
E-BOOK

# PROGRAMADOR Full Stack Javascript

O Guia para você se tornar um Programador(a) Full Stack acima da média.

Leonardo Scorza • Lucas Queiroga
Isaac Pontes • Juliana Conde

onebitcode

**E-BOOK**

# PROGRAMADOR FULL STACK JAVASCRIPT

O guia para você se tornar um
Programador Full Stack acima da média

Leonardo Scorza • Lucas Queiroga
Isaac Pontes • Juliana Conde

onebitcode

# CAPÍTULOS

# O INÍCIO

***"O começo é a metade do todo"***
*Platão*

Bem vindo(a) Programador(a), você acaba de cruzar uma linha importante, você deu início a uma jornada incrível, onde junto comigo, você vai dar todos os passos até se tornar um Programador(a) Full Stack acima da média.

Se você está aqui, significa que você já sabe como o mundo da programação é interessante, cheio de oportunidades e que através dele você pode criar coisas realmente impressionantes, ganhar bem e trabalhar de onde quiser.

Através da parte em vídeo do treinamento Programador(a) Full Stack JavaScript e deste guia, eu vou te conduzir pelos tópicos mais importantes do mundo da Programação Web de forma progressiva.

Para que ao final dessa jornada você esteja apto(a) a aproveitar todas essas oportunidades e sinta-se extremamente orgulhoso(a).

## Nosso compromisso

Antes de apertarmos o Start, eu quero fazer um compromisso com você.

Nós vamos te oferecer:

- Aulas de alta qualidade
- Exercícios com Resolução
- Um guia incrível (este livro)
- Projetos completos e realistas
- Suporte de primeira

Mas você também precisa fazer a sua parte:

- Seguir as aulas com dedicação
- Realizar os exercícios e projetos
- Ler este guia com atenção
- Tirar suas dúvidas na comunidade
- Preparar seu portfólio como ensinamos

Seguindo esses passos, não tem erro, com certeza você vai ter sucesso nessa jornada e vai se tornar um Programador(a) Full Stack acima da média.

## O OneBitCode

Eu ainda não me apresentei oficialmente, eu sou o Leonardo Scorza, você pode me chamar de Léo, já que agora somos companheiros de jornada.

Eu sou programador(a) há 12 anos, já trabalhei para empresas do Brasil e do Exterior, já trabalhei como CLT, PJ e freelancer e tive a oportunidade de viajar (como nômade digital) para quase 20 países enquanto trabalhava online.

Eu conto mais sobre a minha história e sobre como é ser um Programador(a) nômade no nosso Livro Programador Nômade, como você é nosso(a) aluno(a), tem um desconto gigantesco nele, basta [clicar aqui].

Também tenho o orgulho de dizer que fundei o OneBitCode há 6 anos, junto com a Flávia Modesto (minha esposa).

Hoje o OneBitCode cresceu e temos um time incrível (somos quase 10 pessoas), prontos para te ajudar durante toda a sua jornada na Programação.

Eu também tenho a felicidade de saber que nesse tempo já ajudamos milhares de pessoas a terem sucesso na programação (e de certa forma na vida, já que a vida e o trabalho estão intimamente ligados).

Já recebemos milhares de depoimentos de alunos que conseguiram seu primeiro trabalho, ou que migraram para oportunidades melhores (no Brasil e no Exterior) ou que fundaram suas próprias empresas.

Ficamos realmente felizes em ter você no time do OneBitCode, seja bem vindo(a). Estamos animados para receber o seu depoimento de sucesso ao final do treinamento 🤘

## A metodologia C.O.D.E

O OneBitCode é diferente de tudo que existe no mercado porque nós sabemos que proporcionar uma experiência de aprendizado incrível, faz toda a diferença no sucesso do aluno.

Então baseado na nossa experiência ensinando programação há 6 anos, nós desenvolvemos uma metodologia própria de ensino chamada **C.O.D.E**.

### C = CLAREZA

**Passo a passo bem definido.**
Em todos nossos conteúdos, criamos uma base sólida e fácil de aprender para que o aluno compreenda o porquê ele está ali, o que ele vai conseguir desenvolver e onde ele vai conseguir chegar.

