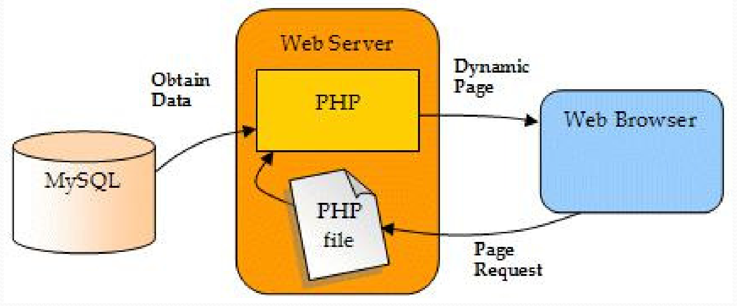
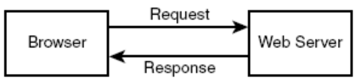
Programming IV: SCS2108

Handout 3: PHP Databases

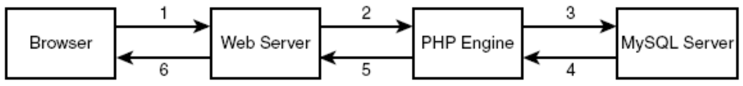
Prasad Wimalaratne, PhD, SMIEEE
spw@ucsc.cmb.ac.lk



Client Server Relationship



The client/server relationship between a web browser and web server requires communication.



The basic web database architecture consists of the web browser, web server, scripting engine, and database server.

Create database/ tables

- CREATE DATABASE testdb;

```
CREATE TABLE `symbols` (
  `id` int(11) NOT NULL auto_increment,
  `country` varchar(255) NOT NULL default '',
  `animal` varchar(255) NOT NULL default '',
  PRIMARY KEY  (`id`)
) TYPE=MyISAM;

INSERT INTO `symbols` VALUES (1, 'America', 'eagle');
INSERT INTO `symbols` VALUES (2, 'China', 'dragon');
INSERT INTO `symbols` VALUES (3, 'England', 'lion');
INSERT INTO `symbols` VALUES (4, 'India', 'tiger');
INSERT INTO `symbols` VALUES (5, 'Australia', 'kangaroo');
INSERT INTO `symbols` VALUES (6, 'Norway', 'elk');
```

- A **database engine** (or **storage engine**) is the underlying software component that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database. Most database management systems include their own application programming interface (API) that allows the user to interact with their underlying engine without going through the user interface of the DBMS.
- The term "database engine" is frequently used interchangeably with "database server" or "database management system". A 'database instance' refers to the processes and memory structures of the running **database engine**
- Note :
- **MyISAM** is the default storage engine for the MySQL relational database management system versions prior to 5.5
- MySQL 5.5 and greater have switched to the InnoDB engine to ensure referential integrity constraints, and higher concurrency.

```
<html>
<head>
<basefont face="Arial">
</head>
<body>

<?php

// set database server access variables:
$host = "localhost";
$user = "test";
$pass = "test";
$db = "testdb";

// open connection
$connection = mysql_connect($host, $user, $pass) or die
("Unable to connect!");

// select database
mysql_select_db($db) or die ("Unable to select database!");
```

1

2

3

```
// create query
$query = "SELECT * FROM symbols";

// execute query
$result = mysql_query($query) or die ("Error in query:
$query. ".mysql_error());
// see if any rows were returned
```

4

5

```
if (mysql_num_rows($result) > 0) {
    // yes
    // print them one after another
    echo "<table cellpadding=10 border=1>";
    while($row = mysql_fetch_row($result)) {
        echo "<tr>";
        echo "<td>".$row[0]."</td>";
        echo "<td>" . $row[1]."</td>";
        echo "<td>".$row[2]."</td>";
        echo "</tr>";
    }
    echo "</table>";
}
else {
    // no
    // print status message
    echo "No rows found!";
}
```

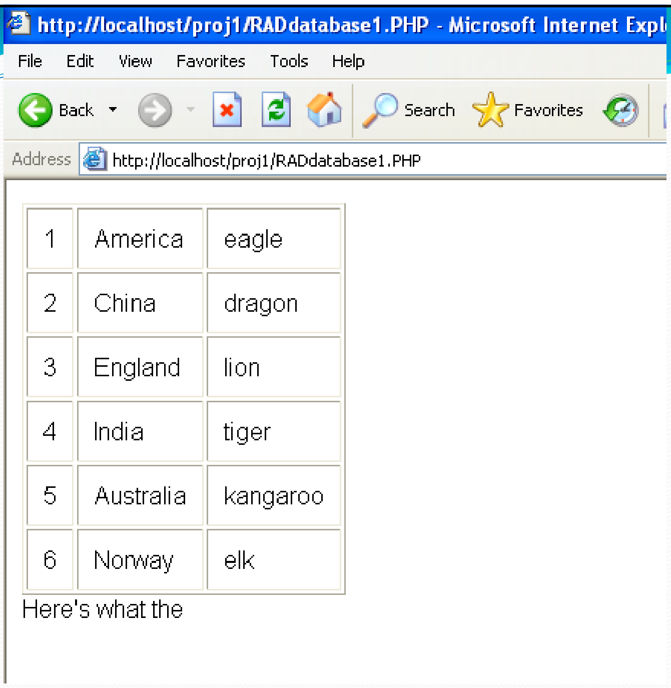
6

```
// free result set memory
mysql_free_result($result);

// close connection
mysql_close($connection);

?>

</body>
</html>
```

