EBook Gratis

APRENDIZAJE pdo

Free unaffiliated eBook created from
Stack Overflow contributors.

#pdo

Tabla de contenido

Acerca de

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

Capítulo 1: Empezando con pdo

Observaciones

Esta sección proporciona una descripción general de qué es pdo y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro de pdo, y vincular a los temas relacionados. Dado que la Documentación para pdo es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Examples

Instalación o configuración

PDO es un comando de conexión de base de datos universal en PHP que admite 12 tipos de bases de datos diferentes, por ejemplo , *MySQL, MongoDB, NoSQL* . Una gran ventaja de la DOP es que calcula su código para admitir el tipo de base de datos, por lo que no necesita tener ninguna posibilidad al pasar a otro sistema de base de datos.

Resumen

	DOP	MySQLi
Soporte de base de datos	12 conductores diferentes	MySQLi
API	OOP	OOP + procesal
Conexión	Fácil	Fácil
Parámetros nombrados	Sí	No
Mapeo de objetos	Sí	Sí
Declaraciones preparadas (del lado del cliente)	Sí	No
Actuación	Rápido	Rápido
Procedimientos almacenados	Sí	Sí

Conexión a la base de datos con DOP:

La parte de la conexión parece incómoda, pero debemos lidiar con eso. El resto de la DOP es simple y útil, también ayuda a hacer que la parte segura sea aún más fácil.

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

La conexión PDO es orden siguiente:

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

Selección segura de la base de datos mediante DOP:

```
// 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);
```

El código que acaba de leer, es agentes inyectados de inyección SQL.

Tutoriales:

Instalar:

Cómo instalar PDO si no lo tienes

Guías:

Tutorial de W3Schools
Tuts + Tutorial (Recomendado)

Configuración de PDO Atributo Errormode

PDO :: setAttribute establece un atributo en el identificador de la base de datos. La descripción de setAttribute es:

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

PDO :: ATTR_ERRMODE: este atributo se utiliza para informar errores. Puede tener uno de los siguientes valores.

- `PDO::ERRMODE_SILENT` : Si ATTR_ERRMODE no está configurado en el código, ERRMODE_SILENT es el valor predeterminado del atributo ATTR_ERRMODE. Establece códigos de error. En modo silencioso, si hay un error en SQL, PDO no generará excepciones; DOP no emitirá advertencias; simplemente devolverá falso. El valor de PDO :: ERRMODE_SILENT es 0. El script se ejecutará sin generar ningún error o advertencia.
- `PDO::ERRMODE_WARNING` : Este valor aumenta E_WARNING. En el modo de advertencia, si hay un error en SQL, PDO emitirá advertencias pero la secuencia de comandos continuará ejecutándose. El valor de PDO :: ERRMODE_WARNING es 1. El script se ejecutará y generará una advertencia sobre el error.
- `PDO::ERRMODE_EXCEPTION` : este valor `PDO::ERRMODE_EXCEPTION` excepciones. En el modo de excepción, si hay un error en SQL, PDO lanzará excepciones y el script dejará de ejecutarse. El valor de PDO :: ERRMODE_EXCEPTION es 2. El script dejará de ejecutarse generando el error que produce la excepción.

Ejemplo: Ahora vamos a ver los diversos valores del atributo ATTR_ERRMODE con algunos ejemplos. Para hacerlo, cree una base de datos llamada `learn_project_db` e inserte una tabla llamada `user_table` dentro de ella. El siguiente fragmento de código SQL se puede utilizar para lograrlo:

```
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');
```

Al principio vamos a ver qué pasará si no configuramos `ATTR_ERRMODE` y tendremos un error en la consulta SQL. Crea un archivo PHP llamado `default.php` y prueba esto:

```
<?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;
?>
```

Observe que, el nombre de la tabla es `wrong_user_table` en la consulta que no está definida en la base de datos que creamos anteriormente. Pero, como no configuramos el `ATTR_ERRMODE` , ejecutará el script sin lanzar ninguna excepción ni emitir ningún aviso. Se generará una matriz vacía como conjunto de resultados. También debemos notar que, el valor de `PDO::ATTR_ERRMODE` es 0.

Ahora, vamos a comprobar qué ocurrirá si configuramos el `ATTR_ERRMODE` con el valor `PDO::ERRMODE_WARNING` y tendremos un error en la consulta SQL. Crea un archivo PHP llamado `warning.php` y prueba esto:

```
<?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 salida de `warning.php` es:

```
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
```

Esta vez, cuando configuramos el valor `ATTR_ERRMODE` con `PDO::ERRMODE_WARNING` , se mostrará un mensaje de advertencia. La secuencia de comandos se ejecuta correctamente y muestra una matriz vacía como salida con el valor de `PDO::ATTR_ERRMODE` de 1.

Por fin, vamos a ver qué ocurrirá si configuramos el `ATTR_ERRMODE` con el valor `PDO::ERRMODE_EXCEPTION` y tendremos un error en la consulta SQL. Crea un archivo PHP llamado `error.php` y prueba esto:

```
<?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;
?>
```

La salida de `error.php` es:

```
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
```

Esta vez, cuando configuramos el valor `ATTR_ERRMODE` con `PDO::ERRMODE_EXCEPTION` , lanzará la `PDOException` que genera un error fatal. El script dejará de ejecutarse después de que se lance la excepción. Este es el enfoque más común para manejar los errores relacionados con la consulta de la base de datos. En la mayoría de los `ATTR_ERRMODE` , establecemos el atributo `ATTR_ERRMODE` con este valor para manejar cualquier excepción que pueda residir en la consulta SQL.

Lea Empezando con pdo en línea: https://riptutorial.com/es/pdo/topic/4393/empezando-con-pdo

Capítulo 2: Prepara y ejecuta tus sentencias SQL

Observaciones

Advertencia

La declaración preparada no puede importar un parámetro salvaje para los nombres de tabla. Por ejemplo, esta declaración siguiente no es correcta:

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

La consulta preparada correcta sería:

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

Examples

Uso

```
// 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();
```

Lea Prepara y ejecuta tus sentencias SQL en línea:
https://riptutorial.com/es/pdo/topic/6511/prepara-y-ejecuta-tus-sentencias-sql

Creditos

S. No	Capítulos	Contributors
1	Empezando con pdo	arsho, Community, hjpotter92, Ilyas Mimouni, manian, TheCrazyProfessor, Your Common Sense
2	Prepara y ejecuta tus sentencias SQL	Anwar Nairi, Your Common Sense