PHP et MySQL avec PDO

Vous avez obtenu la note de 0 sur 8

Le 07/03/2018 à 08:54 vous aviez obtenu la note de 4 sur 8

Le 07/03/2018 à 08:57 vous aviez obtenu la note de 8 sur 8

Quel style de fetch permet de récupérer les résultats d'une requête sous forme d'un tableau associatif ?

Plusieurs réponses sont possibles.

Explication:

Les méthodes fetch() et fetchAll() permettent de récupérer les résultats d'une requête SELECT préalablement exécutée (c.f. Fiche "Les classes PDO et PDOStatement").

PDO dispose de plusieurs façons de retourner les lignes, en voici une liste non exhaustive :

- PDO::FETCH_BOTH (Valeur par défaut) : Retourne un tableau indexé par les noms de colonnes et aussi par les numéros de colonnes, commençant à l'index 0
- PDO::FETCH_NUM : Retourne un tableau indexé par le numéro de la colonne, commençant à l'index 0
- PDO::FETCH_ASSOC : Retourne un tableau associatif indexé par le nom de la colonne

Exemple :

```
$query = $db->query('SELECT * FROM ma_table');
$results = $query->fetchAll(PDO::FETCH_BOTH); // Renvoie toutes les lignes de résultat sous forme d'un tableau associatif
$results = $query->fetchAll(PDO::FETCH_NUM); // Renvoie toutes les lignes de
```

`résultat sous forme d'un tableau indexé numériquement`

Consulter la liste complète des styles de fetch dans la documentation PHP :

http://php.net/manual/fr/pdostatement.fetch.php

PDO::FETCH_NUM

VPDO::FETCH_BOTH

PDO::FETCH_ASSOC

VPDO::FETCH_OBJ

Quel est le style de fetch par défaut de la classe PDO ?

Explication:

Les méthodes fetch() et fetchAll() permettent de récupérer les résultats d'une requête SELECT préalablement exécutée (c.f. Fiche "Les classes PDO et PDOStatement").

PDO dispose de plusieurs façons de retourner les lignes, en voici une liste non exhaustive :

- PDO::FETCH_BOTH (Valeur par défaut) : Retourne un tableau indexé par les noms de colonnes et aussi par les numéros de colonnes, commençant à l'index 0
- PDO::FETCH_NUM : Retourne un tableau indexé par le numéro de la colonne, commençant à l'index 0

- PDO::FETCH_ASSOC : Retourne un tableau associatif indexé par le nom de la colonne

PDO::FETCH_NUM

PDO::FETCH_ASSOC

VPDO::FETCH_BOTH

Préparer une requête avec la classe PDO permet de se prémunir des injections SQL, vrai ou faux ?

Explication:

Une injection SQL consiste à profiter d'une faille dans les paramètres transmis à une requête SQL, pour lui faire exécuter un code SQL malicieux.

Une protection plus efficace consiste à préparer les requêtes et à transmettre les valeurs des paramètres avec la méthode bindValue().

VVrai

Faux

La méthode query de la classe PDO permet de se prémunir des injections SQL, vrai ou faux ?

Explication:

La seule protection efficace contre les injections SQL consiste à préparée les requêtes et à leur transmettre les valeurs des paramètres avec la méthode bindValue().

Vrai

VFaux

Les failles XSS consistent à injecter du code Javascript dans une page, vrai ou faux ?

Explication:

Les attaques XSS (cross-site scripting) consistent à profiter d'une faille sur une page web, afin d'y injecter du code Javascript qui s'exécutera à chaque affichage de la page.

Le cas le plus courant survient lors de la transmission de données de formulaire, données qui seront insérées en base de données, puis affichées sur la page d'accueil d'un site.

VVrai

Faux

Quelle(s) fonction(s) PHP peut-on utiliser pour se prémunir des failles XSS ?

Plusieurs réponses sont possibles.

Explication:

Pour se prémunir des failles XSS, il existe des fonctions PHP qui permettent de supprimer/nettoyer le code HTML présent dans les chaînes de caractère.

Exemple : **form.php**

```
$pseudo = strip_tags($_POST['pseudo']); // Supprime toutes les balises HTML
$pseudo = htmlspecialchars($_POST['pseudo']); // Remplace tous les caractères HTML par leur équivalent en HTML entities
```

Consulter la documentation PHP pour plus d'informations sur les fonctions strip_tags et htmlspecialchars :

http://php.net/manual/fr/function.strip-tags.php

http://php.net/manual/fr/function.htmlspecialchars.php

Vstrip_tags

---

htmlentities

Vhtmlspecialchars

---

addslashes

Préparer une requête permet de se prémunir des injections SQL, vrai ou faux ?

Explication:

Préparer une requête permet de lui transmettre des paramètres, grâce aux méthodes bindValue() et bindParam() (c.f. "La classe PDOStatement").

Cela permet notamment d'éviter les injections SQL et d'améliorer les performances.

VVrai

Faux

Quelles sont les 2 manières de transmettre des paramètres à une requête via la méthode bindValue ?

Plusieurs réponses sont possibles.

Explication:

Préparer une requête permet de lui transmettre des paramètres, grâce aux méthodes bindValue() et bindParam() (c.f. "La classe PDOStatement").

Cela permet notamment d'éviter les injections SQL et d'améliorer les performances.

VAssocier une valeur à un nom correspondant

VAssocier une valeur à un point d'interrogation (comme paramètre fictif)

Utiliser la concaténation dans la requête SQL

Associer un paramètre à un nom de variable dont la valeur sera définie plus tard