UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Licences Info
Info0303
2019/2020

Devoir Surveillé Terminal (1h30)

Aucun document, outil de calcul ou de communication n'est autorisé.

Exercice 1 (Question de cours)

1°) Expliquer ce qu'est une session.

2°) On souhaite stocker à l'aide d'une session en nombre entier correspondant au nombre de pages parcourues par un utilisateur au cours de sa navigation. Montrer à l'aide d'un script php comment réaliser cette fonction. Commenter les points importants du exemple.

3°) Compléter le **diagramme 1** fourni en annexe en précisant les différentes étapes du dialogue entre le client et le serveur lors de la mise en place d'une session.

4°) A quoi correspond la technologie **AJAX**. Quelle est son intérêt. Donner les éléments indispensables à mettre en place dans une application web pour utiliser **AJAX**.

5°) Compléter le **diagramme 2** fourni en annexe décrivant les échanges clients serveur dans le cadre d'un échange AJAX.

Exercice 2 (Base de données)

Nous souhaitons écrire une classe permettant de gérer les connexions à la base de données d'un site. Les identifiants (l'URL du SGBD, le nom de la base de données, le login et le mot de passe utilisateur) sont spécifiés dans un fichier PHP `DBConfig.php`.

1°) Donner le contenu du fichier `DBConfig.php`.

2°) Rappeler les précautions à prendre concernant ce fichier.

3°) Écrire la classe `MyPDO` contenant uniquement la méthode statique `getInstance` qui permet de créer ou retourner l'instance de l'objet PDO permettant d'accéder à la base de données.

4°) Expliquer comment utiliser cette classe et donnez un exemple d'utilisation en affichant un enregistrement spécifique et le contenu complet d'une table quelconque.

Exercice 3 (Problème)

On souhaite gérer une base de pilotes caractérisés par un *nom*, une *qualification* et une *date de validité*. Ces informations sont stockées dans une base de données.

1°) Expliquer quelles sont la(les) table(s) nécessaire(s) pour cette application et donner les champs.

2°) Donner le contenu de la classe `Pilote`.

3°) Écrivez la classe `PiloteModel` correspondant à la classe CRUD (pour Create, Read, Update et Delete) de la classe `Pilote`. Utiliserer la classe `MyPDO` de l'exercice précédent.

4°) Le code HTML suivant affiche un formulaire permettant la création d'un nouveau pilote.

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Création Pilote</title>
</head>
<body>
<h1>Création d un nouveau pilote</h1>
<form method="post" action="creation.php">
    <label for="nom" >Nom : </label>
    <input type="text" id="nom" name="nom">
    <label for="qualif">Qualification pilote : </label>
    <input type="text" id="qualif" name="qualif">
    <label for="valid">Date de validité</label>
    <input type="text" id="valid" name="datevalidite">
    <button type="submit"></button>
</form>
</body>
</html>
```

a) Donner le code du script `creation.php` qui vérifie que les différents champs du formulaire sont correctement renseignés.

b) Compléter le script `creation.php` afin qu'il ajoute le nouveau pilote dans la base de données.

Annexes - Manuel PHP

Classe **PDO** (version simplifiée) :

```
public __construct(string $dsn [, string $username [, string $passwd
                  [, array $options ]]])
public int exec(string $statement)
public string lastInsertId([ string $name = NULL ])
public PDOStatement prepare(string $statement
                           [, array $driver_options = array() ])
public PDOStatement query(string $statement)
```

Représente une connexion entre PHP et un serveur de base de données (SGBD). À noter que le paramètre `$dsn` du constructeur correspond à la chaîne suivante (pour une connexion à un SGBD *mySQL*) :

```
mysql:host=localhost;dbname=mabase;charset=utf8
```

Classe **PDOStatement** (version simplifiée) :

```
public bool bindColumn(mixed $column, mixed &$param [, int $type
                      [, int $maxlen [, mixed $driverdata ]]])
public bool bindParam(mixed $parameter, mixed &$variable
                     [, int $data_type = PDO::PARAM_STR [, int $length
                     [, mixed $driver_options ]]])
public bool bindValue(mixed $parameter, mixed $value
                     [, int $data_type = PDO::PARAM_STR ])
public bool closeCursor(void)
public int columnCount(void)
public void debugDumpParams(void)
public bool execute([ array $input_parameters ])
public mixed fetch([ int $fetch_style
                   [, int $cursor_orientation = PDO::FETCH_ORI_NEXT
                   [, int $cursor_offset = 0 ]]])
public array fetchAll([ int $fetch_style [, mixed $fetch_argument
                     [, array $ctor_args = array() ]]])
public mixed fetchColumn([ int $column_number = 0 ])
public mixed fetchObject([ string $class_name = "stdClass" [, array $ctor_args ]])
public mixed getAttribute(int $attribute)
public array getColumnMeta(int $column)
public bool nextRowset(void)
public int rowCount(void)
public bool setAttribute(int $attribute, mixed $value)
public bool setFetchMode(int $mode)
```

Représente une requête préparée et, une fois exécutée, le jeu de résultats associé.

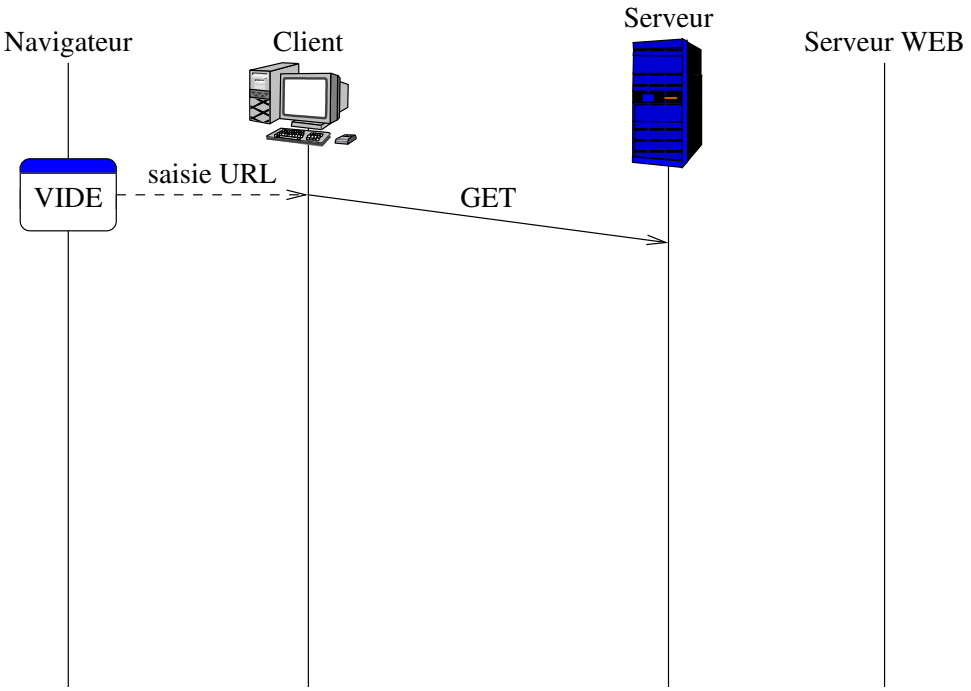
Fonction **unlink** (version simplifiée) - à utiliser avec prudence :

```
bool unlink(string $filename)
```

Efface le fichier dont le nom est `filename`. En cas d'échec, une alerte de niveau `E_WARNING` sera générée.

Document réponse

Question 2 : Session



Question 4 : Ajax