affichez le résultat.
- Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une phrase, et affiche le nombre de mots qui constitue cette phrase, pour chaque mot, afficher le nombre de caractère qui le constitue, et enfin afficher le nombre total de caractères (même remarque pour les accents ^_^)

Nom du fichier : 2f-taille_phrase_mb.php

Instructions :

- Affichez un prompt à l'utilisateur pour lui demander de saisir une phrase.
- Validez la saisie
- Effectuez les calculs et affichez les résultats.
- Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une équation du second degré (forme $ax^2 + bx + c = 0$), et affiche la ou les solution(s) possible(s).

Nom du fichier : 2g-second_degre.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une équation du second degré sous la forme $ax^2 + bx + c = 0$.

Validez la saisie

Récupérez les différents opérandes pour identifier a b et c

Effectuez le calcul de delta et affichez-le.

En fonction de delta, afficher la ou les solutions si elles existent

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir un nombre entier positif et qui affiche un décompte à partir de ce nombre jusqu'à 0 à raison d'un affichage toutes les secondes.

Nom du fichier : 2h-timer.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir un nombre entier positif.

Validez la saisie

Effectuez les calculs du timer et affichez le nouveau résultat chaque seconde.

Le script doit être exécutable en ligne de commande.

Exercice 3 : Tableaux

Créez un script PHP qui demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors la liste des nombres saisis séparés par une virgule.

Nom du fichier : 3a-timplode.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Parmi ces nombres saisis, vous afficherez alors uniquement la liste des nombres premiers séparés par une virgule.

Nom du fichier : 3b-tpremiers.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Filtrez le tableau pour ne garder que les nombres premiers

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant dont les valeurs sont séparées par une virgule.

Nom du fichier : 3c-tcroissant.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant dont les valeurs sont séparées par une virgule suivi d'un retour à la ligne puis la somme totale des nombres saisis.

Nom du fichier : 3d-tsomme.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Calculez la somme des valeurs du tableau

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Créez un script PHP qui demande à l'utilisateur de saisir une série de mots et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant de longueur dont les valeurs sont séparées par une virgule suivi d'un retour à la ligne et du nombre de valeurs saisies.

Nom du fichier : 3e-tstrings.php

Instructions :

Affichez un prompt à l'utilisateur pour lui demander de saisir une série de mots et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Exercice 4 : Calculatrice Basique

Créez une calculatrice basique qui peut effectuer les opérations d'addition, soustraction, multiplication, et division. Le script doit demander à l'utilisateur de saisir deux nombres et l'opération à effectuer. Ensuite, il doit afficher le résultat de l'opération choisie. Assurez-vous de gérer les erreurs comme la division par zéro et les entrées invalides.

Nom du fichier : 4-calculatrice.php

Instructions :

Ouvrez un éditeur de texte et créez un nouveau fichier nommé calculatrice.php.

Demandez à l'utilisateur de saisir deux nombres et l'opération souhaitée.

Validez chaque saisie

Réalisez l'opération demandée

Affichez le résultat.

Le script doit être exécutable en ligne de commande.

Exercice 5 : Arguments

Créez un script PHP prenant un paramètre `o` dont la valeur peut-être 'add', 'subtract', 'multiplies', ou 'divide' qui demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors l'opération correspondante en appliquant l'opérateur sur l'ensemble des valeurs de la liste. La division ne devra avoir que 2 valeurs.

Nom du fichier : 5-tparam.php

Instructions :

- Récupérer la valeur de l'opérateur parmi 'add', 'subtract', 'multiplies', ou 'divide'

- Validez la

- Afficher un prompt demandant à l'utilisateur de saisir une série de nombres (attention au cas particulier de la division) et de terminer sa saisie par une chaîne vide.

- Récupérer chaque saisie

- Validez la

- Stockez la valeur dans un tableau

- Utilisez ce qui vous semble judicieux pour réaliser l'exercice

- Affichez l'opération sous forme de chaîne de caractères

- Affichez le résultat.

- Le script doit être exécutable en ligne de commande.

Complément Web : Le W3

Vous l'avez certainement déjà remarqué, il existe une quantité importante de manière d'écrire du HTML, chacune amenant des capacités et des contraintes différentes.

