COMP 3015

Web Application Development

Week 1 Schedule

1. Course introduction
2. Development environment setup
3. Client-server architecture, understanding request processing at a high level
4. Intro to PHP, writing our first server side programs
5. Lab

Info for COMP 3015

- Official communication (e.g. important info like class cancelled) will be over email
- Join the Discord group! Link on D2L.
  - Use this for problem solving together
  - Collaboration encouraged
- 12 week long course
  - No midterm
  - Final exam on week 12: written + coding
  - ~5 quizzes
  - ~3 assignments
  - ~6 labs

Contact me: **christian_fenn@bcit.ca**, or on Discord: Christian Fenn#6747

Learning Outcomes for COMP 3015

https://www.bcit.ca/courses/web-application-development-with-php-comp-3015/#outcomes

PHP is used as our language of choice in the course, but the topics covered are applicable to any server side software development work.

There is a big focus on understanding HTTP, how servers process requests, and various security topics.

Tips for success in COMP 3015

1. **Ask lots of questions**

2. **Collaborate**
   - You're encouraged to work together - just don't blindly copy and paste code.

3. **Complete all the labs and assignments**
   - These are set up to prepare you for the final and meet all learning outcomes
   - Similar content will be on quizzes and the final exam
   - You'll see similar questions in tech interviews

Development Environment

- **Suggested**: PHPStorm, you can get JetBrains products for free as a student: https://www.jetbrains.com/community/education/#students

- VS Code is OK
  - Need to install PHP extensions for debugging, static code analysis, etc.

Development Environment

- Xdebug debugger: https://xdebug.org/

- Chrome, Firefox or any other web browser you like
  - Devtools for viewing HTTP headers, cookies, network requests, etc.

Quick PHP Facts

- PHP: PHP Hypertext Processor
  - Originally an acronym for “Personal Home Page”

- Originally developed by Danish-Canadian programmer Rasmus Lerdorf in 1994

Quick PHP Facts

- PHP is written in the C programming language: https://github.com/php/php-src
- Used by organizations including Facebook, Slack, MailChimp, Etsy, Wordpress, Square
- PHP is interpreted, not compiled
- Gradual type system

Quick PHP Facts

- Originally PHP was exclusively a templating language
  - Allows for dynamically rendering HTML files on a server, and then sending that dynamically rendered content back to web browsers

```
<body>
    <?php if ($loggedIn): ?>
        <div>The user is logged in</div>
    <?php else: ?>
        <div>The user is not logged in</div>
    <?php endif; ?>
</body>
```

PHP Type System

PHP has gradual, dynamic, weak typing

Gradual: some expressions are typed, some are untyped

Dynamic: type checking is performed at runtime (while the application is executing)

Weak: less strict rules at compile/interpretation time

— **PHP does not have** —

Strong typing: stricter rules at compile/interpretation time. Variables will not be automatically converted between types. Note: dfns of “strong typing” vary.

Static: static analysis of the code is performed at compile time to check type safety

Type Systems, where PHP fits in

Static Typing: types are bound to a variable and checked at compile time (PHP ❌)

Dynamic Typing: types are bound to a value (not a variable) and checked at runtime (PHP ✅)

Strong Typing: variables will not be automatically converted from one type to another (PHP ❌)

Weak Typing: variables will be implicitly converted between types in certain situations (PHP ✅)

