Introduction

1PHPD

1PHPD – Introduction

Course Objectives

By the end of the course, students should:

- Understand better the World Wide Web
- Discover about PHP
- Be setup to write PHP code

Time:

- course: 2h
- exercises: 1h

Summary

1. World Wide Web
2. PHP
3. Setup

1. World Wide Web

1. World Wide Web

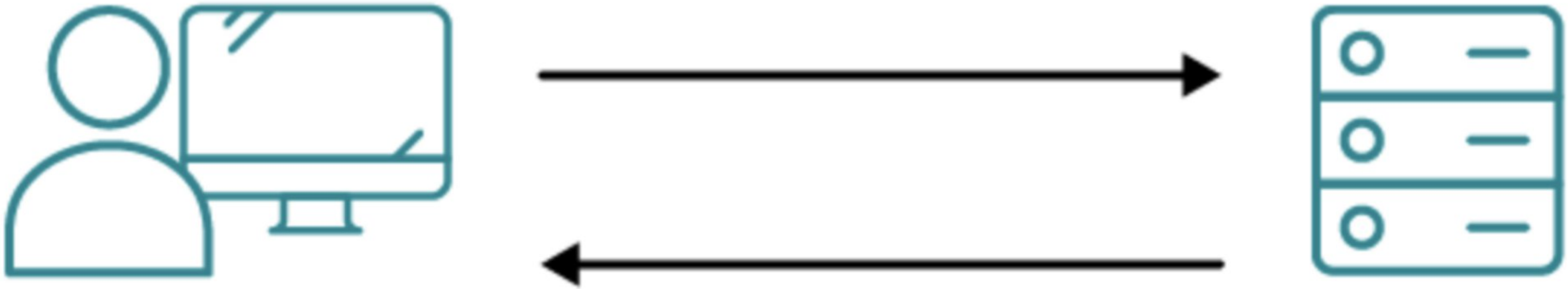
To understand how the web works, it's essential to grasp the interaction between two key entities:

- the client
- the server

This relationship forms the backbone of the World Wide Web, enabling the vast array of services and information we access daily.

1. World Wide Web

If we put that idea into a schema, this would look like this. Which is what you are doing every time you load a web page.



1. World Wide Web

The Client

This is typically a user's device (like a computer, smartphone, or tablet) equipped with a web browser (e.g., Chrome, Firefox, Safari).

The client initiates requests to access web resources.

1. World Wide Web

The Server

A server is a computer system that hosts (stores) web resources such as websites, and it's designed to respond to requests from clients.

Web servers run software (like Apache, Nginx, or Microsoft IIS) that enables them to handle these requests.

1. World Wide Web

The Internet

It's the global network that connects clients to servers, allowing them to communicate.

Data travels across the internet through various routers and switches, following the best available path.

1. World Wide Web

But this is only a simplified schema and explanation of how the World Wide Web (Web) is operating.

As you may have guessed, it is not as simple as two lines between you and a server!
We need a lot of mechanism to be able to translate a simple text as "www.google.com" into a page displaying different results based on your request.

1. World Wide Web

Step 1: The User's Request

It all starts when you type a URL (Uniform Resource Locator) into your web browser or click on a link.

The URL specifies the web resource you want to access, consisting of the protocol (e.g., HTTP or HTTPS), the server's domain name (e.g., www.example.com), and sometimes a specific file path.

1. World Wide Web

Step 2: DNS Lookup

Your browser needs to translate the domain name into an IP address (Internet Protocol address), which uniquely identifies the server on the internet.

This translation is performed by a DNS (Domain Name System) server, functioning like the internet's phonebook.

1. World Wide Web

Step 3: Establishing a Connection

Once the IP address is known, your browser initiates a connection to the server using the TCP/IP protocol (Transmission Control Protocol/Internet Protocol).

If HTTPS is used, a secure connection is established through SSL/TLS (Secure Sockets Layer/Transport Layer Security) encryption.

1. World Wide Web

Step 4: Sending the Request

Through this connection, the browser sends an HTTP (HyperText Transfer Protocol) request to the server.

This request includes the method (e.g., GET for retrieving data), the URL, and headers that carry additional information (like the browser type).

1. World Wide Web

Step 5: The Server's Response

The server receives the request, processes it, and sends back an HTTP response.

This response includes a status code (indicating success or error), headers, and the requested content (like an HTML page).

1. World Wide Web

Step 6: Rendering the Web Page

Your browser receives the response and begins rendering the web page. It interprets the HTML, CSS (Cascading Style Sheets), and JavaScript content to display the page.

If additional resources (like images or stylesheets) are needed, the browser makes further requests to the server.

1. World Wide Web

Step 7: User Interaction and Dynamic Content

As you interact with the web page (clicking links, submitting forms), your browser may make additional requests to the server.