Dans le cadre de ces exercices, je vous propose de travailler avec un framework simple et efficace W3.CSS, développé par W3Schools → <https://www.w3schools.com/css/>

W3Schools a été créé en 1998, et bien que son nom soit dérivé du World Wide Web, il n'est pas affilié au W3C (World Wide Web Consortium). Il est cependant plus léger et généralement plus efficace que Bootstrap par exemple.

Voici un exemple de formulaire :

```
<form class="w3-container" action="index.php?action=add" method="POST">
  <div class="dcontent">
    <div class="w3-row-padding">
      <div class="w3-half">
        <label class="w3-text-blue" for="field1"><b>Champ 1</b></label>
        <input class="w3-input w3-border" type="text" id="field1"
name="field1" />
      </div>
      <div class="w3-half">
        <label class="w3-text-blue" for="field2"><b>Champ 2</b></label><br />
        <input type="text" id="field2" name="field2" />
      </div>
    </div>
    <br />
    <div class="w3-row-padding">
      <div class="w3-half">
        <input class="w3-btn w3-red" type="submit" name="cancel"
value="Annuler" />
      </div>
      <div class="w3-half">
        <input class="w3-btn w3-blue-grey" type="submit"
name="confirm_envoyer" value="Envoyer" />
      </div>
    </div>
  </div>
</form>
```

TD2 : Utilisation de PHP au travers un serveur Web

Réalisez ces exercices pour qu'ils soient utilisables au travers un serveur web, en adaptant les sorties avec du code HTML et en utilisant les demandes de saisie par le biais de formulaires.

Objectifs :

- Comprendre la syntaxe de base de PHP au travers un serveur Web.
- Transmettre, récupérer et manipuler des variables et utiliser des structures de contrôle.
- Organiser une structure de code web

Archive des fichiers : <https://bit.ly/2024-iut-php-td2-kguerrier>

Dans cette archive, vous trouverez une arborescence légère organisée pour faciliter votre développement durant ce TD.

Prenez le temps de l'étudier, de voir le contenu de chaque fichier et de comprendre leurs rôles et la manière dont il est possible de les utiliser.

Le contenu de cette archive doit être présent dans un répertoire TD2, et devra donc être accessible depuis un navigateur internet sur <http://localhost/r301devweb/TD2>

[Accueil](#) [Ex1](#) [Ex2](#)

TD2 - PHP au travers un serveur Web

Dans ces exercices, vous réaliserez des intégrations de PHP au travers un serveur Web.

Réalisez ces exercices pour qu'ils soient utilisables au travers un serveur web, en adaptant les sorties avec du code HTML et en utilisant les demandes de saisie par le biais de formulaires.

Objectifs :

- Comprendre la syntaxe de base de PHP au travers un serveur Web.
- Comprendre les imbrications et les exécutions d'un serveur PHP.
- Transmettre, récupérer et manipuler des variables et utiliser des structures de contrôle.
- Organiser une structure de code web



Powered by [w3.css](#)

L'accès aux différents exercices se fera par le menu du haut et sera accessible depuis chacun d'eux.

Préambule

En l'état, vous constaterez que le menu affiché sur la page d'accueil est incomplet et les liens ne sont pas tous fonctionnels.

Pour commencer, identifiez le fichier qui se charge de l'affichage de ce menu et complétez le afin que l'ensemble des exercices soient présents et accessibles via ce lien.

Exercice 1 : Un classique Hello World en PHP

Créez une page web PHP qui affiche "Hello, World!" via PHP à l'intérieur d'une page HTML.

Nom du fichier : 1-hello_world.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Intégrer un interpréteur PHP et affichez la phrase suivie d'un retour à la ligne.

Exercice 2 : Introduction de Variables

Créez une page web PHP qui utilise des variables pour stocker votre prénom, votre nom, et votre âge. Le script doit afficher une phrase du type : "Je m'appelle [prénom] [nom] et j'ai [âge] ans." à l'intérieur d'une page HTML.

Nom du fichier : 2a-concatenation.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Déclarez des variables pour le prénom, le nom et l'âge.

