Guia Ágil

TechGuide - Alura

PHP

Nivel 1

☐ **PHP - Fundamentals**:

- PHP is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications.

- Knowing the primitive types

- Declaring variables, considering the different types

- Using conditional structures ('if', 'else')

- Knowing the assignment and comparison operators

- Using repetition structures and loops ('while', 'for')

- Using functions, passing parameters and arguments

- Manipulating methods

- Manipulating arrays and lists

☐ **PHP - Collections**:

- A collection represents a group of objects, known as its elements. They are like containers that group multiple items into a single unit. Some collections allow duplication of elements and some do not. Some are ordered and some are unordered.

- Learning the uses and differences between arrays, lists, stacks, hash maps, heaps

- Working with Iterators, Generators and Collections

☐ **Object-oriented Programming Concepts**:

- Object-oriented programming (OOP) is a programming paradigm based on the concept of 'objects', which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods). A common feature of objects is that procedures (or methods) are attached to them and can access and modify the object's data fields. Some of the main concepts are classes and instances, inheritance, and encapsulation.

- How objects work

- Creating and using constructors

- What classes are

- Creating and using Methods

- How encapsulation works

- What inheritance is

- What polymorphism is

- How interfaces work

- What abstractions are

☐ **PHP - Error handling**:

- Error handling refers to the response and recovery procedures from error conditions present in a software application. In other words, it is the process comprised of anticipation, detection and resolution of application, programming or communication errors.

- Knowing and handling the most common exceptions

- Knowing the types of errors and in which situations they can occur

- Using 'try' and 'catch' for error handling

- Learning about error handlers

- Creating specific exceptions according to your application's needs

☐ **PHP - Persistence**:

- The concept of "data persistence" refers to ensuring that the information inputted into an application will be stored in a medium where it can be retrieved consistently. In other words, they are permanent records that are not lost when the session is closed.
- Knowing how to use PDO
- Making insertions, changes and removals in a database
- Creating queries on a database using filters

☐ **PHP - Files and Streams**:

- Streams are the way of generalizing file, network, data compression, and other operations which share a common set of functions and uses. In its simplest definition, a stream is a resource object which exhibits streamable behavior. That is, it can be read from or written to in a linear fashion, and may be able to seek an arbitrary location within the stream.
- Opening, reading and manipulating files
- Applying wrappers and filters on streams
- Using streams
- Knowing encodings and character sets

Nivel 2

☐ **PHP - Composer**:

- Composer is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you.
- Managing dependencies
- Including third-party libraries

☐ **PHP - Testing**:

- Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.
- Using unit tests
- Using integration testing
- Using behavioral testing
- Using mocks

☐ **PHP - MVC**:

- Model–view–controller (MVC) is a software architectural pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements. This is done to separate internal representations of information from the ways information is presented to the user.
- Understanding the concept of the MVC (Model-View-Controller) pattern
- Splitting the responsibilities between the MVC layers

☐ **Laravel**:

- Laravel is a free and open-source PHP framework for developing web systems that use the MVC (model, view, controller) pattern.
- Getting to know the features of the Laravel framework
- Working with routes
- Using Blade files
- Using its ORM features
- Dealing with dependency injection

☐ **PHP - Symfony**:

- Symfony is an open source PHP, full-stack MVC framework, and incorporates many features and good development practices of the PHP domain.

- Getting to know the main features of the framework

- Using components

- Defining resources and routes

- Twig templates

☐ **PHP - Communication with APIs**:

- An API is an interface that software developers use to program interaction with software components or features outside their own code. An even simpler definition is that an API is the part of a software component that is accessible to other components.

- Understanding what a REST API is

- Knowing the basic HTTP communication commands

- Getting to know cURL

- Building APIs in Laravel

- Knowing how to make authenticated requests for Web APIs

☐ **PHP - XDebug**:

- Xdebug is a PHP extension which provides debugging and profiling capabilities.

- Using and enhancing var_dump with XDebug

- Analyzing and understanding the execution stack

- Debugging the code

- Debugging remotely with Docker

Nivel 3

☐ **PHP - Asynchronous**:

- In asynchronous programming the functions are not executed in order. With asynchrony, we can interrupt the code to get some other information needed to continue execution. This means that the code waits for another part of the code and while it waits it executes the other parts.

- Learning how to use sockets

- Getting to know reactive programming libraries

- Learning about coroutines

☐ **PHP - Functional programming**:

- Functional programming is a programming paradigm premised on using functions. Functions are, according to this paradigm, at the epicenter of a code base.

- Understanding in-depth about functions in PHP

- Getting to know arrays manipulation functions

- Performing function composition using pipelines

☐ **Microservices architecture**:

- Microservices are an architectural approach in which software consists of small independent services that communicate with each other and are organized according to their business domains.

- Learning the concept of planned architecture for microservices

- Performing communication using APIs

- Improving the scalability of a system

☐ **Containers**:

- Containers are software packages that contain all the elements needed to run in any environment.

- Kubernetes (also known as k8s or "kube") is an open source container orchestration platform that automates many of the manual processes involved in deploying, managing, and scaling containerized applications.

- Isolating your software to run independently

- Deploying software in clusters

- Modularizing your system into smaller packages

- Getting to know the Docker platform

- Getting to know Kubernetes

☐ **Object Calisthenics**:

- Object Calisthenics are basically programming exercises, as a set of 9 rules to help write better object-oriented code.