*note on strong/weak typing: definitions vary

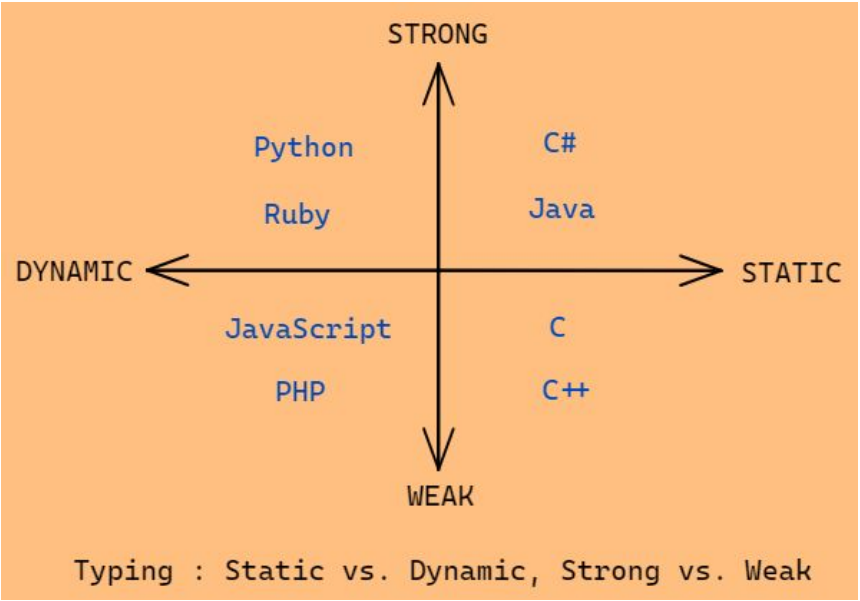


Diagram src: https://stackoverflow.com/a/74194726/3395619

Ports

- When a request is made to a server, the operating system needs a way to determine which running process the data should be sent to.

- Ports identify specific processes and are used in network communication

- Commonly used ports are associated with specific services:
  - 80 → HTTP
  - 443 → HTTPS
  - 22 → SSH

Ports

eg #1. When you sign into Spotify, the OS needs to ensure the response goes to your Spotify application, and not Google Chrome

eg #2. When you go to a website you're making an HTTP request → we need to hit the web server application running on the server

When running server applications, we specify a port number which the server listens for connections on.

How is PHP run?

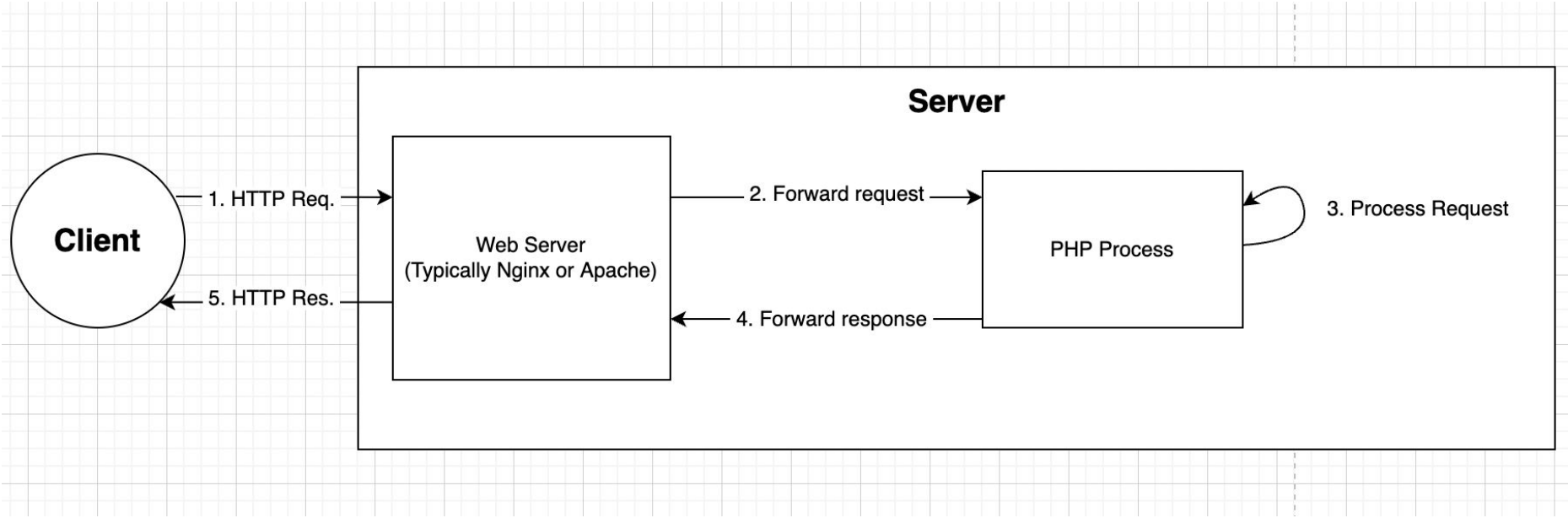
- Command line script e.g. **$ php helloWorld.php**

- As a web application, using a server e.g. **$ php -S localhost:8000**
  - For understanding how running applications using a server works, we will look into client-server architecture.

  - In production environments, and pre-production environments (eg. QA, staging), server such as Apache or Nginx servers are used.

  - For local development we can just use the built-in PHP server.