7

8

- Here's what the


http://localhost/proj1/RADdatabase1.PHP - Microsoft Internet Expl
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
http://localhost/proj1/RADdatabase1.PHP

1	America	eagle
2	China	dragon
3	England	lion
4	India	tiger
5	Australia	kangaroo
6	Norway	elk

Here's what the

Database Connectivity- Steps

- first specify information needed to establish a connection to the database server.
- This information includes the server name, the username and password required <u>to gain access</u> to it, and the name of the database <u>to query</u>. These values are all set up in regular PHP variables.
- **<?php**

```
$host = "localhost";
$user = "test";
$pass = "test";
$db = "testdb";

?>
```

Database Connectivity- Steps

- To begin communication with a MySQL database server, you need to open a connection to that server. All communication between PHP and the database server takes place through this connection. In order to initialize this connection, PHP offers the mysql_connect() function:
- **<?php**

  $connection = mysql_connect($server, $user, $pass);

  ?>
- All the parameters in mysql_connect() are optional, but there are three you will generally need to use anywhere beyond your own machine: the database server name, username and password. If the database server and the Web server are running on the same physical machine, you can use localhost as the database server name – this is in fact the default value supplied by PHP.
- mysql_connect() returns a "link identifier", which is stored in the variable $connection.
- This identifier is used when communicating with the database.

Database Connectivity- Steps

- Once you have a connection to the database server, you must select a database for use with the mysql_select_db() function
- : **<?php**

  mysql_select_db($db) or die ("Unable to select database!");

  ?>
- This function must pass the name of the database to be used for all subsequent queries.
- An optional second argument here is the link identifier; if no identifier is specified, the last opened link is assumed.
- If you have two or more database connections open simultaneously, it's a good idea to specify the link identifier as the second argument to mysql_select_db() -

Database Connectivity- Steps

- The next step is to create the query and execute it. This is accomplished with the mysql_query() function.
- **<?php**

  $query = "SELECT * FROM symbols";
  $result = mysql_query($query) or die ("Error in query: $query. ".mysql_error());

  ?>
- This function also needs two parameters: the query string and the link identifier for the connection. Again, if no link identifier is specified, the last opened link is used.
- Depending on whether or not the query was successful, the function returns true or false; a failure can be caught via the ...or die() clause of the statement, and the mysql_error() function can be used to display the corresponding error message.

Database Connectivity- Steps

- If mysql_query() is successful, the result set returned by the query is stored in the variable $result.
- This result set may contain one or more rows or columns of data, depending on your query.
- You can retrieve specific subsets of the result set with different PHP functions, including the one used here - the mysql_fetch_row() function - which fetches a single row of data as an array called $row. Fields in that row can then be accessed using standard PHP array notation.

- Each time you call mysql_fetch_row(), the next record in the result set is returned. This makes mysql_fetch_row() very suitable for use in a while() or for() loop.
- **<?php**

```
if (mysql_num_rows($result) > 0) {
   while($row = mysql_fetch_row($result)) {
      echo "<td>".$row[0]."</td>";
      echo "<td>".$row[1]."</td>";
      echo "<td>".$row[2]."</td>";
   }
}

?>
```

Querying a Database from the Web

- Basic Steps in Accessing database from web:
  1. Check and filter data coming from the user.
  2. Set up a connection to the appropriate database.
  3. Query the database.
  4. Retrieve the results.
  5. Present the results back to the user.

Checking and Filtering Input Data

- Trimming White Spaces
  - E.g
  - trim() function
    - This function returns a string with whitespace stripped from the beginning and end of *string*
  - $searchterm=trim($searchterm);
  - ‘ abc ‘ $\longrightarrow$ ‘abc’

Strings: Single Quotes vs Double Quotes

- // both output ‘Hello World‘
  - $str = ‘World';
  - echo ‘Hello' . $str;
  - echo “Hello $str";
- Note
  - Parsing variables within strings uses more memory than string concatenation.
  - When writing a PHP script in which memory usage is a concern, consider using the concatenation operator (.) rather than variable parsing.

Formatting Strings for Storage: addslashes() and stripslashes()

- Certain characters are valid as part of a string but can cause problems, particularly when you are inserting data into a database because the database could interpret these characters as control characters.
- The problematic ones are the quotation marks (single and double), backslashes (\), and the NULL character.
- You need to find a way of marking or *escaping* these characters so that databases such as MySQL can understand that you meant a **literal special character** rather than **a control sequence**.

