

# Guia Ágil

---

TechGuide - Alura

---

## Front-end Angular

---

### Nível 1

#### ☐ 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

## ☐ **TypeScript - Fundamentos:**

- TypeScript é uma linguagem de programação fortemente tipada que se baseia em JavaScript.
- Entender a fundo o que são tipos e a importância da tipagem
- Aprender o que é o TypeScript, por que foi criado, como ele funciona e sua relação com o JavaScript
- Conhecer as ferramentas do TypeScript (integração com o editor de código, verificador estático e compilador)
- Escrever código em TypeScript com suas ferramentas (interfaces, enum, decorators, etc)
- Desenvolver aplicações em TypeScript

## ☐ **RxJS - Fundamentos:**

- Criar programas assíncronos
- Criar programas baseados em eventos
- Entender o conceito de observables e sequências de observables
- Entender como usar Observers, Subscription, Subject

## ☐ **Angular - Fundamentos:**

- Construir interfaces utilizando HTML, CSS e TypeScript
- Criar aplicações SPA
- Construir aplicações web, mobile ou desktop
- Integrar dados com API's REST
- Utilizar a composição para criar componentes reutilizáveis
- Utilizar serviços do tipo Resolver
- Manipular requisições criando serviços do tipo Interceptor

## ☐ **O Pattern Observer:**

- Entender o que são Design Patterns
- Atualizar diversos elementos simultaneamente

- Declarar os Subjects
- Criar programas baseados em eventos

## Nível 2

### ☐ Angular - Templates:

- Como usar os templates

### ☐ Angular - Renderização:

- Exibir uma página Angular no navegador
- Realizar renderização no lado do servidor

### ☐ Angular - Services:

- Criar dados que estarão sempre disponíveis
- Dividir a aplicação web em diversas partes

### ☐ Angular - Roteamento:

- Navegar até um componente
- Incluir um parâmetro de rota
- Controlar o fluxo de navegação do seu usuário com guarda de rotas

### ☐ Angular - CLI (Interface de Linha de Comando):

- Inicializar, desenvolver e manter aplicações Angular a partir da interface de linha de comando

## Nível 3

### ☐ Angular - Gerenciamento de Estado:

- Atualizar componentes em tempo real
- Esperar por alterações em algum componente e executar alterações

### ☐ Angular - Formulários:

- Criar formulários com Template Forms
- Criar formulários com reativos com Reactive Forms

#### ☐ **Angular - Módulos:**

- Registrar componentes
- Declarar quais componentes podem ser usados por componentes de outros módulos
- Declarar quais services serão injetados
- Aprender a modularizar uma aplicação

#### ☐ **Angular - Injeção de Dependências:**

- Declarar as dependências de suas classes
- Injetar dependências em um componente

#### ☐ **Angular - 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

#### ☐ **GraphQL:**

- GraphQL é uma linguagem de consulta e manipulação de dados de código aberto para APIs. É considerada uma alternativa para arquiteturas REST.
- Aprender o que é GraphQL e por que foi criado
- Entender como o GraphQL é utilizado no desenvolvimento de APIs
- Criar APIs utilizando as bibliotecas e frameworks para GraphQL

#### ☐ **Apollo Client:**

- Apollo Client é uma biblioteca abrangente de gerenciamento de estado para JavaScript que permite gerenciar dados locais e remotos com o GraphQL.
- Utilizar o Apollo para criar um servidor GraphQL
- Conectar com uma API

## Habilidade Auxiliar: Infraestrutura e Back-end

### ☐ 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 pull 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

#### ☐ **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

#### ☐ **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

# Habilidade Auxiliar: UX e Design

## ☐ Design Systems:

- Um Design Systems (sistema de design) é uma coleção de componentes reutilizáveis, guiados por padrões claros, que podem ser colocados juntos para construir aplicações.
- Criar e manter bibliotecas que serão consumidas e usadas como padrão para a construção de um projeto

## ☐ Figma - Fundamentos:

- Figma é uma aplicação web colaborativa para design de interfaces. O conjunto de recursos do Figma se concentra na interface do usuário e no design da experiência do usuário, com ênfase na colaboração em tempo real, utilizando uma variedade de editores de gráficos vetoriais e ferramentas de prototipagem.
- Criar layouts de páginas e componentes

## ☐ Componentes de design:

- Conhecer os componentes descrevem um layout ou interface

## ☐ Sistemas de cores:

- Definir uma paleta de cores que faça sentido para determinada interface

## ☐ Como usar fontes:

- Escolher a fonte mais adequada para determinado projeto