### O = ORIENTAÇÃO

**Grupo de alunos e instrutores para guiar o aprendizado.**
A partir do contato próximo e informal que possuímos através de nossa comunidade, permitimos que o aluno desenvolva habilidades técnicas e não técnicas e assim torne-se um programador completo.

### D = DOMÍNIO

**Realização de exercícios práticos com resolução e projetos realistas para o domínio completo do conteúdo.**
Após uma base sólida e uma comunidade forte, focamos na prática, pois acreditamos que somente o próprio aluno colocando a mão na massa, será

capaz de compreender seus erros e acertos e evoluir de verdade como programador, podendo inclusive, incrementar seu portfólio com esses projetos criados para obter as melhores oportunidades de job.

## E = ÊXITO

**Guia para que o aluno obtenha o êxito/sucesso em seus objetivos a partir do conteúdo aprendido.**
Após as 3 primeiras etapas, encerramos os conteúdos com módulos sobre carreira, indicando como o aluno pode conquistar a vaga desejada, o que escrever em seus perfis profissionais, como criar um bom GitHub, como se portar numa entrevista, etc., tudo para que além do conhecimento técnico, o aluno possa obter a confiança necessária para participar de entrevistas, enfrentar diversos desafios e obter sucesso em pouco tempo.

Esta é a metodologia usada neste treinamento, que vai guiar o seu progresso de aprendizado e te levar aos seus objetivos na programação.

Agora que você já entendeu como o meu compromisso com você é sério, como o OneBitCode realmente pode impulsionar a sua carreira e como a metodologia C.O.D.E vai te ajudar a dominar a programação de verdade, nós podemos dar Start tanto no livro quanto no treinamento.

Então bora começar a nossa jornada rumo a dominar o Full Stack JavaScript 🤘

# O TREINAMENTO

***"Tudo é possível desde que você dedique seu tempo, seu corpo e sua mente".***
*Michael Phelps*

Você, assim como eu, já deve ter assistido muitos filmes e/ou lido muitos livros, então você já sabe que antes do herói ou da heroína triunfarem, eles precisaram passar por um grande treinamento.

Esse é o seu grande treinamento na jornada Full Stack, abaixo eu vou te apresentar tudo que vai acontecer aqui para que você esteja ciente do poder de transformação desse treinamento.

## A entrega (Aulas + Guia + Comunidade)

Eu respeito verdadeiramente o seu tempo e as suas metas, por isso, esse treinamento é direto, eficiente e focado em trazer os melhores resultados para você.

Para atingirmos esses pontos, nós dividimos ele em três partes:

### 1 - As Aulas:

Nas aulas você vai ter os vídeos + textos de acompanhamentos (divididos por módulos), que vão te ensinar os conceitos detalhadamente, te propor exercícios com resolução e te guiar na criação de dois grandes projetos (falarei sobre eles na próxima seção).

### 2 - O livro:

O Objetivo deste livro é te dar uma visão geral sobre a carreira de programação, te incentivar na jornada, trazer uma lista de ferramentas e links fundamentais (de cada tecnologia ensinada no treinamento) e te ensinar algumas técnicas importantes, como o guia de sobrevivência aos Bugs (para você ter mais autonomia).

### 3 - A comunidade:

A comunidade é o complemento perfeito para os dois primeiros, nela você vai se comunicar com os outros alunos, com os instrutores e comigo.

Lá você pode tirar suas dúvidas, técnicas e não técnicas, pedir aconselhamento, criar uma rede de amigos(as) (que vai te manter mais motivado(a)) ou simplesmente bater um papo sobre programação e o mundo.

Como eu disse anteriormente, nós realmente estamos entregando uma experiência completa de aprendizado.

## O Plano (como seguir esse treinamento)

Uma das mais perigosas armadilhas para quem está se aventurando pelo mundo da Programação é a falta de clareza, e um dos aspectos mais intensos dela é: **Não saber o que e quando estudar**.

