


Front-End do zero

@iuricode

Índice

Esse ebook tem como objetivo principal apresentar e ensinar o básico da área front-end de uma maneira completa e acessível para todos.

HTML Fundamentos

O que é HTML?

HTML, ou Hypertext Markup Language é uma linguagem de marcação (não de programação) da web - cada vez que você carrega uma página da web, você está carregando um código HTML. Pense em HTML como o esqueleto de uma página da web, ele é responsável pelos textos, links, listas e imagens - ele oferece conteúdos (enquanto javascript fornece comportamento dinâmicos e o css estilos).

Iniciando

HTML é escrito em arquivos .html. Para criar uma página HTML é fácil, entre em seu editor de código e salva em arquivo em branco como meu-site.html (você pode nomeá-lo como quiser).

Sintaxe

Os elementos HTML são escritos usando tags. Todas as tags tem uma chave de abertura e fechamento (por exemplo, <tag>) e uma tag de fechamento que tem uma barra após o primeiro colchete (por exemplo, </tag>).

```
<tag>
 iuricode
</tag>
```

Por exemplo, se você deseja criar um parágrafo, usaremos as chaves de abertura e fechamento :

```
<p>
 Um programador sem café é um poeta sem poesia.
</p>
```

Os elementos podem entrar dentro de outros elementos:

```
<pai>
   <filho>
    Esta tag está dentro de outra tag também
    conhecida como a tag pai.
   </filho>
 </pai>
```

Estrutura de uma página HTML

A estrutura inicial do seu html é essa:

```
<! DOCTYPE html>
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

Vou explicar sobre elas

- <! DOCTYPE html> - Essa tag (não tem fechamento dela) informa ao seu navegador que o arquivo faz parte de um documento HTML5.

- <html> - Essa é tag que seu código todo vai fichar dentro <html> seu código aqui </html>.

- <head> - O head contém informações sobre seu site, mas não é o conteúdo que vai aparecer na sua página. Ela conterá coisas como: links para folhas de estilo (CSS), título pa sua página, link de fontes e tudo aquilo que você quiser linkar (só não recomendo link seu script dentro dessa tag).

- <body> - No body contém todo o conteúdo que vai aparecer na sua página. Todo o código que você escrever irá dentro dele.

Indentação adequada

As quebras de linha entre suas tags são super importante para escrever um bom código HTML.

```
<! DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <h1> Aqui temos um título </h1>
    <p> E aí, já curtiu esse ebook? </p>
  </body>
</html>
```

Abaixo temos um indentação não recomendada:

```
<! DOCTYPE html> <html> <head> </head> <body>
<h1> Não faça isso! </h1> </body> </html>
```

Tag

Agora vamos apresentar as principais tags do HTML.

Tags de títulos

Os elementos de cabeçalho H representam seis níveis de título de seção. <h1> é o nível de seção mais alto e <h6> é o mais baixo. Quanto maior for o nível do H menor vai ser o tamanho de pontos da fonte. Vejamos:

```
<h1> tamanho 24 pontos </h1>
<h2> tamanho 18 pontos </h2>
<h3> tamanho 14 pontos </h3>
<h4> tamanho 12 pontos </h4>
<h5> tamanho 10 pontos </h5>
<h6> tamanho 8 pontos </h6>
```

Ficara assim na página:

tamanho 24 pontos
tamanho 18 pontos
tamanho 14 pontos
tamanho 12 pontos
tamanho 10 pontos
tamanho 8 pontos

Tags para texto

As tags de texto, definem diferentes formatações para diversos tipos de texto. Desde estilos de fonte, parágrafos, quebra de linha ou até mesmo spans. Enfim iremos conhecê-las:

 - Sendo a principal tag de texto, é usada para constituir um paragráfo. Exemplo:

```
<p>Este é o primeiro parágrafo do texto.</p>
```

 - Mesmo tendo a sua funcionalidade parecida com o uso dos parágrafos. Spans são geralmente utilizados apenas para omitir uma tag ou guardar uma pequena informação como uma legenda de um formulário. Exemplo:

```
<p><span>Um texto</span></p>
```

<pre></pre> - É a tag utilizada para representar texto recém-formatado. Um texto dentro desse elemento é tipicamente exibido em uma fonte não proporcional da mesma maneira em que o texto original foi disposto no arquivo. Espaços em branco são mantidos no texto da mesma forma em que este foi digitado. Exemplo:

