

CS25010: More About PHP

Jonathan Francis Roscoe <jjr6@aber.ac.uk>
<http://users.aber.ac.uk/jjr6>
Department of Computer Science, Aberystwyth University

November 29, 2011

Introduction

Web programming is a vast topic. In today's lecture I introduced you to some of the more advanced, yet interesting aspects of PHP. This handout is a review of what we discussed in the lecture and I hope will provide you with useful information and links to develop your knowledge beyond what you've already learnt in CS25010.

Some PHP tips

With dynamic weak typing, PHP is a laid back language allowing functional code to be written fast.

- Use a strong editor with supporting tools - Notepad++, Eclipse (with plugin), vim, Netbeans
- Use server-side includes and mix PHP and HTML as little as possible
- Test your code (yes - unit testing and acceptance testing is possible)
- Debug your code
- Take security seriously
- Frameworks can significantly improve the speed of your develop, as well as providing features for efficiency, security and reliability.

PHP 5 and Object-Oriented

Introduced in PHP 5, there is a lot in common with Java's syntax, rules and conventions.

- Interfaces and abstract classes
- Magic Methods (e.g. `__toString()`)
- Exception classes and `try..catch`

With simple yet powerful polymorphism:

- Instantiation from String
- `class_exists()`, `method_exists()` and other tricks

PHP 5 introduced a few other significant features:

- Type Hinting in methods
- Reflection
- Method overloading (quite different from Java)

A Very Simple OO Example

```
interface Animal {
    function speak();
}

class Cow implements Animal{
    function speak(){
        print "Moooooo!";
    }
}
```

```

    }
}
$classname = "Cow";
$a = new $classname;
$a -> speak ();

```

Ajax

Advanced Javascript can be tremendous fun when combined with server-side facilities. Ajax is a generic term from asynchronous data transfer between a web page and the server. It can be used to build sophisticated applications and has become almost ubiquitous on the World Wide Web.

To get you started, I've provided a very simple example. Googling for information on XMLHttpRequest will turn up many more examples, as well as full documentation on its use.

hello.php:

```
<?php echo "Hello World!"; ?>
```

index.php:

```

<html><head><title>Ajax Example</title>
<script type="text/javascript">
    var request = XMLHttpRequest();

    function makeRequest(){//4 - response to handle, 0 = no object
    if (request.readyState == 4 || request.readyState == 0)
    {
        request.open("GET", "hello.php", true);
        request.onreadystatechange = handleResult; //set the callback
        request.send();
    }
}
function handleResult(){
    if (request.readyState == 4) //we got a response
        document.getElementById('server_says').innerHTML
            = request.responseText;
}
</script>
</head><body>
<input type="button" value="Go!" onClick="makeRequest();" />
<p>The server says <span id="server_says"></span>
</body></html>

```

Though beware of XMLHttpRequest with older browsers (notably IE < 7).

Debugging

It can sometimes be difficult to understand how and why PHP code is behaving the way it is, but there are some useful tools for debugging server-side applications

- Static Analysis - lint, PHP Codesniffer, IDE features
- Dynamic Analysis - xdebug, Valgrind

PHP has error reporting of its own (this can be configured in php.ini), and some frameworks may provide debugging features.

Browser Extensions

Web development can be a deviation from your familiar development techniques, fortunately various extensions can be invaluable in debugging the information that the client has, particularly when working with Ajax.

- Web Developer Toolbar
 - Firefox

- Chrome
- Firebug
 - Firefox
 - Chrome, Internet Explorer, etc (Lite)
- IE WebDeveloper

Testing

Unit tests

- Test each module of code
- Core to test driven development methodology
- PHP-Unit is the PHP equivalent of JUnit, a tool you probably already know. You can use it to run subsections of your code, and test expected output.

Functional tests

- Tests of end user experience
- Selenium is a powerful tool that allows automatic execution of your webpages. This makes it easy to fully automate the testing of your entire site.

Frameworks

Frameworks can be compared to more specific software such as blogs (e.g. Wordpress) or Content Management Systems (eg. Drupal). Typically, they will make stricter use of OO and be based on the MVC framework. They're general aims are:

- Faster development
- Reduced overhead
- Reusability (Don't Repeat Yourself)

Frameworks provide support for a number of common needs, including:

- Session management
- Database interaction (usually Object Relational Mapping)
- Ajax
- Testing
- Deployment
- Templating

Popular Frameworks

- Zend Framework
- Symfony
- Codeigniter
- Rails
- Zope

Security

General Rules for Securing PHP

PHP is often used to create public facing websites on the Internet, which typically include privileged areas. There are a few general security issues to be aware of:

- Ensure user is authenticated & authorised as appropriate
- Input should be validated & sanitised
- Avoid careless debugging/error reporting (configure php.ini not to print error messages on a live system)
- Development data (test files, svn metadata) should be removed
- Server should be appropriately configured
- Utilise security software that can isolate holes in your application..

