Atitit php pdo的api使用

目录

[1.1. PHP PDO简介 1](#_Toc22783)

[1.2. 若要使用数据库长连接，：PDO::ATTR\_PERSISTENT 1](#_Toc29652)

[2. 其他设置 2](#_Toc27146)

[2.1. )、PDO有三种错误处理方式：PDO::ERRMODE\_EXCEPTION 2](#_Toc20922)

[2.2. 返回的字段名称大小写处理PDO::ATTR\_CASE 2](#_Toc8293)

[2.3. 指定数据库返回的NULL值在PHP中对应的数值DO::ATTR\_ORACLE\_NULLS 2](#_Toc13602)

[2.4. Tisye.php 2](#_Toc32484)

[2.5. tisye mng list.php 查询 3](#_Toc11040)

## PHP PDO简介

PDO (PHP Data Object) 是 PHP5 的新特性，在 PHP6 中，PDO 将是默认的数据库处理方式，所以我们有必要学习一下 PDO。

在 PHP5 以前的版本中，对于数据库的处理通常是通过一些扩展来支持的。如 windows 版本下的 php\_mysql.dll、php\_mssql.dll 这些扩展分别用于处理 mysql 和 MS SQL Server 等数据库。现在，POD 的出现，为我们提供了统一的数据库处理接口。

PDO 是面向对象方式处理数据库连接操作的，其预定义了以下 3 个类：

1. PDO：代表一个 PHP 和数据库之间的连接。
2. PDOStatement：代表一条预处理语句以及语句执行后的联合结果集（associated result set）。
3. PDOException：返回PDO触发的错误。

常用的数据库操作方法如下：

* PDO::query()：主要是用于有记录结果返回的操作，如 SELECT 操作。
* PDO::exec()：主要是针对没有结果集合返回的操作，比如 INSERT、UPDATE、DELETE 等操作，它返回的结果是当前操作影响的列数。
* PDO::prepare()：主要是用于预处理操作，需要通过 PDOStatement::execute() 来执行预处理里面的 SQL 语句，这个方法可以绑定参数，功能强大，相对比较复杂。

## 若要使用数据库长连接，：PDO::ATTR\_PERSISTENT

默认不是长连接，需要在最后加如下参数

$dbh = new PDO("mysql:host=localhost;dbname=db\_demo","root","password","array(PDO::ATTR\_PERSISTENT => true)");    
$dbh = null; //(释放)

# 其他设置

## )、PDO有三种错误处理方式：PDO::ERRMODE\_EXCEPTION

* PDO::ERRMODE\_SILENT不显示错误信息，只设置错误码
* PDO::ERRMODE\_WARNING显示警告错
* PDO::ERRMODE\_EXCEPTION抛出异常

可通过以下语句来设置错误处理方式为抛出异常

$db->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

当设置为PDO::ERRMODE\_SILENT时可以通过调用 errorCode() 或 errorInfo()来获得错误信息，当然其他情况下也可以。

## 返回的字段名称大小写处理****PDO::ATTR\_CASE****

1. 、因为不同数据库对返回的字段名称大小写处理不同，所以PDO提供了**PDO::ATTR\_CASE**设置项（包括PDO::CASE\_LOWER，PDO::CASE\_NATURAL，PDO::CASE\_UPPER），来确定返回的字段名称的大小写。

## 指定数据库返回的NULL值在PHP中对应的数值****DO::ATTR\_ORACLE\_NULLS****

3)、通过设置**PDO::ATTR\_ORACLE\_NULLS**类型（包括PDO::NULL\_NATURAL，PDO::NULL\_EmpTY\_STRING，PDO::NULL\_TO\_STRING）来指定数据库返回的NULL值在PHP中对应的数值。

# ****、PDO常用方法及其应用****

* PDO::query() 主要是用于有记录结果返回的操作，特别是SELECT操作
* PDO::exec() 主要是针对没有结果集合返回的操作，如INSERT、UPDATE等操作
* PDO::prepare() 主要是预处理操作，需要通过$rs->execute()来执行预处理里面的SQL语句，这个方法可以绑定参数，功能比较强大（防止sql注入就靠这个）
* PDO::lastInsertId() 返回上次插入操作，主键列类型是自增的最后的自增ID
* PDOStatement::fetch() 是用来获取一条记录
* PDOStatement::fetchAll() 是获取所有记录集到一个集合
* PDOStatement::fetchColumn() 是获取结果指定第一条记录的某个字段，缺省是第一个字段
* PDOStatement::rowCount() :主要是用于PDO::query()和PDO::prepare()进行DELETE、INSERT、UPDATE操作影响的结果集，对PDO::exec()方法和SELECT操作无效

## Tisye.php

<?php

error\_reporting(E\_ALL^E\_NOTICE);

parse\_str( $\_SERVER[ 'QUERY\_STRING' ],$parr);print\_r($parr);

  $json\_str=json\_encode($parr);

file\_put\_contents("C:\\data\\tisyi\\".time(), $json\_str);

 print ('ok');

// echo "Hello world!<br>";

$aa=333;

$mysql\_conf = array(

  );

$pdo = new PDO("pgsql:host=" . $mysql\_conf['host'] . ";port=5432;dbname=" . $mysql\_conf['db'], $mysql\_conf['db\_user'], $mysql\_conf['db\_pwd']);//创建一个pdo对象

$pdo->exec("set names 'utf8'");

$sql = "select \* from user where name = ?";

//sprintf("%s love %s.", "a","b");

// use exec() because no results are returned

$sql = "INSERT INTO tisye提现表 (org开户机构, acc账户, name名字,amt金额)VALUES ('%s', '%s', '%s','%s') ";

print\_r($\_GET);

$sql=sprintf($sql,$\_GET['org开户机构'],$\_GET['acc'],$\_GET['name'],$\_GET['amt金额']);

print\_r($sql);

 print\_r( $pdo);

//try{

  $pdo->exec($sql);

  echo "New record created successfully";

// }

// catch(PDOException $e)

//     {

//     echo $sql . "<br>" . $e->getMessage();

//     }

?>

## tisye mng list.php 查询

<?php

//tisye mng list.php

error\_reporting(E\_ALL^E\_NOTICE);

parse\_str( $\_SERVER[ 'QUERY\_STRING' ],$parr);

//print\_r($parr);

$mysql\_conf = array(

  );

$pdo = new PDO("pgsql:host=" . $mysql\_conf['host'] . ";port=5432;dbname=" . $mysql\_conf['db'], $mysql\_conf['db\_user'], $mysql\_conf['db\_pwd']);//创建一个pdo对象

$pdo->exec("set names 'utf8'");

$sql = "select \* from tisye提现表 where 1=1";

$sth = $pdo->query($sql);

$rows = $sth->fetchAll();

echo json\_encode($rows);

?>

如何使用PDO查询Mysql来避免SQL注入风险？ThinkPHP 3.1中的SQL注入漏洞分析！ - 52php - 博客园.html