Nós desenvolvemos um plano de 8 semanas, onde dizemos exatamente quais aulas do treinamento você deve fazer a cada dia, para que em 8 semanas você consiga finalizar a parte principal do treinamento (tirando os extras).

Esse plano não é obrigatório, você pode fazer o treinamento no seu tempo, porém, eu recomendo fortemente que você siga ele, porque **ele vai servir como um guia e como estímulo para estudar todos os dias**.

Ele fica dentro do Dashboard que vou te apresentar a seguir.

# O Dashboard

O Dashboard é o nosso centralizador das ferramentas que desenvolvemos para te ajudar no caminho. Nele você consegue:

- **Gerenciar seu progresso**
    - Seguindo o plano de 8 semanas ou fazendo no seu tempo (citado anteriormente)
- **Listar as dúvidas a serem sanadas**
    - Para não deixar nada passar
- **Fazer anotações precisas para fixar tudo que foi aprendido**
- Realizar as **provas de conclusão** de módulo e **tirar seus certificados**

Nas primeiras aulas em vídeo do treinamento, nós compartilhamos o link para o Dashboard.

# As tecnologias

A programação é uma área extensa, com muitas tecnologias e conceitos, por isso escolher uma Stack (conjunto de ferramentas) quando você está começando, pode ser complicado.

Neste treinamento nós sugerimos e ensinamos uma Stack que é muito forte no mercado, tem muitas oportunidades e traz muita agilidade e qualidade aos projetos.

Essa Stack é baseada na linguagem de programação JavaScript, então não é necessário aprender outras linguagens de programação (tornando o seu aprendizado mais veloz e eficiente).

## As tecnologias que vamos aprender:

Caso você não compreenda ainda algum termo usado ou não conheça alguma das tecnologias, não se preocupe, falaremos sobre tudo isso durante o decorrer do guia e do treinamento.

**Principal:**

- HTML 5
- CSS 3 + SASS
- CSS Moderno (Flexbox + GRID)
- Bootstrap
- JavaScript + Lógica de programação
- Git e GitHub
- TypeScript
- Banco de dados SQL (PostgreSQL)
- NodeJs (ExpressJS)
- NextJs (React)

**Extras:**

- Bulma
- MongoDB
- ElectronJS

Vamos ensinar detalhadamente essas tecnologias, sempre mesclando aulas teóricas com exercícios práticos (com resolução), quizzes de reforço e aplicação em grandes projetos ao final.

# Os grandes Projetos

Uma coisa que acredito fortemente é: O Programador(a) se desenvolve verdadeiramente, quando trabalha em algo real.

Por isso, nesse treinamento, depois de aprender fortemente a base necessária, nós partimos para o desenvolvimento dos projetos práticos.

## EAD inspirado no Netflix

O primeiro projeto, que faz parte do coração do treinamento, é uma plataforma EAD inspirada na interface do Netflix.

Desenvolvemos do zero, admin, API, frontend e realizamos o deploy completo (deixamos disponível em uma URL).

**As tecnologias usadas nele são:**

- ExpressJS (nodeJs)
- NextJs
- Bootstrap 5
- Sequelize
- PostgreSQL

## APP inspirado no Evernote

O segundo projeto, que está como extra, é um APP de notas inspirado em um dos APPs de notas mais famoso, o Evernote.

Nele nós também desenvolvemos tudo do zero, API e Frontend. Ele é um projeto altamente funcional e cheio de features importantes, que vão reforçar os conteúdos ensinados no treinamento.

**As tecnologias usadas nele são:**

- ExpressJs (nodeJs)
- React
- Bulma
- Mongoose
- MongoDB

Esses projetos vão te dar conhecimento prático, uma visão mais ampla e **confiança para fazer projetos** para você e para outras pessoas/empresas.

## As provas com certificado

Que graça tem jogar um jogo onde você não sabe se está progredindo e não pode mostrar o resultado das suas conquistas para o mundo? Bem pouca.

Por isso, nós desenvolvemos as provas com certificado por módulo, ou seja, ao final de cada módulo você vai ter uma prova referente ao que foi ensinado. Sendo aprovado(a), você vai receber o nosso certificado personalizado com seu nome, o nome do módulo e a carga horária.

