6.470 Web Programming Competition
IAP 2013
PHP Workshop

Prior to the workshop

In general, you should try to become as fluent as possible in reading and writing basic PHP code before the workshop. Try to also understand how PHP fits into the big picture via the explanations below, the lectures posted on the 6.470 website, and tutorials recommended in the Materials section.

Don't worry if it does not become immediately clear! The exercises are designed to help you understand. Bring your questions to the workshop or office hours; I'll also be doing a demo at the beginning of the workshop to hopefully explain some things better.

If you have little to no programming experience…

- Look through the PHP videos and accompanying slides on the 6.470 website, or follow an equivalent tutorial such as the one on w3schools.
- You'll want to master the basics of PHP syntax and be comfortable with the basic features of the language such as loops, arrays, and strings.
- Time permitting, spend some time on the web-programming specifics of PHP ($_GET, $_POST, etc.) (even just lecture 8 will go a long way)

If you have programming experience but have never written PHP for a web application…

- Look through the slides on the 6.470 website or follow an equivalent tutorial such as the one on w3schools, just to get a grasp for PHP's syntax and some of its quirks.
- Arrays in PHP can be confusing at first; if you've never encountered them, lectures 6-7 provide a fairly in-depth treatment of PHP arrays.
- Focus on the applications of PHP to server-side scripting by spending most of your time on lectures 8-10, focusing specifically on GET, POST, and how forms interact with PHP scripts.

If you have experience writing PHP for a web application…

- Try your hand at the exercises! If you have trouble, look over the relevant lectures or head to Google for specific problems.
- Feel free to bring specific questions to the workshop, even if they are not covered in lectures or exercises.

A (brief) introduction to server-side scripting

Recall that when you visit a website, your browser sends an HTTP request to the server that you are trying to visit. The HTTP response sent from this server is what is shown in your browser. Up until now, your websites have probably been relatively static; every visit to a page generated the same result. *The code (HTML, CSS, and Javascript that controls the styling and behavior of a page) that was sent to your browser by the server was the same.* The HTTP request/response looked something like this:

Now, we will look at server-side technologies. *The biggest difference is that, with server-side technologies such as PHP, the client-side code (HTML, Javascript, etc.) that your browser receives is different, depending on the situation.* This ***dynamic*** behavior is due to logic and processing on the server. For example:

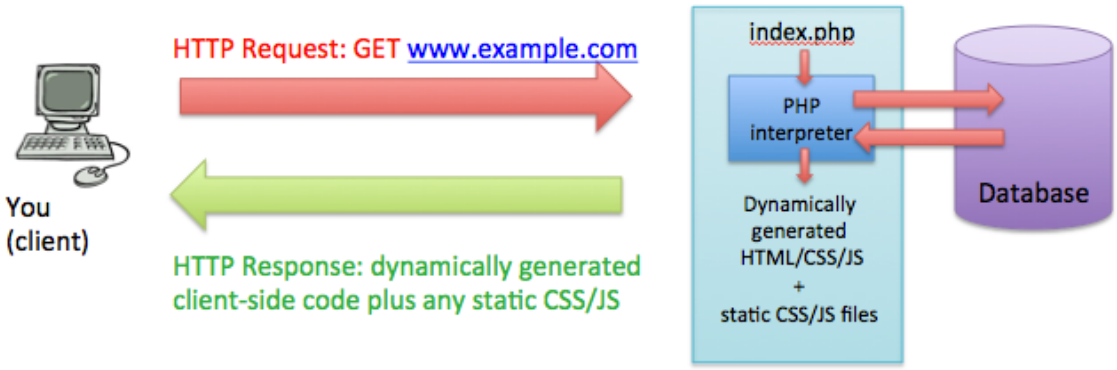
- A restaurant can display an online order form for delivery when the restaurant is open, and a "come back later" message when closed.
- A social network can display different information about a user depending on whether the viewer is a friend or not.
- An online shopping site can direct users that are not logged in to the login page before checkout.

Server-side processing often requires ***input data***, key-value pairs sent from client to server in **GET** or **POST** requests. In addition, the server may access stored user data in the form of ***cookies*** or ***session variables***. (Don't worry if this doesn't make sense now; it'll all be clear after you go through the *Materials and references* section!)

Furthermore, server-side scripts allow us to access ***databases***, which are convenient ways to store data used by websites. The server-side code may create, modify, or delete data in these databases. Finally, the server sends back ***client-side code*** (HTML/CSS/JS), which is displayed in the user's browser.

With server-side scripts and databases, we can do things like:

- Register a user's name, password, and email. When the user logs in, check what was typed against the username and password in the database.
- Someone logs into their online shopping account to buy things at work. Their cart is saved as part of their account, and they checkout later in the day at home, on a different computer.
- Keep track of your friends' interests and activity on a site. Recommend you visit links based on you and your friends' activity.



What you'll learn

- PHP syntax – variables, loops, functions, arrays, all that boring stuff
- How to dynamically generate a response to an HTTP request
- How to extract data from a submitted form
- How to use cookies and session variables to store user state
- How to read from and write to a MySQL database

Materials and references

6.470 video lectures and accompanying slides! These are lectures recorded for IAP 2012 that present all of the basics of PHP and guide you through making a basic backend, culminating in a (very simple) login system.

- http://6.470.scripts.mit.edu/2013/course/php

If you'd rather read something with more guidance than the slides, the w3schools PHP tutorial covers much of the same material. It covers in greater detail how to use PHP with AJAX and also has some other function references.

- http://w3schools.com/php/

The PHP manual is a go-to resource for looking up a specific function. It contains a lot of specifics and examples, but probably isn't the best place to start learning the "big picture".

- http://php.net/manual/en/funcref.php
- Array functions: http://php.net/manual/en/ref.array.php
- Sessions: http://php.net/manual/en/book.session.php
- String functions: http://php.net/manual/en/ref.strings.php
- Math functions: http://php.net/manual/en/ref.math.php
- JSON functions: http://php.net/manual/en/ref.json.php
- URL functions: http://php.net/manual/en/ref.url.php

net.tuts+ has a number of "tutorials" (more like articles), each tackling a specific problem. These are more geared towards more advanced coders who want to add a specific feature, but some are also a good read to get a general idea of some of the more advanced concepts.

- http://net.tutsplus.com/category/tutorials/php/
- In particular: http://net.tutsplus.com/articles/web-roundups/40-invaluable-php-tutorials-and-resources/ contains a number of resources on both OOP in PHP and basics of back-end security

Exercises

After becoming familiar with the syntax and basic functions of PHP, the most important thing is to know how all of the pieces fit together. The interactions between forms (an HTML element), its contents ($GET or $POST elements), the database, and the response can be confusing at first.

The PHP exercises posted on the 6.470 website start off with a few quick functions, designed to get you familiarized with the syntax of PHP and a few string functions. The last two problems are much longer, and involve the creation of a basic login/registration/password recovery system, thus tying together all of the fundamentals that were presented in the slides and tutorials.