eBook Gratuit

APPRENEZ pdo

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#pdo

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: pdo

It is an unofficial and free pdo ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official pdo.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec pdo

Remarques

Cette section fournit une vue d'ensemble de ce qu'est pdo et pourquoi un développeur peut vouloir l'utiliser.

Il devrait également mentionner tous les grands sujets dans pdo, et établir un lien avec les sujets connexes. La documentation de pdo étant nouvelle, vous devrez peut-être créer des versions initiales de ces rubriques connexes.

Examples

Installation ou configuration

PDO est une commande de connexion de base de données universelle en PHP, il prend en charge 12 types de bases de données différents, par exemple *MySQL, MongoDB, NoSQL* . Un gros avantage de PDO est qu'il calcule votre code pour prendre en charge le type de base de données, vous n'avez donc pas besoin de faire quelque chose lorsque vous passez à un autre système de base de données.

Résumé

	AOP	MySQLi
Support de base de données	12 pilotes différents	MySQLi
API	OOP	POO + procédural
Connexion	Facile	Facile
Paramètres nommés	Oui	Non
Cartographie d'objets	Oui	Oui
Déclarations préparées (côté client)	Oui	Non
Performance	Vite	Vite
Procédures stockées	Oui	Oui

Connexion à la base de données avec PDO:

La partie connexion semble maladroite mais nous devons nous en occuper. Le reste du PDO est simple et utile, il aide également à rendre la partie sécurisée encore plus facile.

```
$connection = new PDO("mysql:host=localhost;dbname=myDatabase, username, password);
```

Le PDO Connect est en ordre suivant:

PDO (`database type` **: host =** `host` **; dbname =** `database name` **,** `root` **,** `password` **);**

Sélection sécurisée de la base de données à l'aide de PDO:

```
// We use a array to hold the data about whats the :var is in normal $var
$params = array(
    ':username' => '$username',
    ':email' => $mail,
);

// Prepare the SQL and using named secure parameters ":username"
$pdo->prepare('SELECT * FROM users WHERE username = :username AND email = :email');

// Execute the $params and send them to the $pdo->prepare
$pdo->execute($params);
```

Le code que vous venez de lire est protégé par des agents SQL injection

---

Tutoriels:

Installer:

Comment installer PDO si vous ne l'avez pas

Guides:

Tutoriel W3Schools
Tuts + Tutorial (Recommandé)

Code d'erreur du paramètre PDO

PDO :: setAttribute définit un attribut sur le descripteur de base de données. La description de setAttribute est la suivante:

```
public bool PDO::setAttribute ( int $attribute , mixed $value )
```

PDO :: ATTR_ERRMODE: cet attribut est utilisé pour signaler les erreurs. Il peut avoir l'une des valeurs suivantes.

- `PDO::ERRMODE_SILENT` : Si ATTR_ERRMODE n'est pas défini dans le code, ERRMODE_SILENT est la valeur par défaut de l'attribut ATTR_ERRMODE. Il définit les codes d'erreur. En mode silencieux, en cas d'erreur dans SQL, PDO ne générera aucune exception. PDO n'émettra aucun avertissement; il retournera simplement faux. La valeur de PDO :: ERRMODE_SILENT est 0. Le script s'exécute sans générer d'erreur ou d'avertissement.
- `PDO::ERRMODE_WARNING` : Cette valeur augmente E_WARNING. En mode d'avertissement, s'il y a une erreur dans SQL, PDO émettra des avertissements mais le script continuera à fonctionner. La valeur de PDO :: ERRMODE_WARNING est 1. Le script s'exécute en générant un avertissement concernant l'erreur.
- `PDO::ERRMODE_EXCEPTION` : cette valeur génère des exceptions. En mode exception, s'il y a une erreur dans SQL, PDO lancera des exceptions et le script cessera de fonctionner. La valeur de PDO :: ERRMODE_EXCEPTION est 2. Le script arrête l'exécution en générant l'erreur qui génère l'exception.

