Guia Ágil

TechGuide - Alura

PHP

Nivel 1

- [ ] **Lógica de Programação**:
  - Aprender lógica de programação, fundamental para o desenvolvimento de software
  - Conhecer as bases para se criar, analisar e resolver problemas computacionais de forma estruturada e eficiente
  - Entender o que são tipos de dados
  - Declarar variáveis, considerando os diferentes tipos
  - Conhecer os operadores de atribuição e comparação
  - Usar estruturas condicionais
  - Usar estruturas de repetição e laços
  - Usar funções, passando parâmetros e argumentos
- [ ] **PHP - Fundamentos**:
  - PHP é uma linguagem de programação que permite aos desenvolvedores web criar conteúdo dinâmico que interage com bancos de dados. O PHP é basicamente usado para desenvolver aplicações de software baseadas na web.
  - Conhecer os tipos primitivos

- Declarar variáveis
- Usar estruturas condicionais ('if', 'else')
- Conhecer os operadores de atribuição e comparação ('=', '==', '===')
- Usar estruturas de repetição e laços ('while', 'for')
- Usar funções, passando parâmetros e argumentos
- Manipular arrays e listas

- [ ] **PHP - Coleções**:

- Uma coleção representa um grupo de objetos, conhecidos como seus elementos. Eles são como recipientes que agrupam vários itens em uma única unidade. Algumas coleções permitem a duplicação de elementos e outras não. Algumas são ordenadas e outras não ordenadas.
- Aprender os usos e diferenças entre arrays, listas, pilhas, hash maps, heaps
- Trabalhar com Iterators, Geradores e Collections

- [ ] **Conceitos de Orientação a Objetos**:

- A Programação Orientada a Objetos é um paradigma de programação de software baseado na composição e interação entre diversas unidades chamadas de 'objetos' e as classes, que contêm uma identidade, propriedades e métodos). Ela é baseada em quatro componentes da programação: abstração digital, encapsulamento, herança e polimorfismo.
- Como funcionam objetos
- Criar e utilizar construtores
- O que são classes
- Criar e utilizar métodos
- Como funciona encapsulamento
- O que é herança
- O que é polimorfismo
- Como funcionam interfaces
- O que são abstrações

- [ ] **PHP - Manipulação de Erros**:
    - O tratamento de erros refere-se aos procedimentos de resposta e recuperação de condições de erro presentes em um aplicativo de software. Em outras palavras, é o processo composto de antecipação, detecção e resolução de erros de aplicação, de programação ou de comunicação.
    - Tratar exceções pré-definidas
    - Uso de 'try' e 'catch'
    - Criar exceções específicas
    - Aprender sobre error handlers
    - Fazer o processo de Debug
- [ ] **PHP - Persistência**:
    - O conceito de "persistência de dados" refere-se a garantir que as informações inseridas na aplicação serão armazenadas em um meio em que possam ser recuperadas de forma consistente. Ou seja, são registros permanentes e que não são perdidos quando há o encerramento da sessão.
    - Conhecer o PDO
    - Fazer inserções, alterações e remoções em um banco de dados
    - Fazer consultas em um banco de dados usando filtros
- [ ] **PHP - Arquivos e Streams**:
    - Os 'streams' (fluxos) são a forma de generalizar operações com arquivos, rede, compressão de dados e outras que compartilhem um conjunto comum de funções e usos. Em sua definição mais simples, um stream é um objeto que exibe um comportamento 'streamable'. Ou seja, ele pode ser lido ou escrito de uma forma linear, e pode ser capaz de buscar uma localização arbitrária dentro do fluxo.
    - Abrir, ler e manipular arquivos
    - Aplicar wrappers e filtros nos streams
    - Usar streams
    - Conhecer encodings e character sets