Formatting Strings for Storage: addslashes() and stripslashes()

- To *escape* these characters, add a backslash in front of them.
  - For example, “ (double quotation mark) becomes \” (backslash double quotation mark), and \ (backslash) becomes \\ (backslash backslash).
    - This rule applies universally to special characters, so if you have \\ in your string, you need to replace it with \\\\.
- PHP provides two functions specifically designed for escaping characters.
- Before you write any strings into a database, you should reformat them with addslashes(),
  - $feedback = addslashes($feedback);

Setting Up a Database Connection

- PHP5 has a library for connecting to MySQL.
- This library is called mysqli (the *i* stands for improved).
- The mysqli library is suitable for use with MySQL version 4 and later.
- At version 4, a new connection protocol that is much faster was added to MySQL, and mysqli allows you to take advantage of it.
- The mysqli library allows you to use either an object-oriented or procedural syntax.

Setting Up a Connection (ctd..)

- To begin communication with a MySQL database server, a connection has to be established to the server.
- All communication between PHP and the database server takes place through this connection.
- In order to initialize this connection, PHP offers the mysql_connect() function

Setting Up a Connection (ctd..)

- You use the following line in the script to connect to the MySQL server:

```
@ $db = new mysqli('localhost', 'bookorama',
   'bookorama123', 'books');
```

- This line instantiates the mysqli class and creates a connection to host '*localhost*' with username '*bookorama*', and password '*bookorama123*'.
- The connection is set up to use the database called 'books'.

Setting Up a Connection (ctd..)

- The object-oriented approach invokes methods on this object to access the database.
- If procedural approach is preferred, mysqli provides a procedural function. To connect in a procedural fashion, use

```
@ $db = mysqli_connect('localhost', 'bookorama',
   'bookorama123', 'books');
```

- function returns a resource rather than an object.
- resource represents the connection to the database, and you need to pass this resource into all the other mysqli functions.
- method is similar to the file-handling functions, such as fopen().

Setting Up a Connection (ctd..)

- Most of the mysqli functions have an object-oriented interface and a procedural interface.
- Generally, the differences are that the procedural version function names start with mysqli_ and is required to pass in the resource handle obtained from mysqli_connect().
- Database connections are an exception to this rule because they can be made by the mysqli object's constructor.

Setting Up a Connection (ctd..)

- The result of an connection function call has to be checked for a valid database connection.

```
if (mysqli_connect_errno())
{
echo 'Error: Could not connect to database. Please try again later.';
exit;
}
```

- (This code is the same for the object-oriented and procedural versions.)
- The mysqli_connect_errno() function returns an error number on error, or zero on success.

Setting Up a Connection (ctd..)

- Note the connection statement above, the line of code begins with the error suppression operator, @.
  - The the "@" operator suppresses error messages, not return values.
  - This handles any errors gracefully. (This could also be done with exceptions)

Setting Up a Connection (ctd..)

- **The Error Suppression Operator**
  - The error suppression operator (@) can be used in front of any expression.
  - For example,
  - $a = @(57/0);
  - Without the @ operator, this line generates a divide-by-zero warning. With the operator included, the error is suppressed.
  - If you are suppressing warnings in this way, you should write some error handling code to check when a warning has occurred.
  - If you have PHP set up with the track_errors feature enabled, the error message will be stored in the global variable $php_errormsg.
  - The behaviour of these functions is affected by (see settings in \wamp\Apache2\bin\php.ini)

Choosing a Database to Use

- The database to be used is specified as a parameter to the mysqli constructor or the mysqli_connect() function.
  mysqli_select_db() function can be used to change the default database.
- It can be accessed as either
  $db->select_db(*dbname*)
  or as
  mysqli_select_db(*db_resource, db_name*)
- The procedural version begins with mysqli_ and requires the extra database handle parameter.

- This function should only be used to change the default database for the connection

Querying the Database

- To perform the query, you can use the mysqli_query() function.
- set up the query to run:
  $query = “select * from books where $searchtype like ‘%$searchterm%’”;

  Use of ‘ and “
- E.g: search for the user-input value ($searchterm) in the field the user specified ($searchtype).
- Notice the use of ‘like’ for matching rather than equal.
- E.g SELECT * FROM Books
  WHERE tile like ‘%PHP%';

Querying the Database(ctd..)