Exemple: Nous allons maintenant voir les différentes valeurs de l'attribut ATTR_ERRMODE avec quelques exemples. Pour ce faire, créez une base de données appelée `learn_project_db` et insérez à l'intérieur une table appelée `user_table` . L'extrait de code SQL suivant peut être utilisé pour y parvenir:

```
DROP DATABASE IF EXISTS `learn_project_db`;
CREATE DATABASE `learn_project_db`;
USE `learn_project_db`;
CREATE TABLE `user_table` (
  `user_email` varchar(50) PRIMARY KEY,
  `user_password` varchar(50) NOT NULL
);
INSERT INTO `user_table` (`user_email`, `user_password`) VALUES
('test1@example.com', '123'),
('test2@example.com', '1234'),
('test3@example.com', '12345');
```

Au début, nous allons vérifier ce qui se passera si nous ne définissons pas `ATTR_ERRMODE` et nous aurons une erreur dans la requête SQL. Créez un fichier PHP appelé `default.php` et essayez ceci:

```
<?php
    $server = "localhost";
    $db_username = "root";
    $db_password = "";
    $db_name = "learn_project_db";
    $conn = new PDO("mysql:host=$server;dbname=$db_name",$db_username,$db_password);
    $sql_query = "SELECT * FROM wrong_user_table";
    $stmt = $conn->prepare($sql_query);
    $stmt->execute();
    $result_set = $stmt->fetchAll();
    var_dump($result_set);
    /*Get the current error mode of PDO*/
```

```
    $current_error_mode = $conn->getAttribute(PDO::ATTR_ERRMODE);
    echo "<br>";
    echo "Value of PDO::ATTR_ERRMODE: ".$current_error_mode;
?>
```

Notez que le nom de la table est `wrong_user_table` dans la requête qui n'est pas définie dans la base de données créée précédemment. Mais comme nous n'avons pas défini `ATTR_ERRMODE` , le script sera exécuté sans lancer d'exception ni émettre d'avertissement. Il affichera un tableau vide comme jeu de résultats. Nous devrions également remarquer que la valeur de `PDO::ATTR_ERRMODE` est 0.

Maintenant, nous allons vérifier ce qui se passera si nous définissons `ATTR_ERRMODE` avec la valeur `PDO::ERRMODE_WARNING` et que nous aurons une erreur dans la requête SQL. Créez un fichier PHP appelé `warning.php` et essayez ceci:

```
<?php
    $server = "localhost";
    $db_username = "root";
    $db_password = "";
    $db_name = "learn_project_db";
    $conn = new PDO("mysql:host=$server;dbname=$db_name",$db_username,$db_password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
    $sql_query = "SELECT * FROM wrong_user_table";
    $stmt = $conn->prepare($sql_query);
    $stmt->execute();
    $result_set = $stmt->fetchAll();
    var_dump($result_set);
    /*Get the current error mode of PDO*/
    $current_error_mode = $conn->getAttribute(PDO::ATTR_ERRMODE);
    echo "<br>";
    echo "Value of PDO::ATTR_ERRMODE: ".$current_error_mode;
?>
```

La sortie de `warning.php` est la suivante:

```
Warning: PDOStatement::execute(): SQLSTATE[42S02]: Base table or view not found: 1146 Table
'learn_project_db.wrong_user_table' doesn't exist in E:\xampp\htdocs\oop\db.php on line 10
array(0) { }
Value of PDO::ATTR_ERRMODE: 1
```

Cette fois-ci, nous `ATTR_ERRMODE` avec la valeur `PDO::ERRMODE_WARNING` pour afficher un message d'avertissement. Le script s'exécute avec succès et affiche un tableau vide en sortie avec la valeur de `PDO::ATTR_ERRMODE` de 1.