Modern web applications often use AJAX (Asynchronous JavaScript and XML) for dynamic content updates without reloading the entire page.

1. World Wide Web

This is the main steps that are really happening when you navigate online. And this is almost the same (using the same mechanisms) if you are playing online on your console.

But behind the scene there is even more

- How does the DNS know the matching between the URL and the IP? Your computer does not know all the IPs in the world
- How are the information propagated and “sent”? Following and using which format?
- How about IPv4 and IPv6?

1. World Wide Web

Some of the previous questions may already have been answered in other classes, and some other topics will be in the next years.

In a world dominated by software running over internet, understanding the basics of it is crucial. Even more for people that will be directly working on it, be it software - network - cloud engineers or any others disciplines.

1. World Wide Web

This client-server model is the foundation of web interaction, enabling the dynamic and interactive experiences we've come to expect, from searching content on Google, displaying pictures on Instagram or searching your next meal on UberEats.

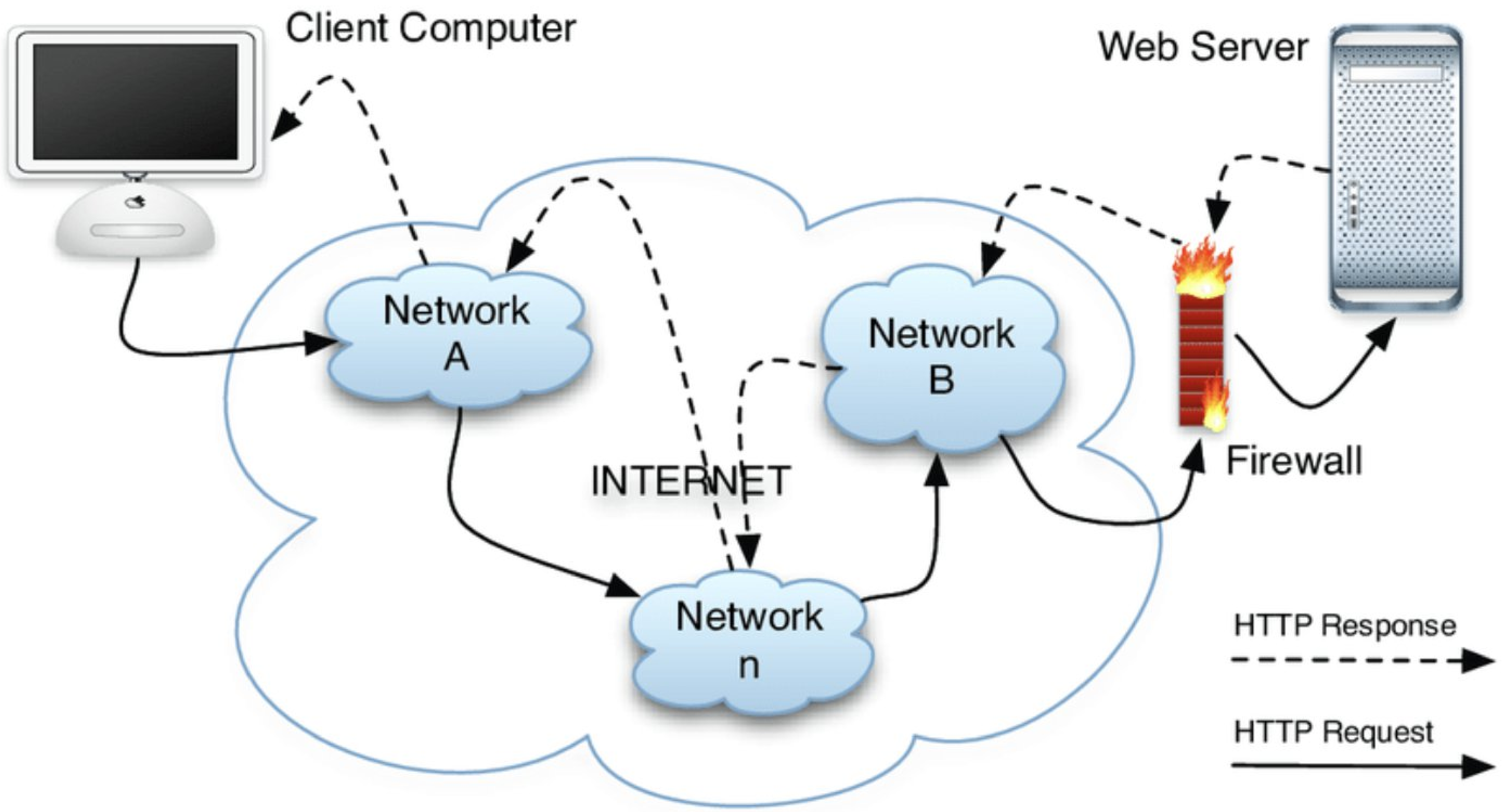
The process, although complex, happens remarkably quickly, illustrating the efficiency and power of web technologies, the globality of the operation taking less than one seconde.