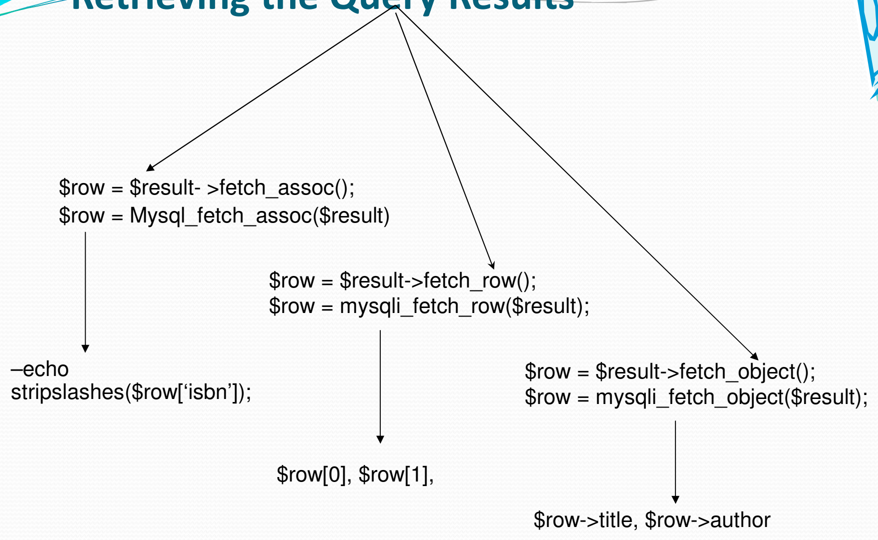
- You can now run the query:

  $result = $db->query($query);
- Or, using the procedural interface

  $result = mysqli_query($db, $query);

- pass in the query to run and, in the procedural interface, the database link (again, in this case $db).
- The object-oriented version returns a result object; the procedural version returns a result resource.
- Both stores the result in a variable ($result) for later use.
- This function returns false on failure.

Retrieving the Query Results



```
$row = $result- >fetch_assoc();
$row = Mysql_fetch_assoc($result)
```

```
–echo
stripslashes($row['isbn']);
```

```
$row = $result->fetch_row();
$row = mysqli_fetch_row($result);
```

```
$row[0], $row[1],
```

```
$row = $result->fetch_object();
$row = mysqli_fetch_object($result);
```

```
$row->title, $row->author
```

Retrieving the Query Results

- One can think of a telephone book as an example of an associative array, where names are the **keys** and phone numbers are the **values**. Using the usual array-like notation, we might write
  - telephone['saman'] = ' 0112 123456 '
  - telephone['Amal'] = '0112789123'
- These entries can be thought of as two records in a database table
  - To retrieve the element from the associative array, we use a similar notation i.e.
  - x = telephone['saman']
  - y = telephone['Amal']

name	telephone
saman	0112 123456
Amal	0112789123

- Further Example: Dictionary
  - dictionary['circle'] = 'a round plane figure whose boundary consists of points equidistant from the centre '
  - dictionary['rectangle'] = 'a plane figure with four straight sides and four right angles, and with unequal adjacent sides. '

Retrieving the Query Results

- A large variety of functions is available to extract data out of the object or resource.
- The result object or identifier is the key to accessing the rows returned by the query.
- Eg. : count the number of rows returned using mysqli_fetch_assoc() function.
- In object-oriented approach, the number of rows returned is stored in the num_rows member of the result object
- $num_results = $result->num_rows;

Retrieving the Query Results (ctd..)

- In procedural approach, the function mysqli_num_rows() returns number of rows extracted by the query.
- Eg:

```
$num_results =  mysqli_num_rows($result);
for ($i=0; $i <$num_results; $i++)
{
// process results
}
```

1. Retrieving the Query Results (ctd..)

- In each iteration call

  $result->**fetch_assoc**()
- Or

  mysqli_fetch_assoc().

- This function takes each row from the result set and returns the row as an array, with each key an attribute name and each value the corresponding value in the array:

  $row = $result->fetch_assoc();

  Or you can use a procedural approach:

  $row = mysqli_fetch_assoc($result);

2. Retrieving the Query Results (ctd..)

- Given the array $row, traverse through each field
- example:
  - echo '<br />ISBN: ';
  - echo stripslashes($row['isbn']);
- Call stripslashes() to tidy up the value before displaying it.
- Several variations can be used to extract results from a result identifier. Instead of an array with named keys, you can retrieve the results in an enumerated array with mysqli_fetch_row(), as follows:
- $row = $result->**fetch_row**($result);
- or
- $row = mysqli_fetch_row($result);

3. Retrieving the Query Results (ctd..)

- Attribute values extracted using $row[0], $row[1], and so on.
- You could also fetch a row into an object with the mysqli_fetch_object() function:
  - $row = $result->fetch_object();
  - or
  - $row = mysqli_fetch_object($result);
- You can then access each of the attributes via $row->title, $row->author, and so on.