Intégrer un interpréteur PHP et affichez la phrase avec les variables suivie d'un retour à la ligne.

Vous multipliez l'affichage de cette phrase de manière à utiliser :

- des concaténation avec guillemets doubles,
- des concaténation avec guillemets simples,
- aucune concaténation,
- l'ajustement d'une variable à l'affichage (augmenter l'âge 1),

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir un nombre positif, puis de saisir un pourcentage de remise et affiche le résultat arrondi à 2 décimales.

Nom du fichier : 2b-manipulation.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir un nombre positif.

Validez la saisie

Effectuez le calcul et affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir un nombre entier et affiche sa valeur absolue.

Nom du fichier : 2c-absolu.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir un nombre entier.

Validez la saisie

Définissez sa valeur absolue et affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir un nombre positif, et affiche un prix remisé et arrondi à 2 décimales, en appliquant 5% par tranche de 100 €, et en ayant une remise maximale de 40%.

Nom du fichier : 2d-remise_max.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir un nombre positif.

Validez la saisie

Effectuez le calcul et affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé..

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir un mot, et affiche le nombre de caractères qui constitue ce mot (attention aux accents ^_^)

Nom du fichier : 2e-taille_mot_mb.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir un mot.

Validez la saisie

Effectuez le calcul et affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une phrase, et affiche le nombre de mots qui constitue cette phrase, pour chaque mot, affiche le nombre de caractère qui le constitue, et enfin affiche le nombre total de caractères (même remarque pour les accents ^_^)

Nom du fichier : 2f-taille_phrase_mb.php

Instructions :

- Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)
 - Affichez un formulaire à l'utilisateur pour lui demander de saisir une phrase.
 - Validez la saisie
 - Effectuez les calculs et affichez les résultats.
 - Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.
-

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une équation du second degré (forme $ax^2 + bx + c = 0$), et affiche la ou les solution(s) possible(s).

Nom du fichier : 2g-second_degre.php

Instructions :

- Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)
 - Affichez un formulaire à l'utilisateur pour lui demander de saisir une équation du second degré sous la forme $ax^2 + bx + c = 0$.
 - Validez la saisie
 - Récupérez les différents opérandes pour identifier a b et c
 - Effectuez le calcul de delta et affichez-le.
 - En fonction de delta, afficher la ou les solutions si elles existent
 - Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.
-

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir un nombre entier positif et qui affiche un décompte à partir de ce nombre jusqu'à 0 à raison d'un affichage toutes les secondes.

Nom du fichier : 2h-timer.php

Instructions :

- Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)
- Affichez un formulaire à l'utilisateur pour lui demander de saisir un nombre entier positif.
- Validez la saisie
- Effectuez les calculs du timer et affichez le nouveau résultat chaque seconde.
- Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Exercice 3 : Tableaux

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors la liste des nombres saisis séparés par une virgule.

Nom du fichier : 3a-timplode.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Parmi ces nombres saisis, vous afficherez alors uniquement la liste des nombres premiers séparés par une virgule.

Nom du fichier : 3b-tpremiers.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Filtrez le tableau pour ne garder que les nombres premiers

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant dont les valeurs sont séparées par une virgule.

Nom du fichier : 3c-tcroissant.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une série de nombres et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant dont les valeurs sont séparées par une virgule suivi d'un retour à la ligne puis la somme totale des nombres saisis.

Nom du fichier : 3d-tsomme.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir une série de nombres et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Calculez la somme des valeurs du tableau

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Créez une page web PHP qui, via un formulaire, demande à l'utilisateur de saisir une série de mots et de terminer sa saisie par une chaîne vide. Vous afficherez alors cette liste triée par ordre croissant de longueur dont les valeurs sont séparées par une virgule suivi d'un retour à la ligne et du nombre de valeurs saisies.

Nom du fichier : 3e-tstrings.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur pour lui demander de saisir une série de mots et de terminer sa saisie par une chaîne vide.