Bem legal né?

## O guia de carreira

O perigo de toda jornada, é termos focado tanto nos detalhes do caminho, que nos esquecemos do destino.

Mas isso não vai acontecer aqui, porque eu sei que você está nesse treinamento porque quer desenvolver as suas habilidades para que depois possa utilizá-las, seja criando projetos para você mesmo(a) ou criando uma carreira de sucesso na programação.

Por isso, nós temos um módulo especial (na parte de vídeo) chamado “Carreira de Programação”, nele, nós vamos te ensinar:

- Como o mercado de TI funciona
- Quais tipos de trabalho um programador pode fazer
- Como preparar o seu portfólio (GitHub e LinkedIn) para atrair vagas
- Como criar um currículo moderno
- Como aplicar para vagas de emprego
- Como abrir um PJ para prestar serviços de programação (da forma correta)
- Como mandar bem nas entrevistas de emprego
- E muito mais

Esse módulo vai te dar a base necessária para atingir os seus objetivos através da Programação, então depois que finalizar a base técnica e os projetos completos, mergulhe nele.

Nenhum desses elementos do treinamento foram escolhidos aleatoriamente, eles foram desenvolvidos no calor da batalha, teste após teste, até entendermos o que mais funciona para dar o suporte necessário para os alunos terem sucesso.

Então, agora que você já sabe o que teremos no treinamento, podemos seguir a nossa jornada 🤘

# O CAMPO DE BATALHA
# - O MERCADO -

**Quanto mais você sua no treinamento, menos sangra no campo de batalha.**

É fundamental que você tenha uma visão de porque e para onde está indo, então eu quero te apresentar a programação e o mercado de Programação (de forma leve e sem blá blá blá).

A primeira pergunta que você deve me fazer é:

## Léo, o que é Programação?

Programação é uma palavra que usamos com dois sentidos, o primeiro, é para referenciar o ato de Programar.

Programar == Planejar e desenvolver instruções em uma linguagem que a máquina compreenda, para que ela faça as tarefas que desejamos.

Exemplo, o conjunto de instruções que faz com que o seu Browser exiba a página do Google ou o conjunto de instruções que faz com que o despertador do seu smartphone toque.

Também usamos a palavra Programação, quando queremos fazer referência a área ou mercado que envolve os profissionais responsáveis por criar essas instruções.

A segunda pergunta que você deve fazer é:

## Léo, o mercado de Programação é bom?

O mercado de Programação (ou TI, Tecnologia da Informação) está bem aquecido, afinal de contas, a cada dia que passa, utilizamos mais máquinas (computadores, smartphones e etc) que precisam de mais pessoas para programá-las.

Sim, é um mercado que paga bem, te permite trabalhar de pijama em casa e ainda te dá a liberdade de trabalhar para outros países e ganhar em uma moeda mais forte.

Naturalmente, que precisa de esforço para se posicionar no mercado e obter seu lugar ao sol, mas o esforço ao meu ver (e baseado nas centenas de carreiras que já acompanhei) é bem menor que na maioria das áreas.

E se você investiu no nosso treinamento + livro, e está nesse momento lendo essas linhas, eu sei que você é uma pessoa acima da média e que vai fazer o necessário para crescer e conquistar um lugar incrível na Programação.

## As especializações

Como eu disse acima, nós como programadores, somos **especialistas em criar instruções** para máquinas.

E como existem tipos diferentes de máquinas e objetivos diferentes de uso dos softwares, isso leva a uma divisão dos profissionais de TI em sub áreas.

Alguns exemplos:

Programadores que desenvolvem APPs (Softwares) que rodam no Mobile (Smartphone), são chamados de Programador(a) Mobile.

Programadores que desenvolvem APPs que rodam no PC (Computador), são chamados de Programador(a) Desktop.

Programadores que desenvolvem a interface dos sites ou WebAPPs (sites mais complexos e com mais interação) que rodam no Browser (Google Chrome, Firefox, Safari, etc) são chamados de Programador(a) Frontend.