Disconnecting from the Database

- free up the result set by calling either
  - $result->free();
  - or
  - mysqli_free_result($result);
- You can then use
  - $db->close();
  - or
  - mysqli_close(*$db*);
- to close a database connection. Using this command isn't strictly necessary because the connection will be closed when a script finishes execution anyway

Summary: PHP and Database

- ***Mysql*** – popular open-source database management system
- ***PHP*** usually works with ***Mysql*** for web-based database applications
- ***LAMP*** applications—Web-based applications that use ***Lynux***, ***Apache***, *Mysql*, and ***php/pearl/python***

Basic Steps to Process DB

1. Connect to host server which has Mysql installed
2. Select a database
3. Form an SQL statement
4. Execute the SQL statement and (optionally) return a record set
5. Extract data from recordset using php
6. Close connection

Connect to Mysql

```
<?php
$host = 'localhost';
$username = 'peter';
$pswd = '!?+&*';
$dbName = "myDB";
$con = mysql_connect($host, $username,
                     $pswd);
if (!$con){
  die('Could not connect: '
      . mysql_error());
}
$db = @mysql_select_db($dbName,
           $con) or die(mysql_error());
?> //The die() function prints a message and exits th
current script.
```

Create a Database

- SQL
  - `CREATE DATABASE database_name`
- PHP

```
$con = mysql_connect("localhost","peter",
              "abc123");
$sql = "CREATE DATABASE myDB";

mysql_query("$sql", $con));
```

Create a Table

- SQL
  - CREATE TABLE table_name

```
(column_name1 data_type,
 column_name2 data_type,
 column_name3 data_type,
 ....
)
```

Create a Table

PHP

- PHP

```
// Connect to Mysql
$con = mysql_connect(. . .);

// Create database
mysql_query("CREATE DATABASE my_db",$con);

// Select DB
mysql_select_db("my_db", $con);

// Create table
$sql = "CREATE TABLE Persons(
         FirstName varchar(15),
         LastName varchar(15),
         Age int
        )";
// Execute SQL statement
mysql_query($sql, $con);
```

Select a Database

- When DB already exists:
- PHP

```
$con =
mysql_connect("localhost","peter",
               "abc123");

$db = mysql_select_db("my_db“,
                      $con);
```

Executing a SELECT Query

- SQL

```
SELECT colName1, colName2, colName3
FROM Persons;
```

- PHP

```
$con = mysql_connect(. . .);
mysql_select_db("my_db“, $con);
$sql = “SELECT FirstName, LastName
         FROM Persons;”;
$result = mysql_query($sql);
```

Printing Results of SQL Statement

- PHP

```
$result = mysql_query($sql);

while($row =
       mysql_fetch_array($result)){
  echo $row['FirstName'] . " " .
       $row['LastName'];
  echo "<br />";
}
```

Inserting Record into Table

- SQL

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)

or

INSERT INTO table_name
                   (column1, column2,
column3,...)
VALUES (value1, value2, value3,...)
```

Inserting Record into Table

```
<?php
$con = mysql_connect(“…",“…",“…");
if (!$con)
  die('Could not connect: ' . mysql_error());
mysql_select_db("my_db", $con);
mysql_query("INSERT INTO Persons (FirstName,
            LastName, Age)
            VALUES (‘Saman',‘Perera','35')");

mysql_query("INSERT INTO Persons (FirstName,
            LastName, Age)
            VALUES (‘Ajith', ‘Piris', '33')");
mysql_close($con);
?>
```

Inserting Record into Table From HTML Form

```
<html>
<body>

<form action="insert.php" method="post">
Firstname: <input type="text" name=“firstname" />
Lastname: <input type="text" name=“lastname" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>

</body>
</html>
```

Inserting Record into Table From HTML Form

```
<?php
$con = mysql_connect("…","…","…");
if (!$con)
  die('Could not connect: ' . mysql_error());
mysql_select_db("my_db", $con);
$sql="INSERT INTO Persons
                      (FirstName, LastName, Age)
      VALUES ('$_POST[firstname]',
              '$_POST[lastname]',
              '$_POST[age]')";
if (!mysql_query($sql,$con))
   die('Error: ' . mysql_error());
echo "1 record added";
mysql_close($con)
?>
```

Update Record

- SQL

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column = some_value;
```

Update Record

```
<?php
$con = mysql_connect("…","…","…");
if (!$con)
  die('Could not connect: ' .
                                    mysql_error());
mysql_select_db("my_db", $con);
$sql = "UPDATE Persons
          SET    Age = '36'
          WHERE  FirstName = 'Saman'
          AND    LastName = "Perera'";
mysql_query($sql, $con);
mysql_close($con);
?>
```

Delete Record

- SQL

```
Delete table_name
WHERE some_column = some_value;
```

Delete Record

```
<?php
$con = mysql_connect(“…",“…",“…");
if (!$con)
  die('Could not connect: ' .
                              mysql_error());
mysql_select_db("my_db", $con);
$sql = “DELETE FROM Persons
        WHERE FirstName = ‘Saman’
        AND LastName = “Perera’”);
mysql_query($sql, $con);
mysql_close($con);
?>
```

Using Other PHP-Database Interfaces

Using Other PHP-Database Interfaces

- PHP supports libraries for connecting to a large number of databases, including Oracle, Microsoft SQL Server, and PostgreSQL.
- In general, the principles of connecting to and querying any of these databases are much the same.
- The individual function names vary, and different databases have slightly different functionality,