Validez chaque saisie

Stockez la valeur dans un tableau

Identifiez si c'est l'utilisateur continue la saisie

Triez le tableau

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Exercice 4 : Calculatrice Basique

Créez une page web PHP qui, via un formulaire, propose une calculatrice basique permettant d'effectuer les opérations d'addition, soustraction, multiplication, et division. Le script doit demander à l'utilisateur de saisir deux nombres et l'opération à effectuer. Ensuite, il doit afficher le résultat de l'opération choisie. Assurez-vous de gérer les erreurs comme la division par zéro et les entrées invalides.

Nom du fichier : 4-calculatrice.php

Instructions :

Faites en sorte que le contenu de la structure de la page HTML soit reprise dans le fichier (menus, contenus, pied de page)

Affichez un formulaire à l'utilisateur de saisir deux nombres et l'opération souhaitée.

Validez chaque saisie

Réalisez l'opération demandée

Affichez le résultat.

Les valeurs saisies doivent rester dans les champs du formulaire une fois envoyé.

Exercice 5 : Arguments

Créez une page web PHP sur la base de l'exercice 4 et ajoutez la possibilité de passer l'opérateur directement dans l'URL afin de préprogrammer une opération. Le paramètre 'action' dont la valeur peut-être 'add', 'sub', 'mul', ou 'div' sera passé en paramètre. Les opérandes ne devront pas pouvoir être passés de cette manière.

Nom du fichier : 5-tparam.php

TD3 : Application Web avec Persistance en POO

Maintenant que vous maîtrisez les principes de fonctionnement des formulaires, il est temps de passer à la réalisation d'une application web permettant de gérer des entités en objet, et de les stocker dans une base de données pour pouvoir les conserver, les rappeler, les mettre à jour, ...

Objectifs :

- Comprendre l'arborescence d'un projet avec une base existante,
- Appréhender la manipulation des sessions, concernant notamment :
 - la gestion de l'authentification utilisateur
 - la gestion d'un moteur de messagerie utilisateur
- Transmettre, récupérer et manipuler des variables au travers des formulaires et des URLs,
- Réaliser des classes pour les entités à manipuler,
- Implémenter la gestion de la persistance en utilisant PDO pour MySQL,

Archive des fichiers : <https://bit.ly/2024-iut-php-td3-kguerrier>

Dans cette archive, vous trouverez une arborescence plus complète que celle du TD2 et qui a été organisée pour faciliter votre développement durant ce TD.

Prenez le temps de l'étudier, de voir le contenu de chaque fichier et de comprendre leurs rôles et la manière dont il est possible de les utiliser.

Le contenu de cette archive doit être présent dans un répertoire TD3, et devra donc être accessible depuis un navigateur internet sur <http://localhost/r301devweb/TD3>

Bienvenue sur le portail de votre projet !

Username

Identifiant

Password

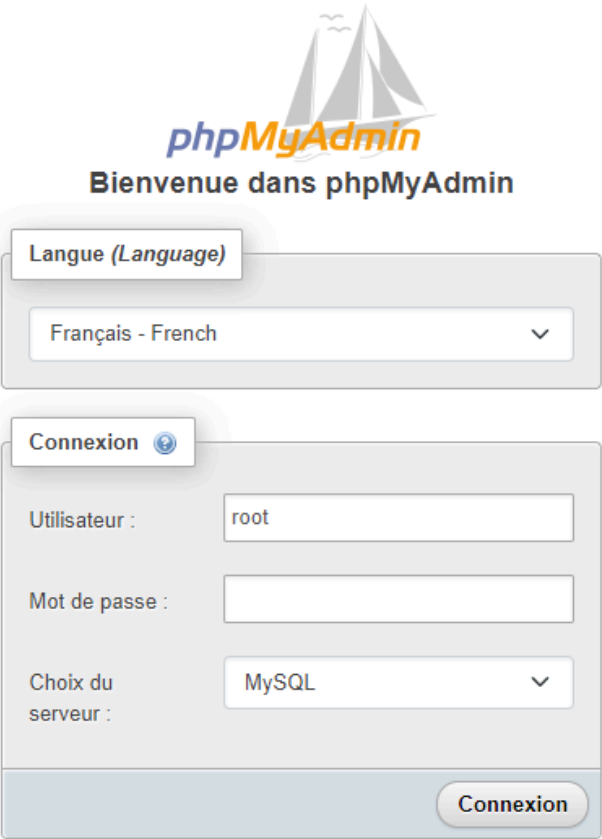
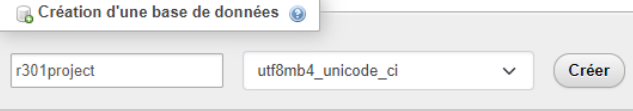
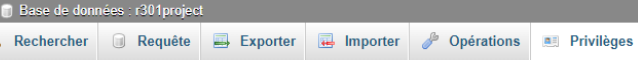
Mot de passe