Nivel 2

- [ ] **PHP - Composer**:
  - O Composer é uma ferramenta para o gerenciamento de dependências em PHP. Ele permite que você declare as bibliotecas das quais seu projeto depende e as gerenciará (instalar/atualizar) para você.
  - Gerenciar dependências
  - Incluir bibliotecas de terceiros
- [ ] **PHP - Testes**:
  - O teste de software é o processo de avaliação e verificação de que um software realmente faz o que deveria fazer. Os benefícios dos testes incluem a prevenção de bugs, a redução dos custos de desenvolvimento e a melhoria do desempenho.
  - Usar testes unitários
  - Usar testes de integração
  - Usar testes de comportamento (behavior)
  - Usar mocks
- [ ] **PHP - MVC**:
  - Model-view-controller (MVC) é um padrão de arquitetura de software comumente usado para desenvolver interfaces de usuário que dividem a lógica do programa relacionado em três elementos interligados. Isto é feito para separar as representações internas de informação das formas como a informação é apresentada para o usuário.
  - Entender o conceito do padrão MVC (Model-View-Controller)
  - Separar as responsabilidades entre as camadas MVC
- [ ] **Laravel**:
  - Laravel é um framework PHP livre e open-source para o desenvolvimento de sistemas web que utilizam o padrão MVC (model, view, controller).
  - Conhecer as características do framework Laravel

- Trabalhar com rotas
- Usar arquivos Blade
- Usar as suas funcionalidades ORM
- Lidar com injeção de dependências

- [ ] **PHP - Symfony**:

- Symfony é um framework PHP de código aberto e de estrutura MVC que incorpora muitas funcionalidades e boas práticas de desenvolvimento do domínio PHP.
- Conhecer as principais características do framework
- Utilizar componentes
- Definir recursos e rotas
- Templates Twig

- [ ] **PHP - Comunicação com APIs**:

- Uma API é uma interface que desenvolvedores de software utilizam para programar a interação com componentes ou recursos de software fora de seu próprio código. Uma definição ainda mais simples é que uma API é a parte de um componente de software que é acessível a outros componentes.
- Entender o que é uma API REST
- Conhecer os comandos básicos de comunicação HTTP
- Conhecer o cURL
- Construir APIs em Laravel
- Saber fazer requisições autenticadas para Web APIs

- [ ] **PHP - XDebug**:

- Xdebug é uma extensão PHP que fornece recursos de depuração e criação de perfil.
- Usar e aprimorar o var_dump com XDebug
- Analisar e entender a stack (pilha) de execução

- Depurar o código
- Depurar remotamente com Docker

Nivel 3

- [ ] **PHP Assíncrono**:
  - Na programação assíncrona as funções não são executadas em ordem. Com o assincronismo, podemos interromper do código para conseguirmos alguma outra informação necessária para a continuar a execução. Isso significa que o código espera por uma outra parte do código e, enquanto espera, executa as demais partes.
  - Aprender a usar sockets
  - Conhecer bibliotecas de programação reativa
  - Aprender sobre corrotinas
- [ ] **PHP Funcional**:
  - A programação funcional é um paradigma de programação baseado no uso de funções. As funções estão, de acordo com este paradigma, no epicentro de uma base de código.
  - Entender a fundo sobre funções em PHP
  - Conhecer funções de manipulação de arrays
  - Realizar composição de funções usando pipelines
- [ ] **Object Calisthenics**:
  - Object Calisthenics são basicamente exercícios de programação, como um conjunto de 9 regras para ajudar a escrever melhor o código orientado a objetos.
  - Conhecer as regras de Object Calisthenics
  - Simplificar classes e métodos
  - Manter o código mais curto e coeso
- [ ] **Arquitetura de Microsserviços**:

- Microsserviços são uma abordagem de arquitetura na qual o software consiste de pequenos serviços independentes que se comunicam entre si e são organizados de acordo com seus domínios de negócio.
- Aprender o conceito de arquitetura planejada para microsserviços
- Realizar a comunicação usando APIs
- Melhorar a escalabilidade de um sistema

☐ **Contêineres**:

- Os contêineres são pacotes de software que contêm todos os elementos necessários para serem executados em qualquer ambiente. Gerenciamento de contêineres é uma área crucial na computação em nuvem e DevOps, que envolve o uso de tecnologias para automatizar o processo de criação, implantação, escalonamento e monitoramento de contêineres. Contêineres são unidades de software padronizadas que permitem aos desenvolvedores empacotar todas as dependências de um aplicativo (código, bibliotecas, configurações, etc.) em um único pacote. Isso permite que o aplicativo seja executado de forma consistente em qualquer ambiente de infraestrutura.
- A tecnologia de contêineres, como exemplificada pelo Docker, fornece um ambiente consistente e portátil para desenvolvimento, teste e implantação de aplicativos, o que é vital para o trabalho eficiente de engenharia de dados. Além disso, o Kubernetes, um sistema de orquestração de contêineres, permite o gerenciamento, a automação e a escalabilidade de aplicações baseadas em contêineres em ambientes de produção. Dominar esses conceitos e tecnologias possibilita a engenheiros de dados construir e manter pipelines de dados eficientes e confiáveis.
- O Kubernetes (também conhecido como k8s ou kube) é uma plataforma de orquestração de containers open source que automatiza grande parte dos processos manuais necessários para implantar, gerenciar e escalar aplicações em containers.
- Isolar seu software para funcionar independentemente
- Implantar software em clusters
- Modularizar seu sistema em pacotes menores
- Conhecer a plataforma Docker

- Conhecer Kubernetes

- [ ] **Reflection e atributos**:

- Os objetos de Reflection (reflexão) são usados para obter informações do tipo em tempo de execução. As classes que dão acesso aos metadados de um programa em execução estão no namespace System.Reflection.
- Escrever código que lê as informações e metadados de objetos em tempo de execução
- Obter nomes de classes em tempo de execução e criar objetos de uma classe

Habilidade Auxiliar: Infraestrutura e boas práticas

- [ ] **Git e GitHub - Fundamentos**:

- Git é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com tudo, desde projetos pequenos a muito grandes com velocidade e eficiência.
- GitHub é um serviço de hospedagem para desenvolvimento de software e controle de versão usando Git.
- Criar um repositório
- Clonar um repositório
- Fazer commit, push e pull de e para o repositório
- Reverter um commit
- Criar branches e pul requests
- Lidar com merge e conflitos

- [ ] **HTTP - Fundamentos**:

- HTTP significa Hyper Text Transfer Protocol. A comunicação entre computadores cliente e servidores web é feita enviando solicitações HTTP e recebendo respostas HTTP.
- Entender a diferença dos verbos HTTP

- Testar os requests e ver os status codes no navegador
- Saber fazer uma requisição HTTP na linha de comando com WGET
- Baixar uma imagem com WGET
- Fazer um post

☐ **JSON**:

- JSON significa JavaScript Object Notation (notação de objeto JavaScript). É um formato de texto para armazenar e transmitir dados.
- Criar um objeto
- Transformar um objeto em uma string
- Transformar uma string em objeto
- Manipular um objeto

☐ **Linha de comando - Fundamentos**:

- CLI é um programa de linha de comando que aceita entradas de texto para executar funções do sistema operacional.
- Conhecer os principais comandos

☐ **Cloud - Fundamentos**:

- Cloud, ou computação em nuvem é a distribuição de serviços de computação pela Internet usando um modelo de preço pago conforme o uso. Uma nuvem é composta de vários recursos de computação, que abrangem desde os próprios computadores (ou instâncias, na terminologia de nuvem) até redes, armazenamento, bancos de dados e o que estiver em torno deles. Ou seja, tudo o que normalmente é necessário para montar o equivalente a uma sala de servidores, ou mesmo um data center completo, estará pronto para ser utilizado, configurado e executado.
- Conhecer a diferença entre IaaS, PaaS e SaaS
- Conhecer os maiores provedores de cloud
- Especializar-se em algum provedor

☐ **SQL - Fundamentos**:

- SQL (Structured Query Language, traduzindo, Linguagem de Consulta Estruturada) é uma linguagem de programação padronizada que é usada para gerenciar bancos de dados relacionais e realizar várias operações sobre os dados neles contidos.
- Conhecer os comandos mais comuns do SQL
- Usar SELECT para consultar uma tabela
- Usar INSERT para inserir dados em uma tabela
- Usar UPDATE para atualizar uma tabela
- Usar DELETE para remover dados de uma tabela
- Usar JOIN para conectar os dados de múltiplas tabelas
- Conhecer as cláusulas (FROM, ORDER BY, etc)

☐ **SOLID**:

- O Solid possui cinco princípios considerados como boas práticas no desenvolvimento de software que ajudam os programadores a escrever os códigos mais limpos, separando as responsabilidades, diminuindo acoplamentos, facilitando na refatoração e estimulando o reaproveitamento do código.

