PDO

PHP Data Object

PHP Hypertext Processor Data Object

PHP Hypertext Processor Hypertext Processor Data Object

PHP Hypertext processor Hypertext Processor Hypertext Processor Data Object

PHP Hypertext Processor Hypertext processor Hypertext Processor Hypertext Processor Data Object

PHP Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Data Object

PHP Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Data Object

PHP Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Hypertext Processor Data Object

PDO + SQL

PDO är numera det rätta sättet att säkert koppla till databaser via PHP

```
mysql_connect(); //the old ways
```

```
mysqli_connect(); //the old ways
```

Mer osäkra kopplingar, PDO gör mycket arbete åt oss

(PHP 4, PHP 5)

mysql_connect — Open a connection to a MySQL Server

Warning This extension was deprecated in PHP 5.5.0, and it was removed in PHP 7.0.0. Instead, the MySQLi or PDO_MySQL extension should be used. See also MySQL: choosing an API guide and related FAQ for more information. Alternatives to this function include:

- mysqli_connect()
- PDO::__construct()

`PDO` behöver: Database source, username, password, (options)

Options är optional, vi kan skippa det men kan behövas senare.

http://localhost:8888/MAMP/

Vi instansierar ett nytt PDO-object

```
$pdo = new PDO(
"mysql:host=localhost;dbname=products;charset=utf8",
"root",
"root"
);
```

Spara instansen!

Vanliga fel

Rätt adress= (localhost eller 127.0.0.1)

Testa ange portnummer

Rätt lösen och användarnamn?

Har du angett charset? (charset ska vara `'utf8'`)

Lägg till i början av filen

```
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
```

PDO-objektet är som ett vanligt objekt

```
$myObj->myFunction();
```

Vi har dock inbyggda funktioner som t.ex. `prepare()`

Prepare

Med `prepare` skriver vi våra SQL-statements

```
$statement = $pdo->prepare("SELECT * FROM pc");
```

När vi har förberett ett statement måste vi utföra det

```
$statement->execute();
```

`$statement` kommer nu att innehålla de hämtade raderna och vi kan loopa igenom dem.

SUPERGLOBALS

I PHP har vi såkallade **Superglobals**

Dessa används främst för att hämta information vid **GET** och **POST**

Varje request innehåller som vi tidigare vet mer data än enbart URLen

Testa att `var_dump` dessa i er index.php

```
$_POST
$_GET
$_SERVER
```

Med `$_POST` och `$_GET` kan vi hämta data som skickas vid varje request

Kom ihåg **HTTP: CRUD**