PEAR

- What is PEAR?
- PEAR is short for "PHP Extension and Application Repository" and is pronounced just like the fruit.
- Refer:

http://pear.php.net/

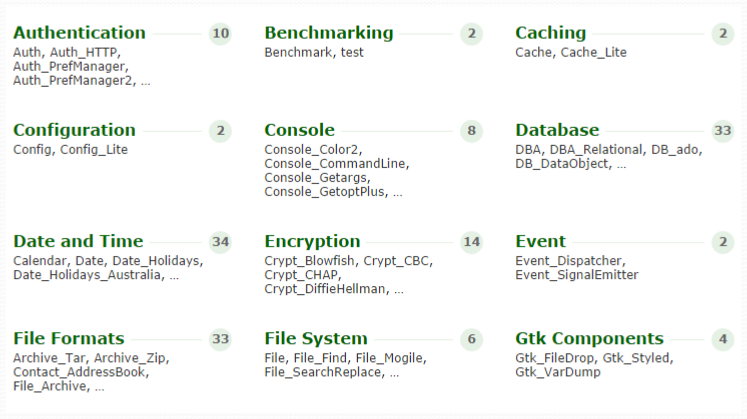
- PEAR is a framework and distribution system for reusable PHP components. You can find help using PEAR packages in the online manual and the FAQ.

- PEAR is short for "PHP Extension and Application Repository" and is pronounced just like the fruit. The purpose of PEAR is to provide:
  - A structured library of open-source code for PHP users
  - A system for code distribution and package maintenance
  - A standard style for code written in PHP,
  - The PHP Extension Community Library (PECL),
  - A web site, mailing lists and download mirrors to support the PHP/PEAR community

- PEAR is a community-driven project governed by its developers. PEAR's governing bodies are subdivided into the PEAR Group, Collectives, and a President. PEAR's constitution (adopted in March 2007) defining these groups is documented.
- The PEAR project was founded in 1999 by Stig S. Bakken and a large community has joined the project.

Pear Components (from http://pear.php.net/)

Packages (598)



Authentication 10
Auth, Auth_HTTP, Auth_PrefManager, Auth_PrefManager2, ...

Benchmarking 2
Benchmark, test

Caching 2
Cache, Cache_Lite

Configuration 2
Config, Config_Lite

Console 8
Console_Color2, Console_CommandLine, Console_Getargs, Console_GetoptPlus, ...

Database 33
DBA, DBA_Relational, DB_ado, DB_DataObject, ...

Date and Time 34
Calendar, Date, Date_Holidays, Date_Holidays_Australia, ...

Encryption 14
Crypt_Blowfish, Crypt_CBC, Crypt_CHAP, Crypt_DiffieHellman, ...

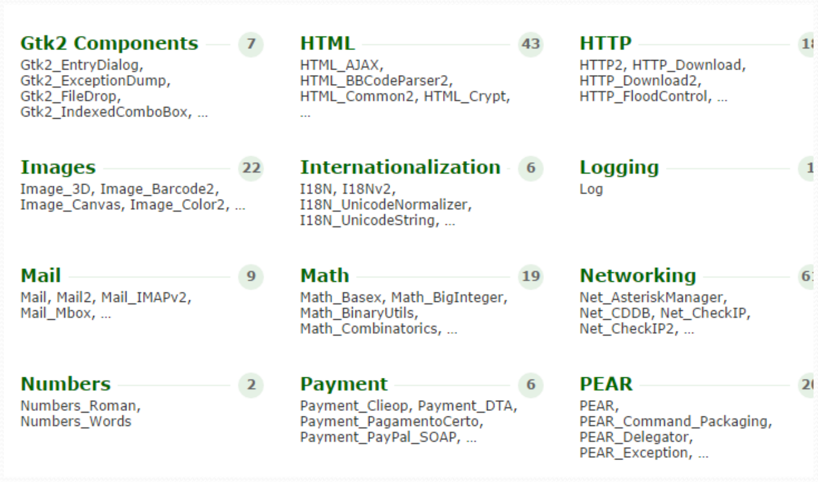
Event 2
Event_Dispatcher, Event_SignalEmitter

File Formats 33
Archive_Tar, Archive_Zip, Contact_AddressBook, File_Archive, ...

File System 6
File, File_Find, File_Mogile, File_SearchReplace, ...

Gtk Components 4
Gtk_FileDrop, Gtk_Styled, Gtk_VarDump

Pear Components (from http://pear.php.net/)



Gtk2 Components 7
Gtk2_EntryDialog, Gtk2_ExceptionDump, Gtk2_FileDrop, Gtk2_IndexedComboBox, ...

HTML 43
HTML_AJAX, HTML_BBCodeParser2, HTML_Common2, HTML_Crypt, ...

HTTP 1
HTTP2, HTTP_Download, HTTP_Download2, HTTP_FloodControl, ...

Images 22
Image_3D, Image_Barcode2, Image_Canvas, Image_Color2, ...

Internationalization 6
I18N, I18Nv2, I18N_UnicodeNormalizer, I18N_UnicodeString, ...

