ITMD 462/562
Web Site Application Development

Lecture 2

Fall 2015 – September 2, 2015

Tonight's Agenda

- Servers and Responses
- Setting up your local development environment
- Introduction to the PHP Language

Servers and the Request-Response Model

Servers

- Why do we need a server?
- Server hardware/OS is a fixed location resource that provides an operating environment for applications to run which listen for network communications and respond.
- A Server application/container is a specific software application that is running on the server OS that listens and responds to specific types of requests.
- Servers run program code logic to form the response

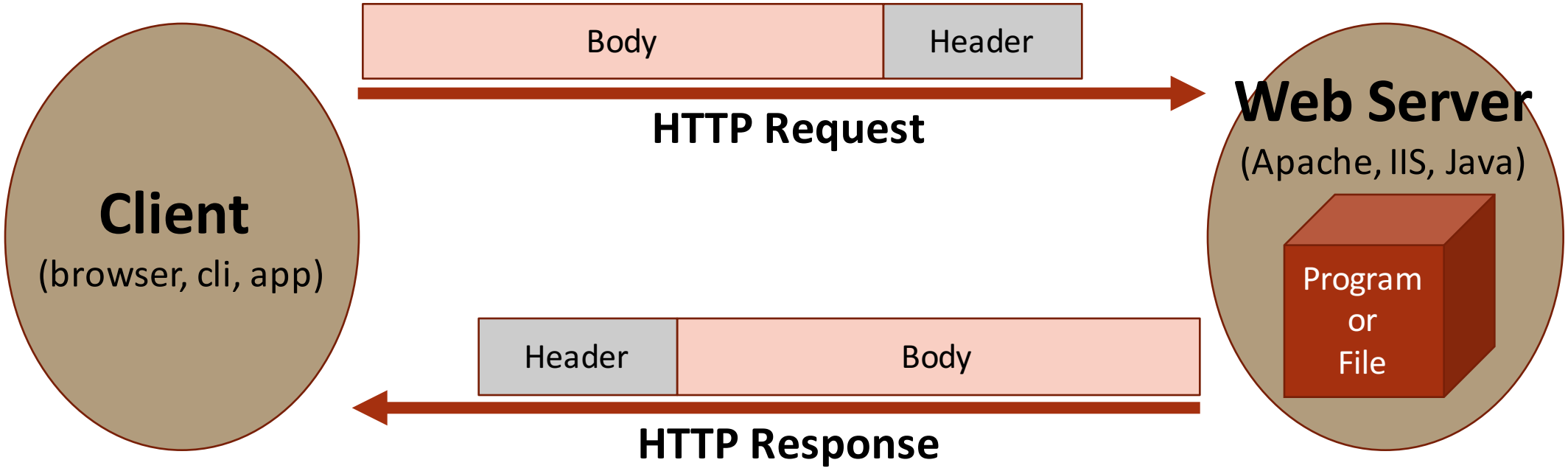
Servers/Applications

- There are several types of servers/containers
- HTTP/Web Server Container
  - Apache
  - IIS
  - nginx
- Java Application Server Container
  - Tomcat
  - Jboss
  - Glassfish
- Many others like FTP, SSH, GAMES

Web Servers

- Web servers listen for HTTP or HTTPS protocol requests on a specific port.
- What is a port?
- HTTP default port is 80
- HTTPS default port is 443
- Web servers job is to listen on the port for the HTTP request and route the request to the folder or file on the file system or another type of resource.

HTTP Request-Response Model



Server Logic

- If the requested resource points to a normal file the web server will then respond with that file and MIME type
- If the requested resource points to a program then the web server routes the request to the program for processing then responds to the client with the results of the program output.
- We will be writing our logic in PHP
- What kinds of things can we write using that logic?
  - Blogs, Social Networks, Forums, Cloud Services, Business Portals, Shopping Carts, Games, and more

Client Logic

- What is a client?
  - Typically a web browser but can be anything that sends an HTTP request. Browsers, apps, command line tools
- What does our clients need to do?
  - Using a URL compose an HTTP GET request
  - Compose a HTTP POST request using form data
- Web Services and REST APIs
  - Using a resource URI send a PUT, DELETE, POST, GET request get data response

Server Programs and Code

- Web server (Apache) receives the request and if it is for a program based resource sends it to the program engine for processing.
- Our program creates the response content and sends it back to the web server and then the web server sends the response to the client.

Web Server

Receive Request
Decipher Request
Access Resources / Run Logic
Create Response
Send Response

Our Program

Local Server

Web Server and PHP Interpreter is required

- The browser can not interpret PHP
- We need a web server to receive the request and hand off the request to our PHP interpreter and then get the response back from the PHP engine and send it to the client
- We need to run a local PHP enabled web server for development

LAMP, WAMP, MAMP, XAMPP

- Typical development environment is a LAMP stack
  - Linux, Apache, MySQL, PHP
- Prepackaged LAMP stacks for all major operating systems
  - WAMP – Windows
  - MAMP – Mac, Windows
  - XAMPP – Windows, Mac, Linux
- [OS] Apache MySQL PHP

PHP Built in Server

- Recent versions (>5.4) of PHP have a built in server.
- Not always preconfigured for your operating system especially database drivers

```
php -S localhost:8080
```

Introduction to PHP

PHP is a programming language

- PHP is a full programming language, not a framework or application
- PHP is an interpreted language at runtime (no complier)
- Syntax similar to C based languages
- Has all your standard programming language data types and control structures
- PHP is very popular and easy to learn and use in the context of web programming
- Can also be use outside of web programming like command line apps but it is much less common.