1. World Wide Web

Understanding this interaction offers insights into the design, development, and optimization of web resources, ensuring they are accessible, responsive, and secure for users worldwide.

This may not be important to you now, but this knowledge is used on a daily basis if you work on huge scale projects and applications. But even understanding that can be useful to build and optimize something like your personal website.

1. World Wide Web



Exercises

2. PHP

2. PHP

PHP, which originally stood for Personal Home Page but now stands for the recursive acronym PHP: Hypertext Preprocessor, is a widely-used open-source server-side scripting language designed specifically for web development.

It enables developers to write dynamically generated web pages efficiently.
PHP scripts are executed on the server, and the result is sent back to the client as plain HTML.

2. PHP

PHP, as a pivotal server-side scripting language for web development, has a rich history marked by continuous evolution and improvements.

PHP was created by Rasmus Lerdorf in 1994.
Initially, it was a simple set of Common Gateway Interface (CGI) binaries written in the C programming language.

Lerdorf initially called this set of scripts "Personal Home Page Tools" to maintain his personal webpage, hence the acronym "PHP."

2. PHP

The original PHP was designed to replace a set of Perl scripts to display his resume and collect certain data, such as traffic on his webpage.

It wasn't intended to become a new programming language; rather, it emerged as a tool to facilitate the creation of dynamic and interactive web pages.

2. PHP

PHP's evolution is marked by significant releases that introduced new features, improvements, and security enhancements.

With more than 30 years of active development, and a lot of changes in the ecosystem, PHP is still a relevant language that is one of the core technology that helped build the web, and all our usages.

2. PHP

PHP/FI 2 (1995)

The second version, called PHP/FI, could communicate with databases, enabling dynamic web content.

This version laid the groundwork for PHP to become a widespread server-side scripting language.

2. PHP

PHP 3 (1998)

This release marked the official beginning of PHP's evolution into a comprehensive programming language.

PHP 3 was a complete rewrite that introduced the concept of Zend Engine, providing the foundation for future development.

It was also the first version to officially use the recursive acronym "PHP: Hypertext Preprocessor."

2. PHP

PHP 4 (2000)

Powered by Zend Engine 1.0, PHP 4 enhanced performance and introduced features like sessions and output buffering, solidifying PHP's role in web development.

2. PHP

PHP 5 (2004)

This version introduced the powerful Zend Engine II with object-oriented programming (OOP) significantly improved, providing better support for classes and objects, and introduced exceptions handling, and more powerful XML processing capabilities.

2. PHP

PHP 7 (2015)

A major leap forward, PHP 7 significantly improved performance thanks to the new Zend Engine 3.0.

It introduced return type declarations, scalar type declarations, a null coalescing operator, and improved error handling.

PHP 7 offered a substantial performance boost compared to PHP 5.6, making applications faster and more efficient.

2. PHP

PHP 8 (2020)

The latest major release, PHP 8 introduced JIT (Just In Time) compilation, attributes, union types, named arguments, match expressions, and improvements in the type system and error handling.

These features aim to make PHP more robust, faster, and more versatile for modern web development needs.

2. PHP

PHP continues to be actively developed, with PHP 8.x being the latest version series.

The PHP community actively supports the language, contributing to its core, developing frameworks, and creating resources and documentation to aid developers.

2. PHP

PHP's journey from a simple scripting tool for personal web pages to a full-fledged programming language powering vast portions of the web illustrates its adaptability, robustness, and the strong community support behind it.

Despite the emergence of numerous web technologies over the years, PHP remains a cornerstone of web development, testament to its enduring relevance and ongoing evolution.

2. PHP

Its popularity and sustained use over the years stem from several key features and capabilities that make it an excellent choice for developers looking to build dynamic and interactive web applications.

2. PHP

Server-Side Scripting

PHP is primarily used for server-side scripting.
This means that the PHP code is executed on the server, and only the resulting HTML is sent to the client's web browser.

This process allows for dynamic content generation based on conditions (like user input, time of day, etc.) on the server side before the content is sent to the user.

2. PHP

Database Integration

PHP has strong support for interacting with databases.

The most common use case is with MySQL databases, but it can interact with a wide variety of database systems, making it a powerful tool for building dynamic web applications that rely on database data.

2. PHP

Cross-Platform

PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.) and is compatible with almost all servers used today (Apache, Nginx, etc.).

This cross-platform support ensures that PHP applications can be deployed on a wide range of server environments.

2. PHP

Rich Library Support

PHP has a rich set of built-in libraries that provide a plethora of functions to perform tasks like working with files, images, XML or encrypting data.