Se connecter



Powered by [w3.css](#)

Mais avant de pouvoir vous connecter, il vous sera nécessaire de préparer votre base de données via phpmyadmin par exemple.

<p>Toujours via votre navigateur, vous accéderez à http://localhost/phpmyadmin qui devrait vous afficher une interface d'authentification.</p> <p>Pour vous connecter, vous utiliserez l'identifiant et mot que vous aurez renseigné lors de l'installation de votre serveur MySQL si vous l'avez personnalisé.</p> <p>Par défaut, les informations de connexion sont l'utilisateur "root" sans mot de passe.</p> <p>Je vous invite bien évidemment à changer cela et à mettre un mot de passe pour sécuriser un minimum votre serveur MySQL.</p>	
<p>Une fois connecté, vous devrez créer une base de données :</p> <p>Nom : r301project,</p> <p>Interclassement : utf8mb4_unicode_ci</p> <p>Cet interclassement permet la prise en charge de tous les caractères encodés jusqu'à 4 octets</p>	
<p>Vous créerez ensuite un user dédié à ce projet et qui n'aura accès qu'à cette table. Pour cela, vous pouvez utiliser l'onglet Privilèges <u>après avoir sélectionné votre base de données.</u></p>	

Vous renseignez les informations de votre utilisateur et lui donnez tous les privilèges uniquement sur la base de données r301project, (**aucun privilège global !**).

Bien évidemment, conservez précieusement ces informations de connexion, elles vous seront utiles pour votre TD ;-)

Ajouter un compte d'utilisateur

Informations pour la connexion

Nom d'utilisateur :
Saisir une valeur
r301_muser

Nom d'hôte :
Tout hôte
%

Mot de passe :
Saisir une valeur
Force : Fort

Saisir à nouveau :

Extension d'authentification
Mise en cache de l'authentification sha2

Générer un mot de passe:
Générer
Ga4ZGkpJPdp)(HyN

Base de données pour ce compte d'utilisateur

☐ Créer une base portant son nom et donner à cet utilisateur tous les privilèges sur cette base.
☐ Accorder tous les privilèges à un nom passe-partout (utilisateur_%).
☒ Donner tous les privilèges sur la base de données r301project.

Privilèges globaux
☐ Tout cocher

Ensuite, vous ajouterez dans votre base de données une table mp_users dont le moteur est InnoDB (pour les clés étrangères) et comprenant la structure suivante :

- rowid
 - clé primaire en int auto increment
- username
 - varchar 64
- password
 - varchar 255
- firstname
 - varchar 64
- lastname
 - varchar 64
- date_created
 - datetime à CURRENT_TIMESTAMP par défaut
- date_updated
 - datetime à CURRENT_TIMESTAMP par défaut
 - et dont la valeur revient à CURRENT_TIMESTAMP à la mise à jour
- admin
 - tiny int à 0 par défaut

Vous ajouterez un index unique sur username

#	Nom	Type
<input type="checkbox"/> 1	rowid	int
<input type="checkbox"/> 2	username	varchar(64)
<input type="checkbox"/> 3	password	varchar(255)
<input type="checkbox"/> 4	firstname	varchar(64)
<input type="checkbox"/> 5	lastname	varchar(64)
<input type="checkbox"/> 6	date_created	datetime
<input type="checkbox"/> 7	date_updated	datetime
<input type="checkbox"/> 8	admin	tinyint(1)

Index

Action	Nom de l'index	Type	Unique	Compressé	Colonne
Éditer Renommer Supprimer	PRIMARY	BTREE	Oui	Non	rowid
Éditer Renommer Supprimer	login	BTREE	Oui	Non	username

Vous créerez enfin un nouvel enregistrement dans cette table avec un username et un password crypté en md5, et dont le champs admin est à 1. C'est cet utilisateur que vous utiliserez pour vous connecter.