```
<pre>
        “As pessoas podem ser realmente
            inteligentes ou ter
        habilidades diretamente aplicáveis,
        mas se elas não acreditarem nisso,
            elas não irão dar duro.”
            - mark zuckerberg
</pre>
```

 - Transforma e deixa o seu conteúdo em formato negrito. Exemplo:

```
<b>Texto em Negrito</b>
```

 - Transforma o seu conteúdo em formato itálico. Exemplo:

```
<i>Texto em Itálico</i>
```

<hr> - Essa tag não necessita de fechamento, ela forma uma linha horizontal, sendo também usada no final não sendo uma regra obrigatória. Exemplo:

```
<p>Este é o primeiro parágrafo do texto.</p>
<hr>
<p>Este é o segundo parágrafo do texto.</p>
```

Tags de imagem

Vamos colocar uma imagem no seu site?
É bem simples colocar uma imagem no HTML. Vejamos:

```
<img src="nomeDoArquivo.formato">
```

Você talvez deve ter se perguntado o que é esse src, src (source) é atributo da tag <img>, nele vai conter a url da imagem que será inserida por você.

Alguns pontos importantes:

- A tag img não tem a chave de fechamento.
- Se você tiver com a imagem de uma pasta deve colocar o caminho dela dentro do src.
- Recomendamos que você coloque o atributo alt para que pessoas com deficiência saberem do que se trata a imagem na página.

```
<img src="nomeDaPasta/nomeDoArquivo.formato"
alt="descrição sobre a imagem">
```

Tags de lista

Vamos falar agora um pouco sobre as listas (ordenadas e desordenadas) e como elas funcionam no HTML. As listas são muito importantes quando queremos listar alguns itens no site e também para a criação de menu de navegação.

- Listas ordenadas.
- Listas desordenadas.

Listas ordenadas

```
<ol>
  <li>html</li>
  <li>css</li>
  <li>javascript</li>
  <li>front-end</li>
</ol>
```

Aparecerá assim no site:

1. html
2. css
3. javascript
4. front-end

Caso você queria deixar em ordem alfabética é simples, coloque o atributo type="a" dentro da tag <ol>.

```
<ol type="a">
  <li>html</li>
  <li>css</li>
  <li>javascript</li>
  <li>front-end</li>
</ol>
```

Aparecerá assim no site:

a. html
b. css
c. javascript
d. front-end

Você também pode deixar com algarismos romanos.

```
<ol type="I">
  <li>html</li>
  <li>css</li>
  <li>javascript</li>
  <li>front-end</li>
</ol>
```

Aparecerá assim no site:

I. html
II. css
III. javascript
IV. front-end

Listas desordenadas

As listas desordenadas seguem o mesmo padrão apenas mudando de `<ol>` para `<ul>`. Em seu visual ele mudara para pontos (na lateral dos itens) no lugar de números.

```
<ul>
  <li>html</li>
  <li>css</li>
  <li>javascript</li>
  <li>front-end</li>
</ul>
```

Aparecerá assim no site:

- html
- css
- javascript
- front-end

Tags para formulário

Essa seção vamos mostrar como montar seu formulário. Formulários HTML são um dos principais pontos de interação entre um usuário e um web site. Um formulário HTML é feito de um ou mais widgets. Esses widgets podem ser campos de texto (linha única ou de várias linhas), caixas de seleção, botões, checkboxes ou radio buttons.

Para construir o nosso formulário de contato, vamos utilizar os seguintes elementos <form> , <label> , <input> , <textarea> e <button>.

Elemento Form

Todos formulários HTML começam com um elemento <form> como este:

```
<form action="/pagina-processa-dados-do-form" method="post">

</form>
```

Todos os seus atributos são opcionais, mas é considerada a melhor prática sempre definir pelo menos o atributo action e o atributo method.

- O atributo action define o local (uma URL) em que os dados recolhidos do formulário devem ser enviados.

- O atributo method define qual o método HTTP para enviar os dados (ele pode ser "GET" ou "POST" (veja as diferenças aqui).

Elementos Label, Input e Textarea

O nosso formulário de contato é muito simples e contém três campos de texto, cada um com uma etiqueta. O campo de entrada para o nome será um campo básico texto de linha única ("input"); o campo de entrada do e-mail será um campo de texto com uma única linha ("input") que vai aceitar apenas um endereço de e-mail; o campo de entrada para a mensagem será um campo de texto de várias linhas ("textarea").

Em termos de código HTML, teremos algo assim:

```
<form action="/pagina-processa-dados-do-form"
method="post">
    <div>
        <label for="nome">Nome:</label>
        <input type="text" id="nome" />
    </div>
    <div>
        <label for="email">E-mail:</label>
        <input type="email" id="email" />
    </div>
    <div>
        <label for="msg">Mensagem:</label>
        <textarea id="msg"></textarea>
    </div>
</form>
```

Algumas observações:

- Observe o uso do atributo for em todos os elementos label; é uma maneira para vincular uma label à um campo do formulário.

- No input temos o atributo type. Esse atributo ele defina a forma que nosso input se comporta. Exemplo: type="text" (é o valor padrão para este atributo), o type="email" ele define que o campo aceita só endereço de e-mail.

- Por último, mas não menos importante, a tag <textarea>. O tipo textarea não é um elemento de auto-fechamento, então você tem que fechá-lo com a tag final adequada.

Elemento Button

Pronto! O nosso formulário está quase pronto; nós temos apenas que adicionar um botão para permitir que o usuário envie seus dados depois de ter preenchido o formulário. Isto é simplesmente feito usando o elemento button:

```
<div class="button">
  <button type="submit">Enviar sua
mensagem</button>
</div>
```

Algumas observações:

- Um clique sobre um botão de submit envia os dados do formulário para a página de web definida pelo atributo action do elemento <form>.

- Um clique sobre um botão de reset redefine imediatamente todos os campos do formulário para o seu valor padrão.

- Um clique em um botão do tipo button faz ...ops, nada! Isso soa bobo, mas é incrivelmente útil para construir botões personalizados com JavaScript, ou seja, ele pode assumir qualquer comportamento através desta linguagem.

Resultado final

```
<form action="/pagina-processa-dados-do-form"
method="post">
    <div>
        <label for="nome">Nome:</label>
        <input type="text" id="nome" />
    </div>
    <div>
        <label for="email">E-mail:</label>
        <input type="email" id="email" />
    </div>
    <div>
        <label for="msg">Mensagem:</label>
        <textarea id="msg"></textarea>
    </div>
    <div class="button">
  <button type="submit">Enviar sua
mensagem</button>
</div>
</form>
```

Última observação

Você pode ter percebi que no nosso formulário usamos a tag div. A tag div não tem nenhum efeito em nosso formulário, como ela tem um propriedade display block como padrão, o uso dela ajuda a quebrar os elementos em linhas, deixando melhor a visualização.

Tags semânticas

Na vida sempre temos uma forma correta de fazer as coisas, no HTML não é diferente! As tags semânticas além de deixar o código melhor para o SEO dos navegadores, ela ajuda outros desenvolvedores a entender seu código só batendo o olho, isso é ótimo para que outros desenvolvedores contribuem em seus projetos. Uma observação: as tags semânticas não tem nenhum efeito na apresentação na página da web.

- Elementos semânticos: tem significado e deixam seu conteúdo claro `<form>`, `<table>`, `<article>`, `<footer>` e `<section>`*.

- Elementos não semânticos: não deixam seu conteúdo claro `<div>` e `<span>`.

Exemplo:

Veja que div é amplo, mas footer dá significado (que é o rodapé). Portanto ao invés de

```
<div id="footer">
```

não seria melhor:

```
<footer>
```

As principais tags semânticas

<article>: expressa um elemento independente, ou seja, que possa ser lido e interpretado sem depender do resto da página. Um bom exemplo é uma notícia de jornal ou uma pergunta no Stackoverflow.

<aside>: por conceito, expressa um conteúdo a parte do conteúdo da página. Na prática, pense em um box de informações dentro de uma página.

<datails>: representa detalhes adicionais que o usuário pode mostrar ou esconder.

<figcaption>: representa uma legenda para um elemento <figure>.

<figure>: representa um conteúdo independente, como ilustrações, diagramas, etc...

<footer>: representa o rodapé de um documento ou section.

<header>: representa o cabeçalho de um documento ou section.

<main>: representa o conteúdo principal de um documento.

<mark>: representa um texto destacado.

<nav>: representa links de navegação.

<section>: representa uma seção dentro de um documento.

<summary>: representa um cabeçalho para um elemento details.

<time>: representa uma data/hora.

CSS Fundamentos

O que é CSS?

CSS ou folhas de estilo em cascata é a linguagens de marcação (não de programação) responsável por adicionar estilos nos sistema web, como cores, tamanhos, posicionamentos. Sem ele, os sites são apenas um monte de texto e links, html acaba virando um Markdown.

Aplicando estilo CSS em uma página HTML

Vamos por etapas:

Etapa 1: Criar um arquivo CSS

A primeira coisa que precisamos fazer é criar um arquivo css do qual nossa página html possa obter seu estilo. Então em seu editor de código crie um novo arquivo chamado style.css.

Etapa 2: Linkar seu CSS no HTML

Precisamos conectar nossa página style.css à página HTML. Dentro da tag <head> de sua página html, vamos adicionar uma tag <link> para conectar a nossa nova folha de estilo.

```
<link rel = "stylesheet" href = "style.css">
```

certifique-se de que sua página html e sua folha de estilo css estão no mesmo nível de pasta, caso ele esteja dentro de uma pasta é simples chamar, basta colocar o nome da página antes do nome do arquivo seperado ele com uma barra (/)

```
<link rel = "stylesheet" href =
"nomeDaPasta/style.css">
```

Etapa 3: Selecione e dê estilo aos elementos. por elemento

Experimente selecionar um elemento e estilizá-lo! Em seu tipo de folha de estilo:

```
h1 {
  color: #00f;
  font-size: 25px;
}
```

Isso para que todos os elementos h1 da sua página fique com um tamanho 25px e com a cor azul. (atualize a página sempre que aplicar um CSS ou HTML)

Folha de Estilos

O que constitui uma folha de estilos? Bem, basicamente uma folha de estilos é algo simples, descrevem um conjunto de regras no css onde apresenta tags para o leitor identificar de forma mais explícita referências ou algum recurso em específico, aplicando estilos. Iremos usar serão definidos dentro do par de tags <head>...</head> como será mostrado a seguir.

```
<style type="text/css" title="mystyles" media="all">
</style>
```

Iremos a explicação básica dos elementos citados acima:

type="text/css - Até aqui nada em especial, isto diz ao navegador que estamos usando um tipo de arquivo puro para descrever os estilos que iremos usar.

title=Meuestilo - Este em especial é algo de sua livre escolha para nomear e mais tarde identificar os estilos presentes, pode conter qualquer nome.

media=all - Quando chega aqui nestas palavras as coisas já começam a ficar interessantes pois começam a se tratar de elementos responsivos para ajudar os "bots" dos navegadores a encontrarem o seu conteúdo no meio de milhares de sites. A palavra media significa que você pode ter livre-arbítrio para escolher o que sua página irá apresentar na tela do navegador do usuário, adaptando cores, altura e largura citamos alguns exemplos:

media=print - Foi designada para o layout de impressão em papel.

media=tv - Designada para televisores.

media=all - Todos os dispositivos.

media=embossed - Para dispositivos que imprimem em braile entre outros.

Aplicando Cor de Fundo

Agora iremos aos estilos, a primeira coisa que iremos realizar é a abertura da tag <body>...</body>. De forma resumida tudo que estiver dentro desta tag irá afetar os elementos da página de forma geral. A definição de body é seguida com um par de chaves.

```
body {
  ...
}
```

Os navegadores por padrão processam as páginas na cor branca e textos na cor escura, primeiramente dentro do body mesmo iremos realizar alterações no padrão de cor de fundo.

```
body {
  background color: #4169E1;
}
```

Repare que eu usei o elemento "#" na hora de definir uma cor, isto acontece pois o css utiliza o formato de cores com o código hexadecimal, o nome das cores em inglês e também o valor RGB (Red, Green, Blue) para definir o seu estilo.

Aplicando cor ao texto

Da mesma maneira podemos definir partir da tag "color", cores ao texto tirando o padrão preto dele de antigamente.