Client-Server Architecture



Intro to PHP: Hello, World!

```php
<?php

echo "Hello, World!" . PHP_EOL;
```

All PHP scripts begin with **<?php**

PHP_EOL is a constant used to get the correct end of line characters based on the OS PHP is running on. “**\r\n**” on Windows, “**\n**” on anything else (Linux, MacOS, etc.).

Intro to PHP: data types

```php
<?php

$daysPerWeek = 7;           // int
$coffeesConsumed = 99999.5; // float

$firstName = "Christian";   // string, with double quotes
$lastName = 'Fenn';         // string, with single quotes

$employed = true;           // booleans
$playsTrumpet = false;

$likesProgramming = TRUE;
$playsPiano = FALSE;

$images = NULL;             // null, nothing there
$pictures = null;

$numbers = [1, 2, 3, 4];    // array
```

See: variables.php

Intro to PHP: variables, conditionals, type juggling

```php
<?php

$numberOfDaysPerWeek = 7;
if ($numberOfDaysPerWeek === '7') {
    echo "True!\n";
} else {
    echo "False! $numberOfDaysPerWeek was not idential to '7' (they're different types)\n";
}
```

See: https://www.php.net/manual/en/language.operators.comparison.php for all PHP comparison operators

See: type-juggling.php

Intro to PHP: Loops, Arrays

```php
<?php

$listOfDrinks = ['Coffee', 'Water', 'Juice', 'Soda'];

for ($i = 0; $i < count($listOfDrinks); $i++) {
    echo $listOfDrinks[$i] . PHP_EOL;
}

foreach ($listOfDrinks as $drink) {
    echo $drink . PHP_EOL;
}

$index = 0;
while ($index < count($listOfDrinks)) {
    echo $listOfDrinks[$index++] . PHP_EOL;
}
```

See: arrays-and-loops.php

Intro to PHP: Associative Arrays

```php
<?php

$beverageInventory = [
    'Coffee' => 99,
    'Water' => 'Unlimited',
    'Juice' => 64,
    'Soda' => 256,
];

foreach ($beverageInventory as $drinkType => $currentInventory) {
    echo "$drinkType: $currentInventory" . PHP_EOL;
}
```

See: associative-arrays.php

Intro to PHP: functions

```php
<?php

/**
 * @param array $values
 * @return integer
 */
function sumValues(array $values): int {
    $sum = 0;
    foreach ($values as $value) {
        $sum += $value;
    }
    return $sum;
}

$numbers = [-10, 10, 20, 100];

$sum = sumValues($numbers);
echo $sum . PHP_EOL;
```

- We should always type declare parameter and return types.

← **int** here means that the function returns an integer.

See: functions.php, scoping.php

pre, post increment

```php
$count = 0;

echo $count++ . PHP_EOL;

echo ++$count . PHP_EOL;
```

What gets printed out?

See: pre-post-increment.php

Optional: filter, map, reduce

These are functions that apply a callback function to each element in an array.

https://www.php.net/manual/en/function.array-filter

https://www.php.net/manual/en/function.array-map

https://www.php.net/manual/en/function.array-reduce.php

See: filter-map-reduce.php

Optional: pass-by-value, pass-by-reference

Pass-by-value: a copy of the function argument is created and passed to the function.

- The variable defined outside the function is not changed by modifications to it within the function.

Pass-by-reference: the memory address of the variable is passed to the function.

- The variable defined outside the function is changed by modifications to it within the function.

Visually explained:
https://blog.penjee.com/passing-by-value-vs-by-reference-java-graphical/

pass-by-val/ref continued

By default, PHP passes arguments by value.

When an object is passed to a function, the address of the object is passed by value (a copy of the address is made, and passed to the function). This acts like pass-by-reference, so often times you'll hear people say that PHP passes objects by reference, even though this isn't exactly true.

*pass by **reference***

cup =

fillCup( )

*pass by **value***

cup =

fillCup( )

www.mathwarehouse.com

Work this week

- Join the Discord channel
  - Use it to ask questions to me, and your classmates.
- Set up a private, “monolithic” Git repository on GitHub
  - One repository to hold all of your labs, assignments, etc.
  - Invite “**ChristianFenn**” to be a collaborator
- Complete the lab on D2L
  - Get Xdebug working
    - See YouTube video on this
  - Due next week
  - Please use the starter kit on D2L
- Review the slides
  - Make sure to understand the “Client-Server Architecture” slide that has the architecture diagram. Very important!
  - Slides will be posted early so you can pre-read if you like.