Avant de pouvoir vous utiliser le portail d'authentification du projet, il est nécessaire de placer les informations de connexion avec votre base de données dans le fichier .env :

<pre>DB_HOST='localhost' DB_NAME='aRemplacer' DB_PORT='' DB_USER='aRemplacer' DB_PASS='aRemplacer'</pre>	<p>Ce projet a été initialisé en incluant le composant DotENV, qui permet de générer des variables d'environnement à partir d'un fichier local de configuration nommé .env</p> <p>Remplacez le contenu de ce fichier par les informations relatives à l'utilisateur SQL que vous avez créé</p>
--	--

A partir de maintenant, votre projet PHP devrait pouvoir interroger votre base de données et vous devriez donc pouvoir vous connecter en utilisant les informations de l'utilisateur que vous avez inséré dans la table mp_users.

Une fois connecté, vous devriez arriver sur cette interface d'accueil :

Bienvenue admin

Les étudiants

Les enseignants

Les cours

Bienvenue sur votre page d'accueil de projet

Dans un premier temps, vous devrez réaliser un portail pour gérer :

•

•

•

Les étudiants

Les enseignants

Les cours

f

o

s

i

a

m

s

n

a

p

p

i

n

t

w

i

t

i

n

Powered by [w3.css](#)

Exercice 1 : Gestion des étudiants

- a. Créer une table `mp_etudiants` qui permet de stocker les données des étudiants, à savoir :
 - `rowid` comme clé primaire en int autoincrément
 - `numetu` comme `varchar(16)` destiné à stocker le numéro étudiant
 - `firstname` et `lastname`
 - `birthday`
 - `diploma` comme `varchar(16)` (par exemple BUT, Licence Pro, Master, ...)
 - `year` qui contient le numéro de l'année en cours
 - `td` `varchar(2)`
 - `tp` `varchar(2)`
 - `adress` en `text`, `zipcode` en `varchar(8)`, et `town` en `varchar(32)`
 - `fk_user` qui permet de pointer vers l'utilisateur associé
- b. Créer une classe permettant de manipuler ces enregistrements, et avec les spécificités suivantes :
 - `diploma` en tant que tableau pour faciliter les saisies et l'ajout de nouvelles valeurs
 - une fonction `create` qui permet de sauvegarder un nouvel étudiant en base de données,
 - une fonction `fetch` qui permet de recharger un étudiant en particulier à partir de `rowid`, de `numetu`
 - une fonction `fetchAll` qui permet de recharger tous les étudiants
 - une fonction `find` qui permet de récupérer une liste d'étudiants répondants à des critères particuliers
 - une fonction `update` qui permet de mettre à jour un étudiant existant
 - une fonction `delete` qui permet de supprimer un étudiant de la base de données (n'oubliez pas l'utilisateur ;-))

et bien évidemment toutes les fonctions que vous trouverez judicieuses pour votre utilisation ^_^.

Lors de la création de l'étudiant, le mot de passe utilisateur sera demandé en même temps que les autres informations et l'utilisateur sera donc créé en même temps que l'étudiant. En cas de problème lors de la création de l'utilisateur, la création de l'étudiant devra donc être annulée également.

- c. Identifiez le(s) fichier(s) utilisé(s) pour créer un nouvel étudiant, implémentez le formulaire, son traitement, et l'enregistrement des données en base en utilisant la classe.
- d. Identifiez le(s) fichier(s) utilisé(s) pour la liste des étudiants et complétez-les pour les rendre fonctionnels.
La liste devra également comprendre des filtres de recherche.
Pour chaque enregistrement, vous ajouterez également des icônes de modification et de suppression.

