PHP a MySQL

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

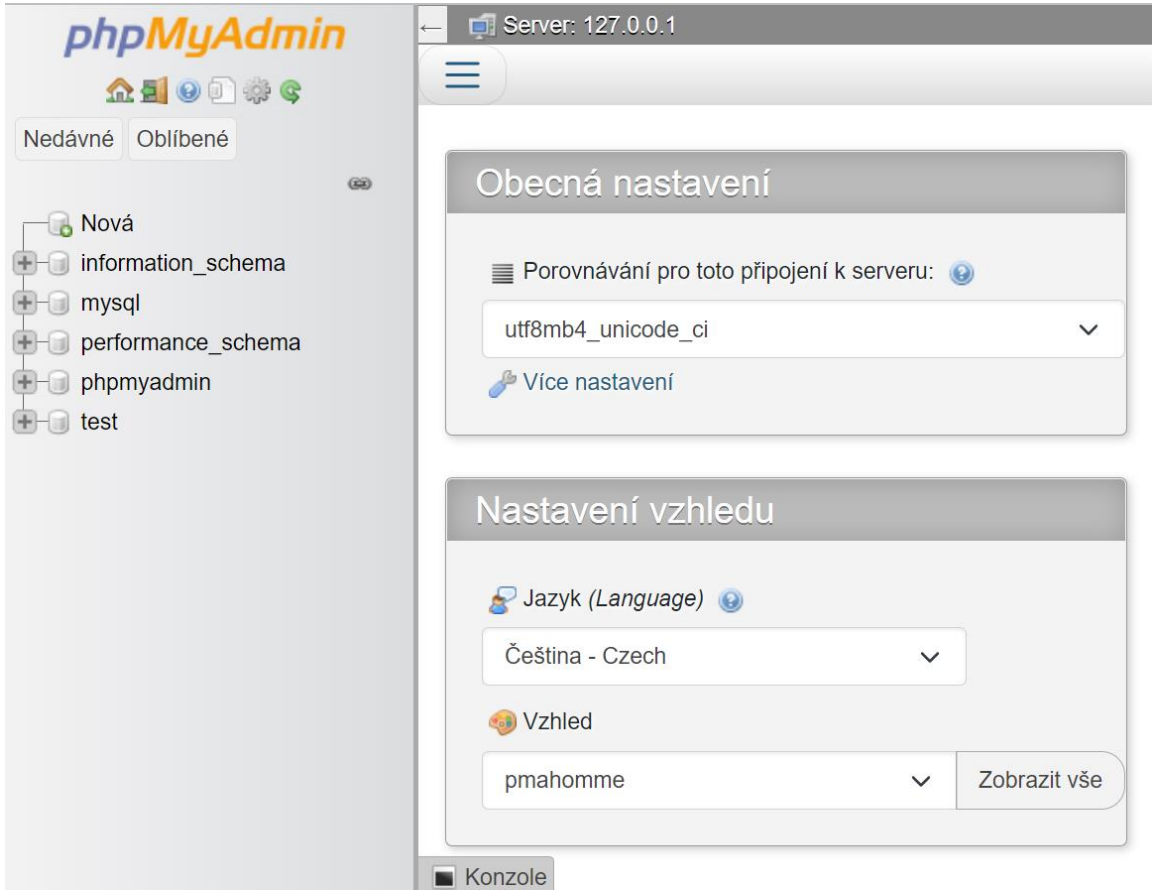
KMIWEBA Webové aplikace

PHP a MySQL

- Jazyk PHP podporuje práci s celou řadou různých databází.
- Nejběžněji se, zejména pro menší projekty využívají MySQL a MariaDB databáze.
- Přestože jsou MySQL a MariaDB rozdíly, mají mnoho společného.
- Pro jednoduchost se omezíme pouze na MySQL databázi.

Správa databáze

- Databázi je možné spravovat přímo prostřednictvím příkazové řádky nebo klienta (pokud je k dispozici).
- Při použité ve webovém prostředí se často využívají jednoduché administrační nástroje jako například phpMyAdmin nebo velmi populární český projekt Adminer (oba jsou napsány v jazyce PHP).
- Správu databáze si samozřejmě můžeme naprogramovat sami, ale typicky se struktura databáze vytváří při vývoji a po nasazení se příliš nemění.



