Multimedia im Netz (Online Multimedia)

Wintersemester 2014/15

Übung 02 (Hauptfach)

Organization: Slides

- Slides for the tutorial (HF) in English from now on
- consistent with lecture...

Organization: Topics

#	Date	Topic
2	20.10.2014	Database access, MySQL, PHP
3	27.10.2014	HTML5 & JavaScript Introduction
4	03.11.2014	Web Interactivity: Drag & Drop with HTML5
5	10.11.2014	jQuery: DOM Traversal and Manipulation, jQuery UI
6	17.11.2014	jQuery: AJAX with PHP backend
7	24.11.2014	NodeJS: Introduction, static web content
8	01.12.2014	NodeJS: Express, RESTful Web-Apps
9	08.12.2014	Excursus: Google Charts, Google Maps, Design Patterns
10	15.12.2014	Mashups with a multitude of technologies
--	22.12.2014	Christmas break / programming consultations
11	12.01.2014	Web apps with offline storage
12	19.01.2014	Questions
13	26.01.2014	Buffer

PHP & MySQL

- Multiple functions and APIs available for PHP to work with databases:
  - mysql („Deprecated“ since PHP 5.5.0)
  - mysqli (i is for „improved“)
  - PDO (PHP Data Objects)
- „mysql“ is still supported for older PHP versions
- It is highly recommendable to use mysqli or PDO

MySQL at the CIP-Pool

- Access “Datenbank Management” here: https://tools.rz.ifi.lmu.de/
- Create a new account (required)
- Create a new database (required)
- Connect to db2.cip.ifi.lmu.de

Test Connection

```php
<?php
    $c = mysql_connect("localhost", "user", "password");

    if($c){
        echo "Connection to database has been established.";
    } else {
        echo "Could not connect to database";
    }
?>
```

Mysql (I)

- Establish connection
```
$c = mysql_connect("localhost", "user", "password");
```
- Create database
```
$query = "CREATE DATABASE mydb";
mysql_query($query, $c);
```
- Select database for further calls
```
mysql_select_db("mydb‟);
```
- Create table
```
$query = "CREATE TABLE Personen(Name CHAR(30))";
mysql_query($query, $c);
```
- Close connection
```
mysql_close($c)
```

Mysql (II)

- Exemplary queries:

```
$query = "SELECT name FROM mydb";
$query = "INSERT INTO mydb VALUES ('$name')";
$query = "UPDATE mydb SET name='$name' WHERE pId=2";
```

- Further information: http://dev.mysql.com/doc/

- PHP-statement for a MySQL Query

```
$results = mysql_query($query);
```

- Process the results

```
mysql_fetch_array($result);
mysql_fetch_array($result, MYSQL_ASSOC);
mysql_fetch_array($result, MYSQL_NUM);
```

Mysqli

- Offers two interfaces
  - procedural
  - object-oriented
- Supports „prepared“ statements (recommendable)
- Supports multiple statements within one query
- Supports transactions
- Improved debugging tools

Mysqli (procedural)

- Establish connection
```
$c = mysqli_connect("localhost", "user", "password",
"mydb");
```
- Select database
```
mysqli_select_db("mydb“);
```
- Close connection
```
mysqli_close($c)
```
- PHP statement for MySQL query
```
$results = mysqli_query($c, $query);
```
- Process the results:
```
mysqli_fetch_array($result);
mysqli_fetch_array($result, MYSQLI_NUM);
mysqli_fetch_array($result, MYSQLI_ASSOC);
```

Object Oriented Programming in PHP (I)

- OOP paradigms / concept
  - classes
  - objects
- PHP class signature:

classMMN.php

```php
<?php
    class classMMN{

        //put members and methods here

    }
?>
```

Object Oriented Programming in PHP (II)

Define member variables of a class:

classMMN.php

```php
<?php
    class classMMN{
        var $name = "Multimedia im Netz";
        var $semester = "Wintersemester 2014/15";
        var $professor = "Prof. Dr. Heinrich Hussmann";
        var $termin = "Donnerstag: 10-13 Uhr";
    }
?>
```

Object Oriented Programming in PHP (III)

- Importing and instantiating a class

mmn.php

```php
<?php
    require_once("classMMN.php");
    $mmn = new classMMN();
?>
```