Programadores que desenvolvem a parte lógica, geralmente associada ao banco de dados e que roda no servidor (Computador destinado ao armazenamento de sites, dados e softwares) e que tem o propósito de dar funções (como permitir cadastros e consultas) ao Frontend, Mobile e Desktop, são chamados de Programador(a) Backend.

Nesta jornada, você vai se tornar um Programador(a) Full Stack, ou seja, vai dominar o Frontend e o Backend focados no desenvolvimento de projetos Web.

Na parte de aulas (em vídeo) do treinamento, passaremos detalhadamente por cada um dos tópicos fundamentais do Frontend e Backend de forma prática.

Incrível né?

## Programador raiz VS programador reclamão

Existe um espaço muito grande para você ter sucesso na programação, ou seja, para conseguir seus objetivos através da programação, mas é claro, você precisa fazer o esforço necessário.

Podemos identificar dois perfis que entram na Programação: o programador raiz (esse vai longe) e o programador reclamão (esse morre na praia), e eu sinceramente te aconselho a ser o primeiro.

**O programador raiz:**

- Estuda constantemente e com atenção
- Prepara seu portfólio adequadamente
- Quando tem dúvidas, pesquisa no Google ou Stack Overflow antes de perguntar aos outros porque sabe que 90% das respostas estão lá.
- Entende como o mercado de TI funciona e faz o necessário para tirar o melhor dele
- Não fica pulando de tecnologia em tecnologia tentando achar a tecnologia que vai resolver todos os problemas
- Respeita quem está o ensinando, porque entende que essa pessoa já esteve no seu lugar e atingiu o que ele busca.

- É cooperativo(a) com os colegas.

**O programador reclamão:**

- Estuda de forma aleatória e sem atenção
- Cria um GitHub e LinkedIn desatualizado, com erros de português e sem nexo
- Pergunta antes de pesquisar porque não valoriza o tempo do outro
- Reclama do mercado de T.I e fica falando que está saturado por isso não tem oportunidades
- Fica pulando de tecnologia em tecnologia e não é especialista em nada
- Não respeita os professores
- Só quer ajuda e respeito, mas nunca ajuda e nem respeita ninguém.

Eu entendo que entrar em um novo mundo, nos deixa ansiosos e muitas vezes frustrados, mas temos que ter a maturidade emocional de saber que é assim mesmo e que vai passar.

O foco nos estudos, o respeito aos professores (guias) e aos colegas, o realismo em relação ao mercado de TI e a criação de um portfólio adequado, são os alicerces que vão te permitir criar uma carreira incrível no mundo da programação.

***"Tudo que é feito com amor, prospera".***

Os conceitos que eu te apresentei aqui, servem como uma visão geral de como navegar pelo mercado de programação de forma mais eficiente, seja entendendo o seu propósito nele, determinando qual especialização você deseja dominar e compreendendo como se comportar para ter sucesso.