Logging 1
Log

Mail 9
Mail, Mail2, Mail_IMAPv2, Mail_Mbox, ...

Math 19
Math_Basex, Math_BigInteger, Math_BinaryUtils, Math_Combinatorics, ...

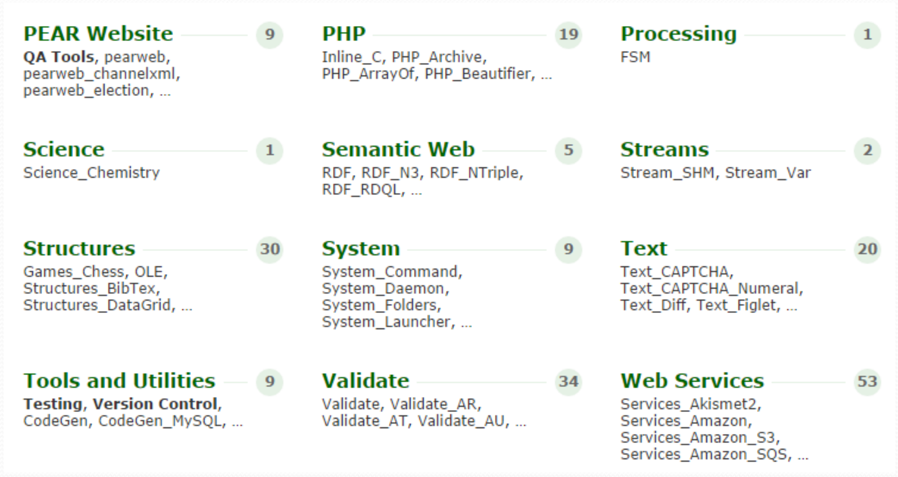
Networking 6
Net_AsteriskManager, Net_CDDB, Net_CheckIP, Net_CheckIP2, ...

Numbers 2
Numbers_Roman, Numbers_Words

Payment 6
Payment_Clieop, Payment_DTA, Payment_PagamentoCerto, Payment_PayPal_SOAP, ...

PEAR 2
PEAR, PEAR_Command_Packaging, PEAR_Delegator, PEAR_Exception, ...

Pear Components (from http://pear.php.net/)



PEAR Website 9
QA Tools, pearweb, pearweb_channelxml, pearweb_election, ...

PHP 19
Inline_C, PHP_Archive, PHP_ArrayOf, PHP_Beautifier, ...

Processing 1
FSM

Science 1
Science_Chemistry

Semantic Web 5
RDF, RDF_N3, RDF_NTriple, RDF_RDQL, ...

Streams 2
Stream_SHM, Stream_Var

Structures 30
Games_Chess, OLE, Structures_BibTex, Structures_DataGrid, ...

System 9
System_Command, System_Daemon, System_Folders, System_Launcher, ...

Text 20
Text_CAPTCHA, Text_CAPTCHA_Numeral, Text_Diff, Text_Figlet, ...

Tools and Utilities 9
Testing, **Version Control**, CodeGen, CodeGen_MySQL, ...

Validate 34
Validate, Validate_AR, Validate_AT, Validate_AU, ...

Web Services 53
Services_Akismet2, Services_Amazon, Services_Amazon_S3, Services_Amazon_SQS, ...

Example use of PEAR:
Using a Generic Database Interface: PEAR DB

- PEAR DB abstraction layer.
- This is one of the core components of PEAR and probably the most widely used of all the PEAR components.
- DB component is usually installed with PEAR
- If not, refer to the “PEAR Installation” section in Appendix A,“Installing PHP and MySQL.”
  - http://pear.php.net/manual/en/installation.php
-

Using a Generic Database Interface: PEAR DB

- If you want to use a database that doesn’t have a specific library available in PHP, you can use the generic ODBC functions. ODBC, which stands for *Open Database Connectivity,*
- In addition to the libraries that come with PHP, available database abstraction classes such as PEAR::DB allow you to use the same function names for each different type of database.
- PEAR MDB2 is a merge of the PEAR DB and Metabase php database abstraction layers.
  It provides a common API for all supported RDBMS. The main difference to most other DB abstraction packages is that MDB2 goes much further to ensure portability.
  http://pear.php.net/package/MDB2

Pear DB Example

```
// set up for using PEAR DB
require_once('DB.php');
$user = 'bookorama';  $pass = 'bookorama123';  $host = 'localhost';
   $db_name = 'books';
// set up universal connection string or DSN
$dsn = "mysqli://$user:$pass@$host/$db_name";
// connect to database
$db = &DB::connect($dsn) ;
//:: for accessing static variables and methods of a class -
//Address-of operator (&)

// check if connection worked
if (DB::isError($db))
{
echo $db->getMessage();
exit;
}
```