Enfin, nous allons vérifier ce qui se passera si nous définissons `ATTR_ERRMODE` avec la valeur `PDO::ERRMODE_EXCEPTION` et que nous aurons une erreur dans la requête SQL. Créez un fichier PHP appelé `error.php` et essayez ceci:

```
<?php
    $server = "localhost";
    $db_username = "root";
    $db_password = "";
    $db_name = "learn_project_db";
```

```
    $conn = new PDO("mysql:host=$server;dbname=$db_name",$db_username,$db_password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql_query = "SELECT * FROM wrong_user_table";
    $stmt = $conn->prepare($sql_query);
    $stmt->execute();
    $result_set = $stmt->fetchAll();
    var_dump($result_set);
    /*Get the current error mode of PDO*/
    $current_error_mode = $conn->getAttribute(PDO::ATTR_ERRMODE);
    echo "<br>";
    echo "Value of PDO::ATTR_ERRMODE: ".$current_error_mode;
?>
```

Le résultat de `error.php` est:

```
Fatal error: Uncaught exception 'PDOException' with message 'SQLSTATE[42S02]: Base table or
view not found: 1146 Table 'learn_project_db.wrong_user_table' doesn't exist' in
E:\xampp\htdocs\oop\db.php:10 Stack trace: #0 E:\xampp\htdocs\oop\db.php(10): PDOStatement-
>execute() #1 {main} thrown in E:\xampp\htdocs\oop\db.php on line 10
```

Cette fois, comme nous définissons la valeur `ATTR_ERRMODE` avec `PDO::ERRMODE_EXCEPTION` , il lancera une `PDOException` qui génère une erreur fatale. Le script cessera de s'exécuter une fois l'exception levée. C'est l'approche la plus courante pour gérer les erreurs liées aux requêtes de base de données. Dans la plupart des cas, nous définissons l'attribut `ATTR_ERRMODE` avec cette valeur pour gérer toute exception pouvant résider dans une requête SQL.

Lire Démarrer avec pdo en ligne: https://riptutorial.com/fr/pdo/topic/4393/demarrer-avec-pdo

Chapitre 2: Préparer et exécuter vos instructions SQL

Remarques

Attention

Une instruction préparée ne peut pas prendre en compte un paramètre sauvage pour les noms de table. Par exemple, cette déclaration suivante n'est pas correcte:

```
$query = "SELECT name, city FROM ? WHERE id = ? AND country = ?";
```

La requête préparée correcte serait:

```
$query = "SELECT name, city FROM users WHERE id = ? AND country = ?";
```

Examples

Usage

```
// 1. Connect to the database (this example with MySQL)
$host = 'localhost';
$database = 'users';
$user = 'root';
$password = '';
$dsn = "mysql:host=$host;dbname=$database";
$pdo = new PDO($dsn, $user, $password);

// 2. Prepare your query

// 2.1 First way
$query_1 = "SELECT name, city FROM users WHERE id = ? AND country = ?";
$statement_1 = $pdo->prepare($query_1);

// 2.2 Second way
$query_2 = "SELECT name, city FROM users WHERE id = :id AND country = :country";
$statement_2 = $pdo->prepare($query_2);

// 3. Execute your query

// 3.1 With the first way
$statement_1->execute([1, 'US']);

// 3.2 With the second way
$statement_2->execute([
    ':id' => 1,
    ':country' => 'US'
]);

// 4. Fetch your data
```

```
$data_1 = $statement_1->fetchAll();
$data_2 = $statement_2->fetchAll();
```

Lire Préparer et exécuter vos instructions SQL en ligne:
https://riptutorial.com/fr/pdo/topic/6511/preparer-et-executer-vos-instructions-sql

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec pdo	arsho, Community, hjpotter92, Ilyas Mimouni, manian, TheCrazyProfessor, Your Common Sense
2	Préparer et exécuter vos instructions SQL	Anwar Nairi, Your Common Sense