Sanitising Input

You've probably seen or implemented some form of client-side security - such as a Javascript popup if you haven't don't complete a form properly. These mainly exist to be quick and look pretty, they are not at all secure. So you must **sanitise input on the server-side**.

```
$username = filter_var(FILTER_SANITIZE_STRING, $_POST['username']);
```

Regular expressions can be used to go beyond simply removing harmful characters and can validate some requirements:

```
if(preg_match("/^http/", $_POST['url']))
    echo "Got a URL!";
else
    echo "URL is no good.;"
```

Fortunately, if you do make some horrible mistake, there are tools to protect you. Like Pixy, a server-side code scanner used many shared webhosts.

Security Testing Tools

There are plenty of tools available to help you test your site, popular ones include:

- <http://code.google.com/p/skipfish> - Skipfish is an open source web application security scanner from Google
- Metasploit - a generic vulnerability scanner modules exist to test off-the-shelf (wordpress, drupal, etc) software <http://www.metasploit.com/modules/>
- Nikto2 - another open-source tool
- <http://evuln.com/tools/php-security/> - a free static code scanner for PHP
- <http://loadimpact.com/> - Remote load testing (Are your database queries efficient enough?!)

Environments

Bugs in new code will have a serious impact on usability and security. A common technique for releasing code is to develop on isolated systems, then gradually deploy them in a limited or beta form, before going live.

- Development
Usually the programmer's local machine. May use specialist data sets. Incomplete/untested functionality. Full debugging and error output.
- Testing
A local network server. Provides common test data for all developers. All code should have been partially tested and obtained from development branch of version control. Full debugging and error output.
- Staging/Pre-Production ("Bleeding Edge")

A live server with limited accessibility. May be available to beta users. Should be treated as the live system. All code should have been fully tested and taken from stable branch of version control.

- Production

The live server. Code on this server should not be edited directly. System should be updated through some form of scheduled deployment procedure. Debugging information printed here is a security risk and unpleasant to end users - errors should be caught and politely reported.

Setting Up Your Own System

If you're serious about developing PHP, I suggest you configure your own Linux Apache MySQL PHP (LAMP) server. This is easy to do on most distributions, an example for Ubuntu can be found here: http://www.howtoforge.com/ubuntu_lamp_for_newbies

You can still develop and test PHP code on Windows, XAMPP is a package that can configure various web servers for you: <http://www.apachefriends.org/en/xampp.html> This package is targeted at all major operating systems.

Developing on your own system, be it a VPS, virtual machine or your regular desktop will give you a wealth of experience in the configuration of servers providing PHP. If you're excited (or perhaps bored?..) enough you can compile PHP with debug symbols for all kinds of adventures.

PHP is not just for the web

If you really like PHP you could use it in every facet of your computing work. To start with, try writing your shell scripts in PHP using the command line version (php5-cli on debian systems). Take it a step forward with <http://gtk.php.net/> for creating GUIs.

And other libraries

Plenty more third-party libraries exists for you to play with, including:

- <http://php.net/manual/en/book.image.php> - gdLibrary for creating images
- <http://pchart.sourceforge.net/> - pchart library
- <http://teethgrinder.co.uk/open-flash-chart/> - flash chart creator
- <http://www.php.net/manual/en/intro.pdf.php> - PDFlib can let you create PDFs of your webpages on the fly

Useful PHP Tidbits

- Server configuration information: `phpinfo()`
- Server Side Inclusion: `include()`, `require()`, `include_once()`, `require_once()`
- Validation/Sanitisation functions: `preg_match()`, `strip_tags()`, `filter_var()`, `mysql_real_escape_string()/pg_escape_string()`
- HTML string output: `htmlspecialchars()`
- Special predefined variables: `$_SERVER[]`, `$_SESSION[]`, `$_ENV[]`
- Find files on the system - `glob()`
- PEAR is a framework for managing PHP extensions, available from repositories such as PECL
- Apache module for nice URLs: `mod_rewrite`
- Run shell commands: `exec("ls -lah")`
- Many configuration options in `php.ini` - if you manage the server
- Doxygen (generic Javadoc) works well with PHP

Related Reading

- <http://php.net/> - is fantastic documentation with almost every little bit including user contributed examples
- "Billions of Hits: Scaling Twitter" presentation
- <http://www.unmaskparasites.com/> - tool for website vulnerabilities
- <https://www.owasp.org/> Lots of security information including an injections cheat sheet
- <http://www.symfony-project.org/book/> - the Symfony book, lots of good MVC information
- <http://www.php.net/>
- <http://www.phpframeworks.com/> - comparison of frameworks
- <http://browsershots.org/> - screenshots of your website in dozens of configurations (platform, browser, resolution)