# 26. PHP – Propojení s DB

K databázi se připojuje, pokud je třeba pracovat s větším množstvím dat. Databáze je prakticky jediným rozumným řešením. Díky SQL je práce s daty podstatně rychlejší a poskytuje širokou škálu nástrojů pro jejich spravování. Pomocí PHP se poté jen odešle zformulovaný dotaz a získá se již zpracovaný výsledek.

Pokud se jedná o jednodušší projekt, ve kterém bude třeba připojovat se k DB jen v jediném souboru, je možné skript definující parametry připojení (a realizující připojení samotné) vložit přímo do něj. U větších projektů se však vyplatí pro tento účel vytvořit externí skript, který lze pak pokaždé snadno vložit.

Komunikace s DB pomocí myslql\_\*() je už zastaralou variantou a v novějších verzích PHP plánují vývojáři jejich podporu odstranit. Jejich funkce už v současnosti nahrazuje PDO\_MySQL.

#### <?php

/\*\*Data source name

\* Typ DB

\* Adresa server

\* Port serveru

\* Jméno DB, ke které se připojuje

\*/

#### $dsn = "mysql:host=127.0.0.1;port=3306;dbname=telefony";

/\*\*Uživatel, který může spravovat Db\*/

#### $user = "root";

/\*\*Heslo uživatele\*/

#### $password = "";

/\*\*Vytvoření instance PDO

\* PDO::ERR\_MODE\_EXCEPTION 🡪 chyby se ohlašují jako výjimky

\* PDO::MYSQL\_ATTR\_INIT\_COMMAND 🡪 Kódování

\*/

#### try{

#### $db = new PDO($dsn, $user, $password,

#### [PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

#### PDO::MYSQL\_ATTR\_INIT\_COMMAND => 'SET NAMES utf8',

#### PDO::ATTR\_PERSISTENT => TRUE]);

#### } catch(PDOException $e){

#### echo 'Connection failed: ' . $e->getTraceAsString();

#### }

#### ?>

V souborech, kde je třeba se připojit k DB, tak se pouze vyžádá soubor s připojením.

## Důležité funkce pro práci s DB

### Prepare

Příprava SQL dotazu. Používají se Placeholdery, za které se poté dosadí přesná data.

### Params

Pole parametrů, které se dosazují do dotazu.

### Execute

Spuštění dotazu.

### Fetch

PDO::FETCH\_BOTH, PDO::FETCH\_ASSOC…

Vrací pole, které představuje řádek, s parametry.