Exercice 2 : Gestion des enseignants

- a. Créer une table mp_enseignants qui permet de stocker les données des enseignants, à savoir :
 - rowid comme clé primaire en int autoincrément
 - firstname et lastname
 - birthday
 - adress en text, zipcode en varchar(8), et town en varchar(32)
 - fk_user qui permet de pointer vers l'utilisateur associé
- b. Créer une classe permettant de manipuler ces enregistrements, et avec les spécificités suivantes :
 - une fonction create qui permet de sauvegarder un nouvel enseignant en base de données,
 - une fonction fetch qui permet de recharger un enseignant en particulier à partir de rowid, de numens
 - une fonction fetchAll qui permet de recharger tous les enseignants
 - une fonction find qui permet de récupérer une liste d'enseignants répondants à des critères particuliers
 - une fonction update qui permet de mettre à jour un enseignant existant
 - une fonction delete qui permet de supprimer un enseignant de la base de données (n'oubliez pas l'utilisateur ;-))

et bien évidemment toutes les fonctions que vous trouverez judicieuses pour votre utilisation ^_^.

Lors de la création de l'enseignant, le mot de passe utilisateur sera demandé en même temps que les autres informations et l'utilisateur sera donc créé en même temps que l'enseignant. En cas de problème lors de la création de l'utilisateur, la création de l'enseignant devra donc être annulée également.

- c. Identifiez le(s) fichier(s) utilisé(s) pour créer un nouvel enseignant, implémentez le formulaire, son traitement, et l'enregistrement des données en base en utilisant la classe.
- d. Identifiez le(s) fichier(s) utilisé(s) pour la liste des enseignants et complétez-les pour les rendre fonctionnels. Pour chaque enregistrement, vous ajouterez également des icônes de modification et de suppression.

Exercice 3 : Gestion des cours

Pour cette partie, vous aurez plus de libertés dans la manière de la réaliser. Voici quelques explications :

- Un Cours correspond à un moment passé à une date donnée entre un groupe d'Etudiant (Promo / TD / TP), un Enseignant et concernant une Matière particulière.
 - Une Matière est définie par un numéro de matière, un nom, un coefficient et fait partie d'un Module.
 - Un Module est donc défini par un numéro de module, un nom, un coefficient et peut contenir plusieurs Matières.
- a. Commencez par réaliser les éléments permettant de gérer les Modules (table, classe, liste, ajout, modification, ...), et adaptez le menu pour accéder aux différentes parties.
 - b. Continuez en réalisant les éléments permettant de gérer les Matières (table, classe, liste, ajout, modification, ...), et adaptez le menu pour accéder aux différentes parties.
 - c. Poursuivez en réalisant les éléments permettant de gérer les Cours (table, classe, liste, ajout, modification, ...), et adaptez le menu pour accéder aux différentes parties.

Complément concernant l'utilisation d'un outil de génération de documentation : PhpDocumentor

Le projet PhpDocumentor est un outil permettant qui parse l'ensemble de votre code source et qui génère automatiquement l'ensemble de la documentation en fonction des commentaires présents dans les fichiers. Vous pouvez retrouver l'ensemble de la documentation officielle de ce projet sur <https://phpdoc.org/>

Cet outil se base sur les blocs de commentaires présents dans votre code source, et pour vous aider dans leur génération, voici quelques extensions VSCode qui pourraient vous être utiles (il en existe d'autre, à vous de trouver celles qui vous conviennent ^_^)

Insert Fileheader Comments Block

La première permet de générer et maintenir à jour le bloc présent en haut de chaque fichier, et qui contient les informations générales relative à ce fichier.

Par exemple, l'appel au raccourci par défaut "Ctrl + Alt + i" vous générera un bloc:

```
/**
 * Contrôler d'ajout pour l'élément de salle
 *
 * Il est chargé lorsque pour l'action d'ajout sur l'élément Salle
 *
 * @author: Kevin GUERRIER <guerrier.k@gmail.com>
 * @since: 2024-10-15 21:38:26
 * @Last Modified by: Kevin GUERRIER
 * @Last Modified time: 2024-10-16 12:00:16
 */
```

Les informations reprises automatiquement peuvent être personnalisées dans les paramètres de l'application.

PHPDoc Generator

Cette seconde extension permet de générer un bloc de commentaire spécifique à une fonction, une classe ou une variable, et qui contient les informations qui lui sont propres.

