

IT360: Applied Database Systems

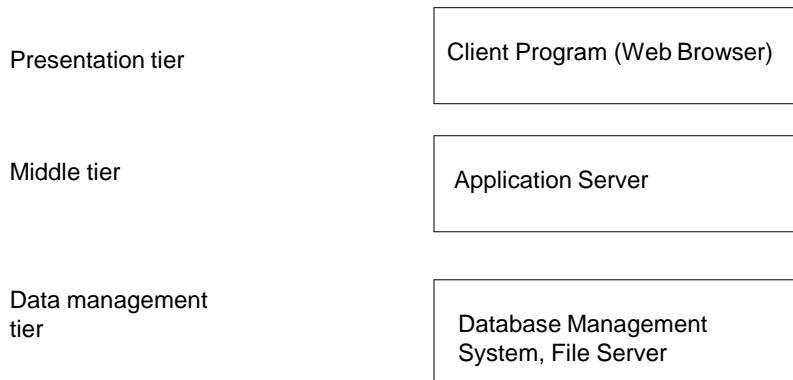
Introduction to PHP

Chapter 1 and Chapter 6 in "PHP and MySQL
Web Development"

Course Topics

- SQL
- Relational model
- Normalization
- Database design
- **PHP**
- MySQL
- Database administration
- Transaction Processing
- Data Storage and Indexing

The Three-Tier Architecture



Example 1: Airline reservations

Build a system for making airline reservations

- Database System
- Application Server
- Client Program

Technologies

Client Program (Web Browser)	HTML, Javascript, XSLT
Application Server	C++, Cookies, XML, XPath, web services, Perl, PHP
Database Management System	SQL, Triggers, Stored Procedures

Web Applications

- Need to choose:
 - Operating system
 - Web server software
 - Database Management System
 - Programming or scripting language

PHP

- PHP: PHP Hypertext Preprocessor
- Server-side scripting language
- PHP pages require a web server with PHP support
- Competitors:

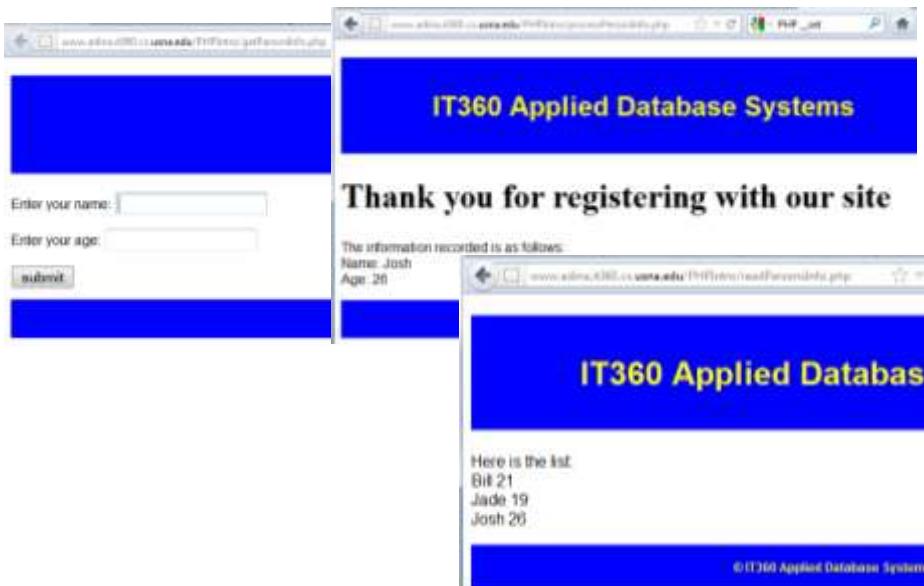
PHP Strengths

- High performance
- Interface to many different database systems
- Built-in libraries
- Ease of learning and use
- Object-oriented support
- Portability
- Open source
- Free
- Availability of support

PHP References

- Online references
 - <http://www.php.net>
- Online Tutorial
 - <http://www.w3schools.com/php/default.asp>
- *PHP and MySQL Web Development* by Luke Welling and Laura Thomson
- IT350 textbook: Internet & WWW How To Program by Deitel, Deitel, and Goldberg

CGI – What does it all look like?