Moreover, there are extensive extensions and libraries available from the community, which add functionalities for accessing various APIs, frameworks, and tools.

2. PHP

Embedded

PHP code can be easily embedded within HTML code by using special PHP tags. This makes it particularly convenient for developers to add dynamic content to web pages without having to use large amounts of scripting in the HTML.

2. PHP

PHP remains one of the most widely used server-side programming languages for web development.

Its popularity is underpinned by several factors, including its ease of use, extensive documentation, and a strong ecosystem of frameworks and content management systems (CMS) like WordPress, Joomla, and Drupal.

2. PHP

Web Usage

PHP is used by a significant portion of websites on the internet.

Various sources estimate that over 75% of websites with known server-side programming languages use PHP.

This high percentage underscores PHP's dominance in server-side web development.

2. PHP

Content Management Systems (CMS)

A substantial part of PHP's popularity can be attributed to WordPress, the world's most popular CMS, which is built on PHP.

WordPress alone powers a considerable percentage of all websites on the internet, contributing significantly to PHP's widespread use.

2. PHP

Frameworks

PHP frameworks like Laravel and Symfony are popular choices among developers for building web applications.

Laravel, in particular, has seen a surge in popularity due to its elegant syntax, robust features, and comprehensive documentation.

2. PHP

Version Distribution

The introduction of PHP 7 marked a significant improvement in performance and efficiency, leading to its rapid adoption among developers.

A substantial portion of PHP sites had migrated to PHP 7.x or newer versions, benefiting from improved speed and enhanced language features.

2. PHP

Developer Preferences

Despite the emergence of newer programming languages and technologies, PHP remains a preferred choice for many developers, especially those working on web applications and CMS-based projects.

Surveys and reports from developer communities like Stack Overflow and GitHub indicate a steady interest in PHP, although there's also growing interest in other technologies like Node.js, Python, and Ruby.

2. PHP

Job Market and Industry Demand

PHP skills continue to be in demand in the job market, especially for roles involving web development, CMS customization, and e-commerce platforms.

While the demand for PHP developers may not match that for developers with expertise in newer technologies, there's still a healthy market for PHP development, particularly in maintaining and upgrading existing PHP-based applications.

2. PHP

Despite competition from other server-side technologies, PHP's ease of use, large community, and robust ecosystem ensure its continued relevance in web development.

The language's adaptability to evolving web standards and practices will likely sustain its usage across various domains, including e-commerce, content management, and custom web applications.

2. PHP

PHP has been a cornerstone of web development for decades, offering the tools and flexibility needed to build everything from small websites to complex web applications.

Despite the rise of other programming languages and technologies, PHP remains popular due to its continuous evolution, vast community support, and adaptability to modern web development needs.

2. PHP

Its ease of use, combined with powerful features, ensures that PHP will continue to play a significant role in the future of web development.

Exercises

3. Setup

3. Setup

Installing and running PHP on your system can vary depending on your operating system (OS), and we can find two majors ways.

- Installing and running PHP only
- Installing PHP with a web server and a full ecosystem

Let’s start by installing PHP only.

3. Setup

macOS - Using Homebrew

- Install Homebrew
- Once Homebrew is installed, run “brew install php” to install the latest version of PHP.
- After installation, you can type php -v in the Terminal to check the PHP version.

3. Setup

Linux (Ubuntu/Debian)

- Open a terminal window and update your package list with “sudo apt update”.
- Install PHP by running “sudo apt install php”.
- Run php -v in your terminal to check the installed PHP version.

Depending on your distribution the package name can be a bit different.

3. Setup

To verify that PHP is correctly installed and running, you can create a simple PHP script

- Open a text editor and create a new file named test.php.
- Add the following PHP code to the file. (code on next slide)
- Save the file and close the text editor.
- Open a terminal or command prompt, navigate to the directory containing test.php, and run the command php test.php.

If PHP is correctly installed and configured, you should see "Hello, World!" printed to the terminal or command prompt.

3. Setup

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

3. Setup

For developing web applications, you'll likely want to run PHP in conjunction with a web server like Apache or Nginx.

For a simpler setup, especially during development, you can use PHP's built-in server:

- Navigate to the directory containing your PHP files.
- Run php -S localhost:8000 to start the PHP built-in server.
- Open a web browser and go to http://localhost:8000/. If you have an index.php file in that directory, it will be served by the PHP server.

3. Setup

For more complex applications or production environments, refer to the specific documentation for integrating PHP with Apache, Nginx, or other web servers, and consider using tools like XAMPP, WAMP, or MAMP for an all-in-one solution that includes PHP, a web server, and MySQL.

As we will use a simple database at the end of the module, and that you will require one for your project, it may be better to already install them.

Exercises