- Getting to know the rules of Object Calisthenics

- Simplifying classes and methods

- Keeping code shorter and more cohesive

☐ **Reflection and attributes**:

- Reflection objects are used for obtaining type information at runtime. The classes that give access to the metadata of a running program are in the System.Reflection namespace.

- Writing code that reads object information and metadata at runtime

- Getting class names at runtime and creating objects of a class

Habilidade Auxiliar: Infrastructure and good practices

☐ **Git & GitHub - Fundamentals**:

- Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

- GitHub is a hosting service for software development and version control using Git.

- Creating a repository

- Cloning a repository

- Committing, pushing and pulling to and from the repository

- Reversing a commit

- Creating branches and pull requests

- Handling merge and conflicts

☐ **HTTP - Fundamentals**:

- HTTP stands for Hyper Text Transfer Protocol. Communication between client computers and web servers is done by sending HTTP Requests and receiving HTTP Responses.

- Understanding the difference between HTTP verbs

- Testing requests and checking the status codes in the browser

- Learning how to make a HTTP request on the command line with WGET

- Downloading an image with WGET

- Performing a POST

☐ **JSON**:

- JSON stands for JavaScript Object Notation. It is a text format for storing and transporting data.

- Creating an object

- Transforming an object into a string

- Transforming a string into an object

- Manipulating an object

☐ **Command Line - Fundamentals**:

- CLI is a command line program that accepts text input to execute operating system functions.

- Knowing the most important commands

☐ **Cloud - Fundamentals**:

- Cloud, or cloud computing, is the distribution of computing services over the Internet using a pay-as-you-go pricing model. A cloud is composed of various computing resources, ranging from the computers themselves (or instances, in cloud terminology) to networks, storage, databases, and everything around them. In other words, everything that is normally needed to set up the equivalent of a server room, or even a complete data center, will be ready to use, configured, and run.

- Knowing the difference between IaaS, PaaS and SaaS

- Knowing the largest cloud providers

- Specializing in a specific provider of your choice

☐ **SQL - Fundamentals:**

- Structured Query Language (SQL) is a standardized programming language that is used to manage relational databases and perform various operations on the data in them.

- Knowing the most common SQL commands

- Using SELECT to query a table

- Using INSERT to insert data into a table

- Using UPDATE to update a table

- Using DELETE to remove data from a table

- Using JOIN to connect data from multiple tables

- Knowing the clauses (FROM, ORDER BY, etc.)

☐ **SOLID:**

- SOLID has five principles that are considered best practices in software development that help programmers write cleaner code by separating responsibilities, reducing docking, easing refactoring, and encouraging code reuse.

☐ **Design Patterns:**

- In software engineering, a Design Pattern is a general, reusable solution to a commonly occurring problem within a given context in software design. It is a description or template for how to solve a problem that can be used in many different situations. Design Patterns are formalized best practices that the programmer can use to solve common problems when designing an application or system.

- Getting familiarized with and applying the main Design Patterns

☐ **Clean Architecture:**

- Clean architecture is a way of developing software, such that just by looking at the source code of a program, you should be able to tell what the program does.

☐ **Domain-Driven Design (DDD) Concepts**:

- Domain-Driven Design (DDD) is an approach to software design and development that is first informed by business requirements. The program components (objects, classes, arrays, etc.) indicate the industry, sector, or business domain in which the business operates.
- Modeling domains effectively
- Basing complex projects on domain models
- Getting to know the building blocks of DDD

Habilidade Auxiliar: Front-end

☐ **HTML - Fundamentals**:

- HTML is a markup language that defines the structure of your content. HTML consists of a series of elements, which you use to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on.
- Learning which tags are required for basic HTML
- Creating a text paragraph
- Displaying an image
- Knowing the difference between 'h1', 'h2', 'h3', etc.
- Creating a hyperlinked text
- Creating a form with relevant fields
- Creating an ordered or unordered list of items
- Creating a list of items within a dropdown list
- Linking to a CSS file
- Creating a table

- Adding IDs and classes

☐ **CSS - Fundamentals**:

- Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML. CSS can be used for very basic document text styling — for example, for changing the color and size of headings and links. It can be used to create a layout — for example, turning a single column of text into a layout with a main content area and a sidebar for related information. It can even be used for effects such as animation.
- Learning the visual structure of a page, with 'margin' and 'padding'
- Establishing the size with 'width' and 'height'
- Learning about the position of an element ('static', 'relative' or 'absolute')
- Learning about the display of an element ('block', 'inline', 'inline-block')
- Learning how to position images in relation to text
- Learning about alignment
- Learning about font style
- Learning the differences and advantages of using the different units of measurement in CSS (%, relative, etc)
- Connecting to the elements (IDs, classes) of an HTML file
- Changing the characteristics of an element when the mouse hovers over it
- Learning box-sizing
- Learning Flexbox
- Learning Grid

☐ **JavaScript - Fundamentals**:

- JavaScript is the world's most popular programming language and is one of the core technologies of the World Wide Web, alongside HTML and CSS. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles.

- Knowing the primitive types

- Declaring variables, considering the difference between 'var', 'let' and 'const

- Using conditional structures ('if', 'else')

- Know the assignment and comparison operators ('=', '==', '===')

- Using repetition structures and loops ('while', 'for')

- Using functions, passing parameters and arguments

- Manipulating arrays and lists

- Getting data from an API

- Making asynchronous calls using 'Async/Await', 'Promise', etc

---