- Object access

mmn.php (Fortsetzung)

```php
echo "Veranstaltung: " . $mmn->name . "</br>";
echo "Semester: " . $mmn->semester . "</br>";
echo "Professor: " . $mmn->professor . "</br>";
```

Object Oriented Programming in PHP (III)

- Add methods:

classMMN.php

```php
<?php
    class classMMN{
        var $name = "Multimedia im Netz";
        var $semester = "Wintersemester 2014/15";
        var $professor = "Prof. Dr. Heinrich Hussmann";
        var $termin = "Donnerstag: 10-13 Uhr";

        function setTermin(var $termin){
            $this->termin = $termin;
        }

        function getTermin(){
            return $this->termin;
        }
    }
?>
```

Object Oriented Programming in PHP (IV)

- Call methods

mmn.php

```php
<?php
    $mmn->setTermin("Mittwoch: 10-13 Uhr");
    echo $mmn->getTermin();
?>
```

Object Oriented Programming in PHP (V)

- Constructor: PHP‘s constructors are methods with a special name: **__construct();**

```
function __construct($name, $sem, $prof, $termin){
    $this->name = $name;
    $this->semester = $semester;
    $this->professor = $professor;
    $this->termin = $termin;
}
```

- Use constructor:

```
$mmn = new classMMN("MMI2", "WS 2014/15",
                    "Prof. Dr. Butz", "Freitag: 14-16 Uhr");
```

Mysqli (object oriented)

- Establish connection
```
$c = new mysqli("localhost", "root", "", "mydb");
```
- PHP statement for MySQL query
```
$results = $c->query($query);
```
- Process the results
```
$results->fetch_assoc();
$results->fetch_row();

$results->fetch_all(MYSQLI_BOTH);
$results->fetch_all(MYSQLI_ASSOC);
$results->fetch_all(MYSQLI_NUM);
```
- Close the connection
```
$c->close();
```

Mysqli: Prepared Statements (I)

- Separate structure and data through the use of wildcards. In the query we use „**?**“ as wildcards

- **Advantages:**
  - You can reuse the query with different parameters
  - More secure (cf. SQL Injections)

- **How to do it:**
  - „Prepare“: Prepare the query. The template is checked for errors an.
  - „Bind“: Bind the parameters to the wildcards
  - „Execute“: The query is executed with the passes parameters

Mysqli: Prepared Statements (II)

- Query with wildcards
```
$q = "SELECT Nachname FROM myDB WHERE Vorname=?";
```

- Prepare the query
```
$query = $c->prepare($q);
```

- Bind the parameters
```
$name = "Isabell";
$query->bind_param("s", $name);
```

- Execute the query
```
$query->execute();
```

Mysqli: Prepared Statements (III)

- Bind result columns to variables
  `$query->bind_result($nachname);`

- Fetch results
  `$query->fetch()`

Example: Prepared Statements

```php
<?php
    include_once 'db_info.php';

    $c = new mysqli($host, $user, $password, $db);
    $query = $c->prepare("SELECT Nachname FROM Uebung02 WHERE Vorname=?");

    $name = "Isabell";
    $query->bind_param("s", $name);
    $query->execute();
    $query->bind_result($nachname);

    while($query->fetch()){
        echo $nachname;
        echo "<br/>";
    }
?>
```

Cryptographic Hashes

- md5()
  - Message Digest Algorithm5
  - generates a 128bit hash value
  - fast, but vulnerable
- sha1()
  - generates a 160bit hash value
  - used in most (older) SSL certificates (they need to be replaced soon)
  - fast, but vulnerable

Password Hashing in PHP (I)

- PHP has built-in password hashing functions
  - password_hash()
  - password_verify()
  - ...
- Don‘t store plain text passwords in databases.... Ever ;)
- Advantages:
  - More secure
  - easy to use
- Disadvantages:
  - only available with PHP >= 5.5.0

Password Hashing in PHP (II)

- Hashing a password:

```
$pwHash = password_hash("password1234",PASSWORD_DEFAULT);
```

- Verifying a hash:

```
if(password_verify("password1234",$pwHash)){
    echo “Your password is correct";
}
```

Assignment 2

- **Thema: Picture Gallery, Authentication**
- Due in: 1 Week
- Due date: 27.10.2014 12:00

Thanks!

What are your questions or comments?