IT350 - CGI Script Basics

- Common Gateway Interface (CGI)
 - “Common”: Not specific to any operating system or language
- Output file generated at runtime:
 1. When a program executed as a CGI script, “standard output” is redirected to client Web server
 2. Web server then redirects output to client's browser

IT350 - How can CGI get data from user?

Technique #1: Forms

- User enters data via a form, submits
- Form directs results to a CGI program
- Script receives data in one of two ways:
 1. Method = “GET”
 2. Method = “POST”

Use language-specific method to get these inside CGI program

Technique #2: URL with parameters

```
<a href="http://www.cs.usna.edu/calendar/view.php?events=seminars">  
Seminars </a>
```

Form Processing

PHP Overview

- PHP tags <?php ?>
 - Mixed with HTML tags
 - File extension .php
- Statements
 - Separated by semicolon
 - if..else.., while, do, for, switch
- Variables
 - \$varname
 - Type determined by content; variables not declared; case sensitive
- Strings
 - Single quotes – literal string
 - Double quotes – interpolated string (variables are replaced with their value)
- Accessing form variables
 - `$_POST['age'], $_GET['age'], $_REQUEST['age']`

PHP Overview

- PHP objects
 - Java-like inheritance
 - public, private, or protected attributes and methods
 - `__construct()`, `__destruct()`,
 - `__set()`, `__get()`
- PHP functions
 - `function myFunction($param1, $param2){...}`
- Files
 - `resource fopen(string $fileName, string $mode)`
 - `int fwrite(resource $handle, string $someText)`
 - `int fclose(resource $handle)`
 - `string fgets(resource $handle)`
 - `boolean feof(resource $handle)`

How everything works: Step 1: Input Form – form.html

```
<?xml version = "1.0" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html><head><title>IT360 PHP test page</title></head>
<body>

<form action="processPersonInfo.php" method="post">

<p><label>Enter your name: <input type="text" name="name"/></label></p>

<p><label>Enter your age: <input type="text" name="age" /></label></p>

<p><input type="submit" name = "submit" value = "Submit"/></p>

</form>
</body> </html>
```

```

<?php
class Page{

    //attributes
    public $content;
    private $title;

    private $header = "<?xml version = '1.0' ?>
        <!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.1//EN'
        'http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd'>";

    //constructor
    public function __construct($title){
        $this->title = $title;
    }

    //set public attributes
    public function __set($name, $value){
        $this->$name = $value;
    }

    //display page
    public function display(){
        echo $this->header;
        echo "<head><title> $this->title </title></head>";
        echo "<body>";
        echo $this->content;
        echo "</body></html>";
    }
} //end class definition
?>

```

Step 1 version 2 – getPersonInfo.php

```

<?php

//bring in the class definitions,
//so we can use them
require('page.inc.php');

//create a new page object
$page = new Page("Input person");

//set the content: in needs to be a form
$page->content = '<form action = "processPersonInfo.php" method = "post">'.
    '<p><label>Enter your name: <input type="text" name="name"/></label></p>' .
    '<p><label>Enter your age: <input type="text" name="age"/></label></p>' .
    '<input type = "submit" value = "submit">';

//display the page
$page->display();

?>

```

```
<?php
```

person.inc.php – part 1

```
/* define a class Person with name and age */
class Person{

    private $name;
    private $age;

    //constructor
    public function __construct(){}

    //default set function invoked when the private fields are set
    //this is a good place to do sanity/security checks
    public function __set($varName, $varValue)
    {
        $varValue = trim($varValue);
        $varValue = strip_tags($varValue);
        if (!get_magic_quotes_gpc()){
            $varValue = addslashes($varValue);
        }
        $this->$varName = $varValue;
    }

    //default get function - nothing special for now
    public function __get($varName)
    {
        return $this->$varName;
    }
}
```