☐ **Design Patterns**:

- Na engenharia de software, um "padrão de projeto" (Design Pattern em inglês) é uma solução geral e reutilizável para um problema que ocorre normalmente dentro de um determinado contexto de projeto de software.
- Conhecer e aplicar os principais Design Patterns

☐ **Clean Architecture**:

- A Clean Architecture (Arquitetura Limpa) é uma forma de desenvolver software, de tal forma que apenas olhando para o código fonte de um programa, você deve ser capaz de dizer o que o programa faz.

☐ **Conceitos de Design Orientado a Domínio (Domain-Driven Design - DDD)**:

- O Design Orientado a Domínio (DDD) é uma abordagem ao projeto e desenvolvimento de software que é primeiramente informado pelos

requisitos de negócios. Os componentes do programa (objetos, classes, matrizes, etc.) indicam a indústria, setor ou domínio empresarial em que o negócio opera.

- Modelar domínios de forma efetiva
- Basear projetos complexos em modelos do domínio
- Conhecer os blocos de construção de DDD

Habilidade Auxiliar: Front-end

- [ ] **HTML - Fundamentos**:

- HTML é uma linguagem de marcação que define a estrutura do seu conteúdo. HTML consiste em uma série de elementos que você usa para mostrar algo de uma determinada maneira ou agir de uma certo modo. As tags podem criar um hiperlink de uma palavra ou imagem para outro lugar, podem colocar palavras em itálico, podem aumentar ou diminuir a fonte e assim por diante.
- Aprender quais tags são necessárias para um HTML básico
- Criar um parágrafo de texto
- Exibir uma imagem
- Conhecer a diferença entre 'h1', 'h2', 'h3', etc
- Criar um texto com hyperlink
- Criar um formulário com campos relevantes
- Criar uma lista de itens ordenada ou não ordenada
- Criar uma lista de itens dentro de uma lista suspensa (dropdown list)
- Conectar com um arquivo de CSS
- Criar uma tabela
- Adicionar IDs e classes

- [ ] **CSS - Fundamentos**:

- Cascading Style Sheets (CSS) é uma linguagem usada para descrever a apresentação de um documento escrito em uma linguagem de marcação

como HTML ou XML. CSS pode ser usado para estilos de texto de documentos muito básicos — por exemplo, para alterar a cor e o tamanho de títulos e links. Ele pode ser usado para criar um layout — por exemplo, transformar uma única coluna de texto em um layout com uma área de conteúdo principal e uma barra lateral para informações relacionadas. Pode até ser usado para efeitos como animações.

- Aprender a estrutura visual de uma página, com 'margin' e 'padding'
- Estabelecer o tamanho com 'width' e 'height'
- Aprender sobre a posição de um elemento ('static', 'relative' ou 'absolute)
- Aprender sobre o 'display' de exibição de um elemento ('block', 'inline', 'inline-block')
- Aprender a posicionar imagens em relação ao texto
- Aprender sobre alinhamento
- Aprender sobre estilo de fontes
- Aprender as diferenças e vantagens de usar as diferentes unidades de medida em CSS (%, relativas, etc)
- Conectar com os elementos (IDs, classes) de um arquivo HTML
- Alterar características de um elemento quando o mouse passar por cima dele ('hover')
- Aprender box-sizing
- Aprender Flexbox
- Aprender Grid

- [ ] **JavaScript - Fundamentos**:

- JavaScript é a linguagem de programação mais popular do mundo e é uma das principais tecnologias da World Wide Web, juntamente com HTML e CSS. Ela possui tipagem dinâmica, orientação a objetos baseada em protótipos e funções de primeira classe. Ela é multi-paradigma e suporta estilos de programação orientados a eventos, funcionais e imperativos.
- Conhecer os tipos primitivos
- Declarar variáveis, considerando a diferença entre 'var', 'let' e 'const'

- Usar estruturas condicionais ('if', 'else')
- Conhecer os operadores de atribuição e comparação ('=', '==', '===')
- Usar estruturas de repetição e laços ('while', 'for')
- Usar funções, passando parâmetros e argumentos
- Manipular arrays e listas
- Aprender o conceito de Orientação a Objetos
- Fazer um CRUD
- Obter dados de uma API
- Fazer chamadas assíncronas usando 'Async/Await', 'Promise', etc

---