PHP Workshop
Outline

• PHP Classes (Brief Overview)
• Accessing MySQL via the mysqli class
  – Connecting to MySQL
  – Querying the MySQL
  – Retrieving results from MySQL
• DEMO
• Notes for Assignment 4
Classes and OOP
(Basic Conceptual Overview)

Define the blueprint of your own class (object)

Instantiate instances of your object and use them in your program

class dog{
  //properties
  //methods
}

$ben = new dog;

$sunny = new dog;

$sugar = new dog;

Reading: http://codular.com/introducing-php-classes
Defining Classes

class dog {
    //Properties
    public $name;
    public $breed;

    //Methods
    public function bark(){
        echo $this->name . " barked... Woof!";
    }

    //Constructor (optional)
    public __construct($nameOfDog){
        $this->name = $nameOfDog;
    }
}
Working with Classes

$sugar = new dog(“Sugar”); //pass “Sugar” to constructor

$sugar->breed = “German Shepherd”; //set breed property equal to “German Shepherd”

echo $sugar->name . “ is a “ . $sugar->breed;
   >>Sugar is a German Shepherd

$sugar->bark(); //call the “bark” method
   >>sugar barked... Woof!
Mysqli Class

The mysqli class is the main class you can use to:

• Set up a connection to the MySQL database
• Send SQL queries to the MySQL database (CRUD Operations)
• Read results (if any) from the query
• Check for any error connecting to or executing SQL queries on a MySQL database

Reading: http://www.w3schools.com/php/php_ref_mysqli.asp
1. Connecting to DB with Mysqli

//Instantiate mysqli object & create connection
$db = new mysqli("localhost", "username", "password", "database_name");

//Check for any errors connecting
if($db->connect_errno){
    //There was an error connecting to the database. Put code here on what you would like to about it like...
    echo "Error: " . $db->connect_error;
}
else{
    //Put code here when you connect to the database.
}

Example: post.php, get.php, get2.php
2. Querying the DB via Mysqli (Single Query)

//Construct some query (it’s just a string)

$query_noresult = "INSERT INTO ...";
$query_result = "SELECT * FROM DIVECUS";

2 Types of Queries

No Result Queries
INSERT, DELETE, CREATE, etc...

Result Queries
SELECT

Example: post.php, get.php, get2.php
3. Retrieving results from MySQL
(No Result Queries)

```php
If($db->query($query_noresult) === TRUE){
    //Your query worked, yay
}
```

Note about INSERT SQL queries
• You can use `$db->insert_id;` to retrieve the AUTO_INCREMENT ID generated for an inserted record. You do NOT need to run a separate query to get that ID value.
```php
if($result = $db->query($query_result)){

    echo $result->num_rows;

    while($row = $result->fetch_array()){ // echo $row[‘Name’];

    }

    $result->free();
}
```

Example: get.php, get2.php
3.1. Result Modes

You can return results in 2 ways (result modes)

$db->query($query_result, MYSQLI_STORE_RESULT)
• Return entire results to memory
• Can use $result->data_seek(index) to jump to different rows in your result set (i.e. after you loop through the results, do $result->data_seek(0); to go to starting point to loop again)
• Default (this is the executed mode if you don’t specify a “result mode”)

$db->query($query_result, MYSQLI_USE_RESULT)
• MySQL “spoon feeds” the rows to server running PHP each time a fetches a row
• Useful if expecting a large dataset (too large to put in memory of PHP server)
• Must free results in order to run another query ($result->free();)

Example: get.php, get2.php
4. Closing Connections

When you are done using the database, make sure to close the connection...

$db->close();
Before DEMO time

You may need to know

• Super Global Variables
• Passing Variables
• JSON
• AJAX
• Security Issues (SQL Injection and XSS)
• Import & Export DB
Super Global Variables

There are predefined “Super Global” variables that are made available to you through the PHP runtime that you can use within your PHP code.