person.inc.php – part 2

```
//return a string that contains the HTML code to get data for a person
public static function getPersonAttributesAsHTMLInput()
{
    $myString = '<p><label>Enter your name: <input type="text" name="name"/></label></p>
    <p><label>Enter your age: <input type="text" name="age" /></label></p>';
    return $myString;
}

//process the person info to insert to file and display confirmation
public function processPerson(){
    //write this person to the default file
    $success = $this->insertToFile();

    //return a confirmation message
    if ($success){
        $confirmation = '<h1>Thank you for registering with our site</h1>.
                        '<p>The information recorded is as follows: <br />'.
                        "Name: $this->name <br /> Age: $this->age </p>';
    }
    else{
        $confirmation = '<h1>Error: we had problems with your registration (probably some file error - permissions??).
                        Please try again.</h1>';
    }

    return $confirmation;
}
```

person.inc.php – part 3

```
/* save the content to a specified file
or "persons.txt" if nothing is specified */
private function insertToFile($fileName="persons.txt")
{
    $fp = @fopen($fileName, 'a');
    if (!$fp){
        return false;
    }
    else{
        $text = "$this->name\t$this->age\n";
        fwrite($fp, $text);
        fclose($fp);
        return true;
    }
}
```

person.inc.php – part 4

```
/*read all info from file and return it in some nice format */
public static function getAllPersonsInfo($fileName = "persons.txt"){

    //read the data from file and construct the content
    $fp = @fopen($fileName, 'r');
    //check for errors
    if (!$fp){
        $content = "<p>ERROR! Could not open file $fileName for
reading.</p>";
    }
    //if everything OK, read the file
    else{
        $content= '<p>Here is the list: <br />';
        //read one line
        $line = fgets($fp);
        while( !feof($fp) ){
            //process the line
            $content .= $line . '<br />';

            //read next line
            $line = fgets($fp);
        }
        $content .= '</p>';
        //close the file
        fclose($fp);
    }

    return $content;
}
?>
```

Step 2 processPersonInfo.php

```
<?php

//bring in the class definitions,
require('page.inc.php'); require('person.inc.php');

//get the params sent by the form
$name = $_POST['name']; $age = $_POST['age'];

//create a new page object
$page = new Page("Registration confirmation");

//check that input params ok
if (empty($name) || empty($age)){
    $page->content = '<p> Name or age not entered!! Try again</p>';
    $page->display();
    exit;
}

//create a new person with these properties
$dummy = new Person();
$dummy->name = $name;
$dummy->age = $age;

//set the content: the confirmation message returned by the processPerson
method for this person; all work is done inside processPerson
$page->content = $dummy->processPerson();

//display the page
$page->display(); ?>
```

Step 3: readPersonsInfo.php

```
<?php

//bring in the class definitions,
//so we can use them
require('page.inc.php');
require('person.inc.php');

//get the input params (from URL)
if (isset($_GET['filename'])){
    $fileName = $_GET['filename'];
}
else{
    $fileName = "persons.txt";
}

//create a new page object
$page = new Page("Persons list");

//set the content: the confirmation message returned by the
processPerson method for this person
//all work is done inside getAllPersonsInfo
$page->content = Person::getAllPersonsInfo($fileName);

//display the page
$page->display();
?>
```

Class Exercise

- Modify getPersonInfo.php to use the `getPersonAttributesAsHTMLInput` method in Person class

Class Exercise

- Create a PHP script that accepts a parameter called *number* from the address bar and prints back the English word for the decimal value (assume number is between 1 and 5). You should create first a file that looks like this, and read from it:

One
Two
Three
Four
Five

PHP Summary

- PHP tags <?php ?>
 - Mixed with HTML tags
 - File extension .php
- Statements
 - Separated by semicolon
 - if..else..., while, do, for, switch
- Variables
 - \$varname
 - Type determined by content; variables not declared; case sensitive
- Strings
 - Single quotes – literal string
 - Double quotes – interpolated string (variables are replaced with their value)
- Accessing form variables
 - `$_POST['age'], $_GET['age']` (if method is GET), `$_REQUEST['age']`

PHP Summary

- PHP objects
 - Java-like inheritance
 - public, private, or protected attributes and methods
 - `__construct()`, `__destruct()`,
 - `__set()`, `__get()`
- PHP functions
 - `function myFunction($param1, $param2){...}`
- Files
 - `resource fopen(string $fileName, string $mode)`
 - `int fwrite(resource $handle, string $someText)`
 - `int fclose(resource $handle)`
 - `string fgets(resource $handle)`
 - `boolean feof(resource $handle)`