```
body {
  background color: #5d665b;
  color: #5d665b;
}
```

Observação: Da mesma maneira que eu coloquei na cor de fundo, nunca esqueça de incrementar o ponto vírgula no final, pois no css a pontuação é algo fundamental podendo causar até erros

Aplicando a margem

Também podemos aplicar a famosa margem muita usada principalmente por mim para alinhar e acertar o posicionamento desejado de seu texto

```
body {
  background color: #5d665b;
  color: #5d665b;
  margin: 50px;
}
```

Class e ID

Class e ID são tipos de atributos que podemos adicionar a um elemento para ser mais específicos com nossos seletores CSS. Até agora, só pudemos selecionar elementos (como <h1>). Embora isso seja útil, às vezes precisamos ser mais específicos. Exemplo, temos várias tags <p> em uma página, mas temos um <p> que queremos mudar de cor. E nesses casos que entrar a Class e o ID para selecionar apenas um elemento. Exemplo:

```
<p class = "souClass">
```

```
<h1 id = "souID">
```

Chamando no CSS

Para fazemos referência a uma classe usando um . e o nome da classe, exemplo:

```
.souClass {
  color: # f00;
}
```

Fazemos referência a um id usando um # e o nome do id:

```
#souID principal {
  color: # f00;
}
```

Dessa forma, todos os elementos que tiverem a Class .souClass e o ID #souID vão ficar com a cor de texto de vermelho.

E qual usar?

Agora te bateu uma dúvida né? Se os dois fazem a mesma coisa, qual devo usar?

A resposta é simples, se você tem vários elementos (no nosso caso é o <p>) e queira usar de cor de texto vermelha em algumas, nesse caso usamos a Class, pense a Class em uma união em comum. Vou dar um exemplo que meu professor disse na faculdade, todos nós tem uma Class em como, que no caso é humano porém, todos nós temos um ID, que no caso é o RG. Então caso queira mudar somente em um elementos, usamos o ID.

Bônus

Aqui vou dizer dois padrões que eu uso sempre no começo do meu CSS, o primeiro é o uso da fonte e o segundo é o reset nos padrões dos navegadores.

Fontes

Para colocar fonte no seu site é preciso chamar ele no seu CSS (ou no HTML), para isso vamos entrar no site do Google Fonts. Logo em seguida, procure uma fonte que você deseja e clique no botões "Select this style" para assim adicionar os estilos que você deseja (perceba que quando você selecionar uma vai abrir uma aba na lateral direita). Nessa aba vai ter duas opções para colocar no seu site, um é o <link> e o outro @import. O link é para importar no seu HTML e o import é para o seu CSS, agora você me pergunta qual eu devo usar?, você pode escolher qualquer um, isso vai com seu gosto. Como para mim a fonte é um estilo, então eu sempre importo ele para meu CSS.

Logo depois de importar precisamos falar para nossa página que queremos aquela fonte em nossos textos e como fazer isso? É bem simples!

Perceba de logo depois do import (na página do Google Fonts), em baixo tem uma código CSS com o nome font-family, é exatamente ele que vamos usar para colocar para dizer que aquela vai ser a fonte da nossa página. No seu CSS, coloque ela.

```
*{
  font-family: 'Nome da sua fonte aqui';
}
```

Uma observação, se você está se perguntando o que é esse * no CSS, ele é um elemento universal, quer dizer que vai aplicada em toda sua página de estilo

Reset

Agora vamos resetar os navegadores, mas espera, o quer isso quer dizer? Quer dizer que os navegadores tem um padrão deles, e alguns desses padrões não são legais, então vamos tirar alguns deles (três para falar a verdade)!

Primeiro vamos zerar os espaçamentos das nossas páginas. Quando criamos um HTML, por padrão, nosso site tem um espaçamentos e quando criamos algo, nossos estilos não fica da forma que queremos, eles acabam sempre tento um espaço em volta dele. Para tirar esse padrão usamos duas coisas. Um é o padding (espaçamento dentro do conteúdo) e o margin (espaçamento fora do conteúdo). Veja o exemplo:

```
*{
  font-family: 'Nome da sua fonte aqui';
  padding: 0;
  margin: 0;
}
```

E por último, vamos aplicar o box-sizing: border-box; no nosso CSS. Mas você como é um gafanhoto curioso vai ser perguntar o que ele faz? e por que devo usar ele?, quando criamos uma caixa algo que use largura e altura, sempre definimos o valor delas (no nosso exemplo vai ser 300px e cada um). Mas caso você queira colocar um elemento dentro dessa tag que você definiu a largura e altura, e coloque um espaçamentos dentro dela (padding) de 30px, o elemento vai deixar de ser 300px e será 330px.

Mas a gente não quer isso, não é? Queremos que o elemento continue 300px (na altura e largura) porém, com um espaçamento dentro dela. É aí que entra o nosso querido box-sizing: border-box;. Veja o exemplo:

```
*{
  font-family: 'Nome da sua fonte aqui';
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}
```

Caso queira testar com e sem o box-sizing: border-box;

Aplique no seu CSS e no HTML:

sem

```
*{
  font-family: 'Nome da sua fonte aqui';
  padding: 0;
  margin: 0;
}
#caixa{
  width: 300px;
  height: 300px;
  background-color: #ddd;
  padding:30px;
}
```

```
<div id="caixa">
<p>iuricode</p>
</div>
```

com (vamos apenas colocar o box-sizing: border-box; no nosso elemento universal)

```
*{
  font-family: 'Nome da sua fonte aqui';
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}
#caixa{
  width: 300px;
  height: 300px;
  background-color: #ddd;
  padding:30px;
}
```

Roadmap Front-End

Roadmap é um roteiro que busca apresentar os caminhos que uma pessoa pode percorrer para alcançar os objetivos e metas traçadas.

Aprender front-end hoje se tornou cada vez mais importante para criação de boas e acessíveis interfaces web e mobile. Antes de se aventurar no mundo dos frameworks javascript, é cada vez mais importante construir uma base sólida e, depois disso, buscar formas de agilizar o seu trabalho com front.

Vou listar aqui, links importantes para qualquer iniciante que queira se aventurar nesse mundo. Aproveite como quiser.

Geral

Antes de mais nada, devemos nos contextualizar com aquilo que vamos aprender. Não é necessário para o mercado, mas é interessante que você saiba como tudo começou, quais eram as dificuldades que as pessoas enfrentavam em outras décadas para desenvolver para web e etc.

- **Como a internet funciona?**
- **O que é DNS?**
- **Protocolo HTTP**

- **A história do front-end**
- **Introdução a Algoritmos - Curso de Algoritmos #01**

HTML

HTML (Linguagem de Marcação de HiperTexto) é o bloco de construção mais básico da web. Define o significado e a estrutura do conteúdo da web. Outras tecnologias além do HTML geralmente são usadas para descrever a aparência/apresentação (CSS) ou a funcionalidade/comportamento (JavaScript) de uma página da web.

- **HTML - W3Schools**
- **HTML - MDN**
- **Curso de HTML**

CSS

CSS (Cascading Style Sheets ou Folhas de Estilo em Cascata) é uma linguagem de estilo usada para descrever a apresentação de um documento escrito em HTML ou em XML (incluindo várias linguagens em XML como SVG, MathML ou XHTML). O CSS descreve como elementos são mostrados na tela, no papel, na fala ou em outras mídias.

- **CSS - MDN**
- **CSS - W3Schools**
- **Display grid, tutorial da Origamid**
- **Display flex, tutorial da Origamid**
- **CSS Tricks**

JavaScript

JavaScript (às vezes abreviado para JS) é uma linguagem leve, interpretada e baseada em objetos com funções de primeira classe, mais conhecida como a linguagem de script para páginas Web, mas usada também em vários outros ambientes sem browser, tais como node.js, Apache CouchDB e Adobe Acrobat. O JavaScript é uma linguagem baseada em protótipos, multi-paradigma e dinâmica, suportando estilos de orientação a objetos, imperativos e declarativos (como por exemplo a programação funcional).

- **Javascript - MDN**
- **Javascript - W3Schools**
- **Curso de Javascript**

Entre no site da Rocketseat, faça um cadastro lá e acesse o starter. Lá você poderá aprender sobre:

- **Javascript**
- **ES6**
- **Node, React e React Native**

Prática

Sim, para você definitivamente aprender algo, você precisa praticar. Dê uma olhada aqui:

- **Desafio: 10 projetos rápidos para treinar Programação e conseguir um Emprego**
- **javascript-code-challenges: A collection of JavaScript modern interview code challenges for beginners to experts**
- **dev-practice: Practice your skills with these ideas.**
- **1000_Projects: Mega List of practical projects that one can solve in any programming language!**
- **app-ideas: A Collection of application ideas which can be used to improve your coding skills.**

Front-End do zero

@iuricode

GitHub: https://github.com/iuricode

Instagram: https://www.instagram.com/iuricode/