PHP Tags

- The file should be a plain text file and needs to end in a .php extension to tell the web server that it is PHP and needs to be processed by the PHP engine.
- Any PHP code needs to be inside the PHP tags
- The PHP tags tell the interpreter what code needs to be evaluated.

<?php ?>

PHP Info Function

- This is a built in function that can check that PHP is working in the server correctly.
- Displays a web page showing all the PHP configuration

```
<?php
  phpinfo();
?>
```

PHP Variables

- Variables are a way to store and access data in memory in your application.
- All variables in PHP are denoted by a leading dollar sign ($)
- Variables are assigned a value with the = operator. Variable on left, expression on the right.
- Do not need to be declared first.
- Do not have intrinsic types.
- Value is equal to its most recent assignment

PHP Variables

- Cannot start with a number. Need to start with A-Za-z_ then number can follow
- No Spaces allowed in name. Use camelCase or underscores

```
<?php
$myVariable = 7;
myBadVariable = 8;
$34myVariable = 123;
$_4myNewVariable = 'This is fine';
$_4myNewVariable = 7;
$another_variable_name = array(5,6,7);
?>
```

PHP Constants

- An identifier for a simple value.
- Can not change during script execution
- Does not start with dollar sign ($). By convention should be all upper case with underscores for spaces.
- Typical variable naming conventions apply
- One useful built-in constant is PHP_EOL

```
Define("FOO", "somevalue");
```

PHP Comments

- Use comments to annotate your work for others and yourself but try not to overuse when the code is clear.

```
// Single line comment


/*
Multi line comment
*/
```

PHP Data Types

- Here is the listing of PHP data types from the php.net manual

http://php.net/manual/en/language.types.php

PHP Numbers

```
$myVar = 8;
$myFloatingPointNum = 3.1415;
$myScientificNum = 4E23;
$myVar++;
$myVar--;
```

PHP Boolean

- Used to hold a true/false value
- No Quotes! Case-insensitive

```
$myVal = true;
$myVal = false;
```

See PHP docs for examples of what coverts to true/false

PHP Strings

- A series of characters.
- Single Quoted vs Double Quoted
- Special characters or same quotes need to be escaped with \

```
$myVal = 'hello’;
$myVal = "hello”;
$myVal = "hello $myName”;
$myVal = 'hello $myName’;
$myVal = "hello \"World\" $myName";
```

String Manipulation

- PHP has many string manipulation functions built-in
- http://php.net/manual/en/ref.strings.php
- The web has a lot of text and HTML is all text based so it is essential to be able to manipulate strings.

```
print, echo
printf
sprintf
htmlspecialchars
rtrim, ltrim
nl2br
strtolower, strtoupper
substr
```

PHP Arrays

- An array is an ordered list or map
- Simple arrays are indexed by a number
- Associative arrays are similar to a hashmap and indexed by a string
- PHP arrays can contain both types of keys

http://php.net/manual/en/language.types.array.php

Indexed Array

- Using numbers as keys

```
$var = array('a', 'b', 'c');

echo $var[0]      // prints out a

$var = array(
0 => 'a',
1 => 'b'
)
```

Associative Arrays

- Similar to a Java HashMap
- Uses string keys

```
$foo = array('bar' => 'baz', 'rc' => 'car' );
$value = $foo['bar'];
```

Advanced Arrays

- Arrays can contain other arrays
- Multidimensional arrays
- Can be very deeply nested and pretty fast in PHP
- `print_r()` function can be helpful in visualizing a deeply nested array.

PHP Operators

- Operators take one or more values and yields a different value
- See the PHP documentation for all the operators and precedence

http://php.net/manual/en/language.operators.php

PHP Math

- PHP has a built-in math function library
- http://php.net/manual/en/ref.math.php
- Working with numeric data is very important and easy.
- Basic Math uses the Math operators, complex math uses these functions.

PHP Control Structures

- We often need to branch code depending on a condition or repeating code
- Language for branching might sound like:
  - Do this if that happens
  - If the data is this value – do this otherwise do that
  - Do this 5 times.
  - Do this for every item in this list.
- Control structures are detailed in the PHP docs

http://php.net/manual/en/language.control-structures.php

Assignments

- Get you local environment installed and working.
- Test the local environment with a phpinfo file and also some basic php to make sure everything is working.
- Possible discussion board posting will be coming. I will email if I want something posted.
- Assignment 1 will be posted soon

Reading

- Read over PHP documentation regarding topics covered tonight. Don't need to read all the pages for each function. Just the high level pages.
- Take a look at the additional resources I posted in blackboard so you know what they contain.