Pear DB Example(ctd..)

```
// perform query
$query = "select * from books where ".$searchtype." like
   '%".$searchterm."%'";
$result = $db->query($query);
// check that result was ok
if (DB::isError($result))
{
echo $db->getMessage();
}
```

```
// get number of returned rows
$num_results = $result->numRows();
// display each returned row
for ($i=0; $i <$num_results; $i++)
{
$row = $result->fetchRow(DB_FETCHMODE_ASSOC);
echo '<p><strong>'.($i+1).'. Title: ';
echo (stripslashes($row['title']));
echo '</strong><br />Author: ';
echo stripslashes($row['author']);
echo '<br />ISBN: ';
echo stripslashes($row['isbn']);
echo '<br />Price: ';
echo stripslashes($row['price']);
echo '</p>';
}
// disconnect from database
$db->disconnect();
```

The htmlspecialchars() function converts some predefined characters to HTML entities.
The predefined characters are:
& (ampersand) becomes &amp;
" (double quote) becomes &quot;
' (single quote) becomes &#039;
< (less than) becomes &lt;
> (greater than) becomes &gt;

Difference in this script?
DB connection

```
$db = DB::connect($dsn);
```

This function accepts a universal connection string that contains all the parameters necessary to connect to the database. You can see this if you look at the format of the connection string:

```
$dsn = "mysqli://$user:$pass@$host/$db_name";
```

After this, you check to see whether the connection was unsuccessful using the isError() method and, if so, print the error message and exit:

```
if (DB::isError($db))
{
echo $db->getMessage();
exit;
}
```

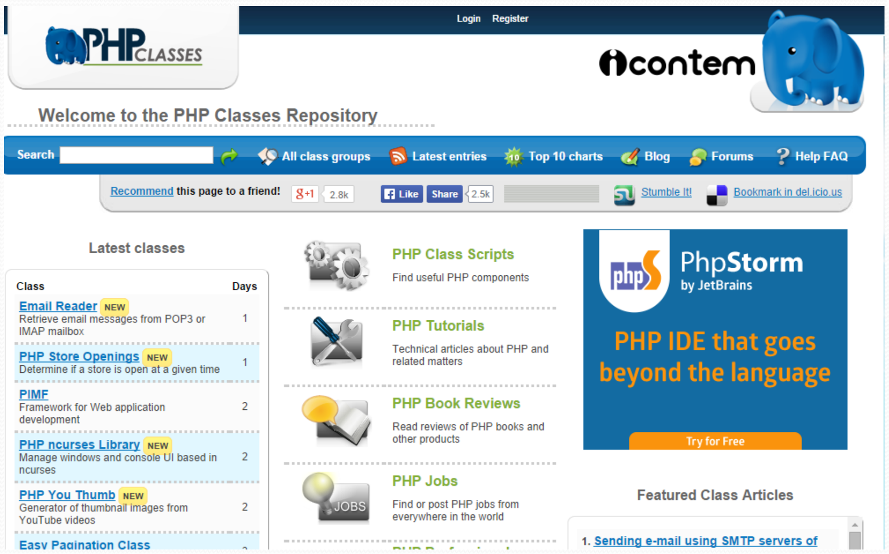
- query DB:
  - $result = $db->query($query);
- check the number of rows returned:
  - $num_results = $result->numRows();
- You retrieve each row as follows:
  - $row = $result->fetchRow(DB_FETCHMODE_ASSOC);
- The generic method fetchRow() can fetch a row in many different formats; the parameter DB_FETCHMODE_ASSOC tells it that you would like the row returned as an associative array.
- After outputting the returned rows, you finish by closing the database connection:
- $db->disconnect();

- The advantages of using pear DB is that you need to remember only one set of database functions and that the code will require minimal changes if you decide to change the database software.

ODBC Connectivity

- See www.php.net for setting up ODBC connection.
- Note : DSN?
  - Short for ***Data Source Name***. Data Source Name provides connectivity to a database through an ODBC driver. The DSN contains database name, directory, database driver, UserID, password, and other information. Once you create a DSN for a particular database, you can use the DSN in an application to call information from the database.

http://www.phpclasses.org/


Login Register

Welcome to the PHP Classes Repository

Search | All class groups | Latest entries | Top 10 charts | Blog | Forums | Help FAQ

Recommend this page to a friend! 2.8k Like Share 2.5k Stumble It! Bookmark in del.icio.us

Latest classes

Class	Days
Email Reader NEW Retrieve email messages from POP3 or IMAP mailbox	1
PHP Store Openings NEW Determine if a store is open at a given time	1
PIMF Framework for Web application development	2
PHP ncurses Library NEW Manage windows and console UI based in ncurses	2
PHP You Thumb NEW Generator of thumbnail images from YouTube videos	2
Easy Pagination Class	

PHP Class Scripts
Find useful PHP components

PHP Tutorials
Technical articles about PHP and related matters

PHP Book Reviews
Read reviews of PHP books and other products

PHP Jobs
Find or post PHP jobs from everywhere in the world

PhpStorm by JetBrains
PHP IDE that goes beyond the language
Try for Free

Featured Class Articles

1. Sending e-mail using SMTP servers of