Guia Ágil

TechGuide - Alura

Front-end

Nivel 1

- [ ] **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

☐ **DOM - Fundamentals**:

- The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.
- Understanding how the DOM tree works
- Accessing and manipulating HTML and CSS elements
- Accessing the parents and children of an element
- Inserting a new element to the tree
- Removing an element from the tree
- Waiting for an event on a certain page element using addEventListener()

☐ **Javascript - Accessibility**:

- Web accessibility is the inclusive practice of ensuring there are no barriers that prevent interaction with, or access to, websites by people with physical disabilities, situational disabilities, and socio-economic restrictions on bandwidth and speed.
- Writing code with accessibility in mind

☐ **SEO Strategies**:

- Search engine optimization (SEO) is the process of improving the quality and quantity of website traffic to a website or a web page from search engines.
- Choosing keywords
- Understanding how Google ranks pages
- Knowing the ranking factors
- Performing Link Building

- [ ] **Responsive Design**:

- Responsive web design (RWD) or responsive design is an approach to web design that aims to make web pages render well on a variety of devices and window or screen sizes from minimum to maximum display size to ensure usability and satisfaction.
- Adjusting your pages to the user's screen size
- Learning about Media queries
- Knowing the concept of Mobile first

Nivel 2

- [ ] **JavaScript - Callbacks and Promises**:

- A Promise is a proxy for a value not necessarily known when the promise is created. This lets asynchronous methods return values like synchronous methods - instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future.
- A callback function is a function passed into another function as an argument, which is then invoked inside the outer function to complete some kind of routine or action.
- An async function is a function declared with the async keyword, and the await keyword is permitted within it. The async and await keywords enable asynchronous, promise-based behavior to be written in a cleaner style, avoiding the need to explicitly configure promise chains.
- Understanding the concept of asynchronous programming

- Writing asynchronous code understanding the concept of promises in JavaScript
- Using JavaScript methods, keywords and objects for handling promises like 'Async/Await', '.then()', 'Promise', etc
- Learning in which situations you need to use asynchronous programming
- Calling APIs with `fetch()`

- [ ] **JavaScript - 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

- [ ] **JavaScript - 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 on what occasions and how to use `throw`
- Creating specific exceptions according to your application's needs

- [ ] **JavaScript - ES6**:

- ECMAScript 2015 was the second major revision to JavaScript. ECMAScript 2015 is also known as ES6 and ECMAScript 6.

- Knowing the differences and most importante features of ES6

- [ ] **JavaScript - Modularization**:

- JavaScript modules allow you to break up your code into separate files, which makes it easier to maintain the code-base.
- Isolating parts of code in modules
- Using import and export

- [ ] **Front-end Semantic Versioning**:

- SemVer (Semantic Versioning) is a simple set of rules and requirements that dictate how version numbers are assigned and incremented.
- Organizing the dependencies of a project
- Avoiding the "dependency hell"

- [ ] **Jest**:

- Jest is a JavaScript test runner that lets you access the DOM via jsdom. It provides a great iteration speed combined with powerful features like mocking modules and timers so you can have more control over how the code executes.
- Testing components

- [ ] **Cypress**:

- Cypress is a front-end testing tool that allows for setting up, writing, running, and debugging tests.

Nivel 3

- [ ] **Data Structures**:

- In the context of computers, the data structure is a specific way of storing and organizing data in the computer's memory so that these data can be easily retrieved and efficiently used when needed later.
- Knowing the main data structures (linked list, stack, queue, tree, etc)

- Implementing the main data structures

- [ ] **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

- [ ] **JavaScript - Storage**:

- Local storage allows developers to store and retrieve data in the browser. The data stored in local storage will not expire, this means the data will persist even if the tab or the browser window is closed.
- Storing data in the front-end with localStorage
- Manipulating stored data
- Persisting stored data

- [ ] **JavaScript - Concurrency**:

- Concurrency is a programming paradigm for building programs that make use of the simultaneous execution of several interactive computational

tasks, which can be implemented as separate programs or as a set of threads created by a single program.

- Executing tasks in parallel

- [ ] **TypeScript - Fundamentals**:

- TypeScript is a strongly typed programming language that builds on JavaScript.
- Understanding in depth what types are and the importance of typed programming
- Learning what TypeScript is, why it was created, how it works and its relationship with JavaScript
- Knowing the TypeScript tools (integration with the code editor, static checker and compiler)
- Writing code in TypeScript using its tools (interfaces, enum, decorators, etc.)

- [ ] **GraphQL**:

- GraphQL is a new API standard that provides a more efficient, powerful and flexible alternative to REST. It was developed and open-sourced by Facebook and is now maintained by a large community of companies and individuals from all over the world.
- Understanding how GraphQL is used in API development
- Creating APIs using GraphQL libraries and frameworks

- [ ] **Apollo Client**:

- Apollo Client is a comprehensive state management library for JavaScript that enables you to manage both local and remote data with GraphQL.

Habilidade Auxiliar: Infrastructure and Back-end

- [ ] **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

- [ ] **YARN**:

- Yarn is a package manager for your code. It allows you to use and share code with other developers. Code is shared through something called a package (sometimes referred to as a module). A package contains all the code being shared as well as a package.json file which describes the package.
- Managing packages
- Managing dependencies
- Installing packages offline
- Commands
- The yarn.lock file

Habilidade Auxiliar: UX and Design

- [ ] **Design Systems**:

- A design system is a collection of reusable components, guided by clear standards, that can be assembled together to build applications.

- Creating and maintaining libraries that will be consumed and used as a standard for building a project

- [ ] **Figma - Fundamentals**:

- Figma is a collaborative web application for interface design. The feature set of Figma focuses on user interface and user experience design, with an emphasis on real-time collaboration, utilising a variety of vector graphics editor and prototyping tools.
- Creating page layouts and components

- [ ] **Design components**:

- Knowing the components that describe a layout or interface

- [ ] **Color systems**:

- Defining a color palette that makes sense for a given interface

- [ ] **How to use Fonts**:

- Choosing the most appropriate font for a given project