Propojení PHP a MySQL databáze

- Pokud využíváme LAMP balíček, máme již MySQL databázi nainstalovanou a její podpora je v PHP aktivní.
- V jiných případech je obvykle nutné databázi nainstalovat a aktivovat mysql rozšíření.
- Případně je možné využít i jiné (například PDO).
- Rozšíření se spravují v souboru php.ini.
- Následuje příklad připojení k databázi.

Připojení k databázi

```php
<?php
  // připojení k lokální databázi (UNIX socket)
  $db_host = 'localhost'; // $db_host = '127.0.0.1'; // pro TCP/IP spojení
  $db_user = 'root';
  $db_password = 'root';
  $db_db = 'test';
  //$db_port = 8889; // pro TCP/IP spojení

  $mysqli = new mysqli(
    $db_host,
    $db_user,
    $db_password,
    $db_db,
    //$db_port // pro TCP/IP spojení
  );
```

Připojení k databázi

```php
if ($mysqli->connect_error) {
    echo 'Error: '.$mysqli->connect_error;
    exit();
  }

  echo 'Success: A proper connection to MySQL was made.';
  echo '';
  echo 'Host information: '.$mysqli->host_info;
  echo '';
  echo 'Protocol version: '.$mysqli->protocol_version;

  $mysqli->close();
?>
```

Dotaz na databázi

```php
<?php
  // dotaz na databázi
  $query_result = $mysqli->query("SELECT * FROM users;");

  // výpis výsledků (převod na asociativní pole, lze i jinak)
  while ($row = $query_result->fetch_assoc()) {
    echo $row['id'], $row['name'], $row['surname'], $row['date'];
  }
?>
```

Ověření vstupů

```php
<?php
  // pozor, velmi nebezpečné hrozí SQL injection (např. 1' OR '1=1)
  $name = $_POST['name'] ?? null;
  $surname = $_POST['surname'] ?? null;

  if($name && $surname) {
    $mysqli->query("INSERT INTO users (name, surname, date) VALUES ('$name',
'$surname', now());");
    echo $mysqli->error;
  }
?>
```

Ověření vstupů

- Ošetření vstupů je možné provést pomocí funkce real_escape_string().

```php
<?php
  // bezpečné řešení
  $name = $mysqli->real_escape_string($_POST['name']) ?? null;
  $surname = $mysqli->real_escape_string($_POST['surname']) ?? null;
?>
```

Ověření vstupů

- Případně je možné využít funkci prepare().

```
<?php
  $name = $_POST['name'] ?? null;
  $surname = $_POST['surname'] ?? null;

  if($name && $surname) {
    $query = $mysqli->prepare("INSERT INTO users (name, surname, date) VALUES
('?', '?', now());");
    $query->bind_param("ss", $name, $surname);
    $query->execute();
  }
?>
```

Udržení pořádku

- Pro potřeby udržení pořádku adoptujeme jednoduchou MVC architekturu.
- Jedná se o velmi triviální, ale pro naše potřeby dostatečný příklad MVC architektury, demonstující dělení na model, view a controller.
- Dodejme, že se jedná o tzv. monolitickou architekturu. Později ukážeme alternativu v podobě microservices.
- Plnohodnotné řešení by zahrnovalo ORM systém (za předpokladu, že by měl smysl) a šablonovací systém.

Úkoly

ÚKOL 1

Upravte administrační rozhraní tak, aby umožňovalo přihlašování a správu uživatelů. Je tedy třeba implementovat:

- přihlašovací obrazovku
- sekci Users, obsahující: výpis existujících uživatelů (tabulka); možnost přidání nového uživatele; editaci a smazání stávajících uživatelů
- funkci odhlášení
- přehled (výpis) posledních 10 přihlášených uživatelů v sekci Dashboard
- je třeba navrhnout strukturu tabulky (či tabulek) realizující výše uvedenou funkcionalitu
- Přihlášení uživatelé si budou moci změnit své heslo a informace ve svém profilu. Správci budou moci libovolně měnit uživatele. Uživatelé budou mít jméno, příjmení, e-mail (slouží jako login), telefon, pracovnu (krátký text), popisek (delší text), heslo a příznak zda jsou či nejsou správci.