Par exemple, l'appel au raccourci par défaut "Ctrl + Enter" lorsque vous êtes sur la fonction `__set($property, $value)` d'une classe vous générera un bloc:

<pre>/** * @param mixed \$property * @param mixed \$value * * @return [type] */</pre>	<p>que vous pourrez compléter par la description en première ligne</p> <pre>/** * Méthode magique pour définir la valeur d'une propriété * qui est inexistante ou inaccessible. * * @param string \$property Nom de la propriété. * @param null \$value Valeur à assigner à la propriété */</pre>
--	---

Les informations reprises automatiquement peuvent être personnalisées dans les paramètres de l'application.

Déjà, avec ces 2 extensions, vous devriez pouvoir automatiser une grande partie de la génération de vos blocs de commentaire, et n'hésitez pas à aller en parcourir d'autres si besoin.

Utilisation de phpDocumentor

NOTE IMPORTANTE issue des auteurs du projet :

Il existe un package phpDocumentor Composer que vous pouvez utiliser pour installer phpDocumentor.

Cependant : phpDocumentor est une application complexe et ses bibliothèques sont utilisées dans d'innombrables autres bibliothèques et applications (2 de nos bibliothèques ont plus de 150 millions de téléchargements chacune) ; et cela signifie que les risques de conflit entre l'une de nos dépendances et la vôtre sont élevés. Et quand je dis élevé, c'est **vraiment élevé**.

Ainsi, en raison de ce qui précède : nous n'approuvons ni ne soutenons activement l'installation de phpDocumentor à l'aide de Composer

Donc pour revenir sur l'utilisation de l'outil phpDocumentor, vous devez utiliser le script <https://phpdoc.org/phpDocumentor.phar> . Conservez-le en dehors de vos codes sources pour pouvoir l'utiliser de la même manière pour l'ensemble de vos projets.

Ensuite, créez le fichier de configuration phpdoc.xml à la racine de votre projet à partir de la base suivante et ajustez le en fonction de vos besoins.

```

<?xml version="1.0" encoding="UTF-8" ?>
<phpdocumentor configVersion="3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="https://www.phpdoc.org"
  xsi:noNamespaceSchemaLocation="data/xsd/phpdoc.xsd">
  <title>Documentation du XXXXXX</title>
  <paths>
    <output>docs</output>
  </paths>
  <version number="latest">
    <folder>.</folder>
    <api>
      <source dsn=".">
        <path>.</path>
      </source>
      <output>api</output>
      <ignore hidden="true" symlinks="true">
        <path>docs/**/*</path>
        <path>tests/**/*</path>
        <path>vendor/**/*</path>
        <path>composer.*</path>
        <path>.env</path>
        <path>phpdoc.xml</path>
        <path>template_pheader.txt</path>
      </ignore>
      <extensions>
        <extension>php</extension>
      </extensions>
    </api>
  </version>
</phpdocumentor>

```

Vous retrouverez toutes les informations de ces éléments sur la référence de la documentation du site officiel <https://docs.phpdoc.org/guide/references/configuration.html>

Les éléments particuliers concernent notamment le contenu de la balise title, le contenu de la balise output, et le contenu de la balise ignore.

Une fois prêt, vous pouvez lancer la génération de la documentation en exécutant la commande :

```
php C:\phpDocumentor\phpDocumentor.phar run -v -c ".\phpdoc.xml"
```

et vous devriez donc obtenir l'ensemble de la documentation qui sera générée dans le répertoire "docs" (ou celui que vous avez précisé dans la balise output).

L'accès à cette documentation depuis un navigateur internet devrait vous fournir un affichage proche de celui-ci :

Documentation du TD3 R301 - Dev Web

 Search (Press "/" to focus)

Packages

[Application](#)

Reports

[Deprecated](#)

[Errors](#)

[Markers](#)

Indices

[Files](#)

Documentation

Table of Contents

Packages

 [Application](#)

Classes

 [Enseignant](#)

Classe permettant de gérer l'entité Enseignant

 [Etudiant](#)

Classe permettant de gérer l'entité Etudiant

 [Matiere](#)

Classe permettant de gérer l'entité Matière

 [myAuthClass](#)

Classe permettant de gérer l'authentification

 [Salle](#)

Classe permettant de gérer l'entité Salle