<table>
<thead>
<tr>
<th>Super Global</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>$_SERVER</td>
<td>Contains info about the web server environment such as Server Name, Request Method, Document Root, etc.</td>
</tr>
<tr>
<td>$_GET</td>
<td>Contains any GET variables in the URL query string</td>
</tr>
<tr>
<td>$_POST</td>
<td>Contains any POST variables submitted via a form post submission</td>
</tr>
<tr>
<td>$_COOKIE</td>
<td>Contains any HTTP Cookie Info</td>
</tr>
<tr>
<td>$_FILES</td>
<td>Contains information on POST file uploads</td>
</tr>
<tr>
<td>$_SESSION</td>
<td>Contains information about any variables registered in a session (if created)</td>
</tr>
</tbody>
</table>

There are some other super globals but these are the main ones...
Passing Variable to PHP (GET)


Start of query string | key=value pairs | Separator

```php
<?php
    $var1 = $_GET['query'];
    $var2 = $_GET['lang'];
?>
```

Note - certain characters cannot be put in the URL (such as space character). You will to “URL Encode” those special characters. For example, a space character will show up as a %20 in the URL query string.

Reference: http://www.url-encode-decode.com/
Passing Variables to PHP (POST)

Your form html or php file

```html
<form method="post" action="post.php">
  <input type="text" name="fname"/>
  <input type="text" name="lname" />
  <input type="submit" value="Submit" />
</form>
```

The `post.php` file that will process the request (which could be the same as the posting file)

```php
<?php
  $var1 = $_POST['fname'];
  $var2 = $_POST['lname'];
?>
```

HTTP POST Request to `post.php`

Fname=Chan
Lname=Kim
JSON

- JSON: JavaScript Object Notation
- JSON is syntax for storing and exchanging text information. Much like XML. JSON is smaller than XML, and faster and easier to parse.

```javascript
var Name = { "firstName":"John", "lastName":"Doe" }
Name.firstName
>> John

json= [ {assign1:42, assign2:38, assign3:45, full_name:'Ranju', id:100},
{assign1:29, assign2:26, assign3:28, full_name:'Manaz', id:101}];

$.each(json, function(k, v){
    myTable += v.id+v.full_name+v.assign1+v.assign2+v.assign3;
});
```

Reference: http://www.w3schools.com/json/
AJAX

• AJAX: Asynchronous JavaScript and XML.
• AJAX is **not** a new programming language, but a new way to use existing standards.

• AJAX is the art of exchanging data with a server, and **updating parts** of a web page – without reloading the whole page.

Security (SQL Injection and XSS)

HTTP POST Request
fname=Arian
lname=SQL Code

A malicious user can submit (POST or GET)
• SQL Code → SQL Injection Attack, or
• HTML tags that could be anything from
  a form to a <script> tag → XSS Attack.

In your PHP...

$query = "INSERT INTO table (fname, lname)";
$query .= "VALUES (" . "$_POST["fname"]", " . "$_POST["lname"]")";"
Preventing SQL Injection & XSS

To prevent this, you have to sanitize your input variables and make sure you output safe HTML

///Sanitize → SQL Injection
$sanitized_variable = $db->real_escape_string($_POST['lname']);

///Output Safe HTML → XSS
echo htmlspecialchars($row['lname']);

You can try to sanitize for HTML tags by using strip_tags($_POST[]) before you input the value into the database. Just make sure that is what you want to do...

Example: post.php
Additional Security Tip

Don’t type out the username and password when instantiating mysqli. Instead, create a special PHP file that defines certain constants outside the root of your website which you can then include and use in your PHP code.

```php
define(SQL_PASSWORD, "mypassword");```

Example: configure.php
Notes for assignment 4

- Sample reports

Print View of each table
Notes for assignment 4

• Please send your DB dump file for TAs testing

Export from a Database, not from each table

Reading: http://www.howtogeek.com/howto/mysql/backup-mysql-database-to-a-file/
Demo Time

demo Time

example.php : main page
student.sql : DB dump
configure.php : configuration of DB

get.php : php codes for get example
get2.php : php codes for get example 2
post.php : php codes for post example
script.js : AJAX call
style.css : css file

Student_ex

Id
full_name
Assign1
assign2
assign3