# AS 3 PRINCIPAIS ARMAS - FERRAMENTAS -

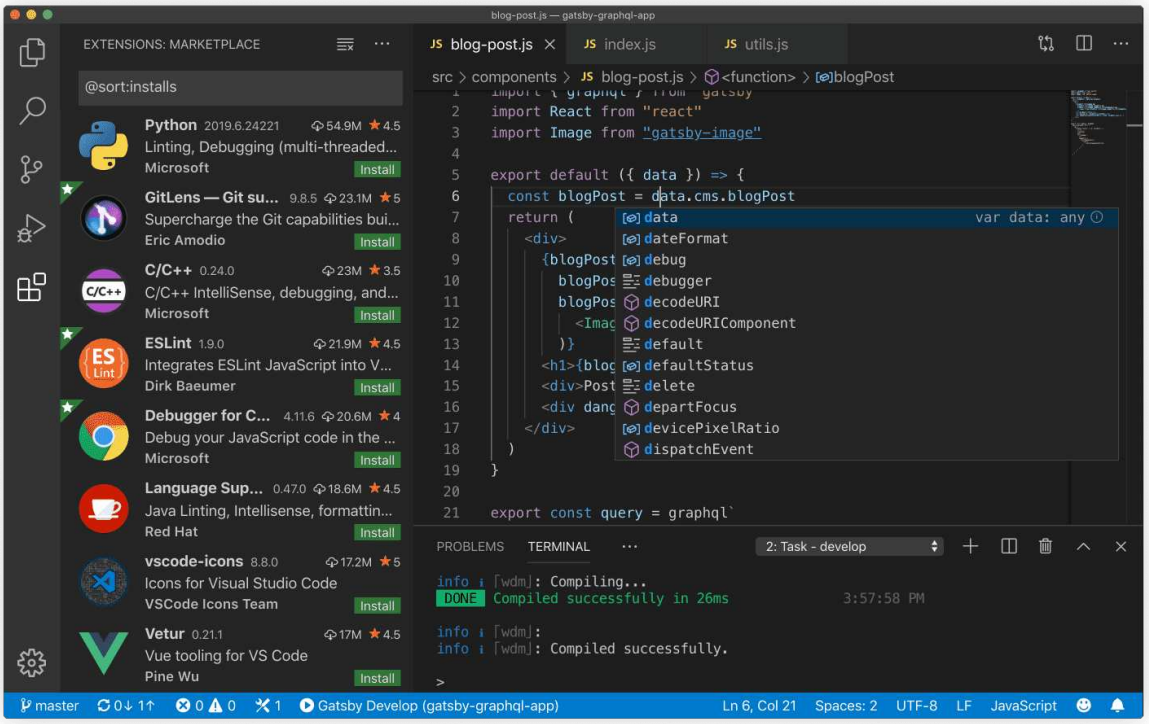
***"O que seria de um(a) guerreiro(a) sem a sua espada e o seu escudo?"***

Bom, nós programadores não enfrentamos dragões diretamente, mas também precisamos das nossas ferramentas para lutar.

Vamos falar aqui sobre as ferramentas básicas que você vai precisar na sua jornada, lembrando que nós ensinamos sobre elas e sobre como instalá-las na parte em vídeo, então aqui é apenas um reforço com algumas adições.

> *Se você tiver preferência por outras ferramentas compatíveis, tudo bem. Mas caso esteja começando agora, recomendo fortemente que utilize as ferramentas mencionadas aqui, pois são as usadas nas aulas.*

## O editor de Textos (Vs Code)



O editor de textos é a ferramenta que usamos a maior parte do tempo, ele serve essencialmente para termos acesso visual aos nossos códigos e podermos editá-los.

Você deve se lembrar do bloco de notas, ele também é um editor de textos, porém, hoje em dia temos editores com ferramentas adicionais que aceleram muito o nosso trabalho.

O Vs Code (desenvolvido pela Microsoft) é o mais popular atualmente, ele é rápido, flexível, permite a adição de uma infinidade de ferramentas (desenvolvidas pela comunidade), tem integração com o terminal (falaremos sobre ele depois) e pode ser instalado em qualquer sistema operacional.

Apenas para título de curiosidade, o Vs Code (que é diferente do Vs Code Studio), foi desenvolvido em JavaScript, que é a linguagem que você vai aprender neste treinamento.

Inclusive, para tornar ele um projeto Desktop, ou seja, que pode ser instalado no Computador, foi usado a biblioteca ElectronJs, no final do treinamento temos uma aula em vídeo ensinando como usar esse framework incrível.

Na parte em vídeo nós demonstramos como instalar o VS Code e navegar pela suas funções básicas, então eu vou me limitar a falar aqui sobre as extensões.

As extensões do Vs Code, são funções (conjuntos de códigos) desenvolvidos pela comunidade para dar mais funcionalidades ao Vs Code.

Nos capítulos deste livro, específicos de cada tecnologia ensinada, nós trazemos uma lista com as melhores extensões para usar associado àquela tecnologia.

***Link para baixar o VS Code:*** *https://code.visualstudio.com/download*

Tenho certeza que você e ele vão se tornar melhores amigos em breve.

# O browser e seu console

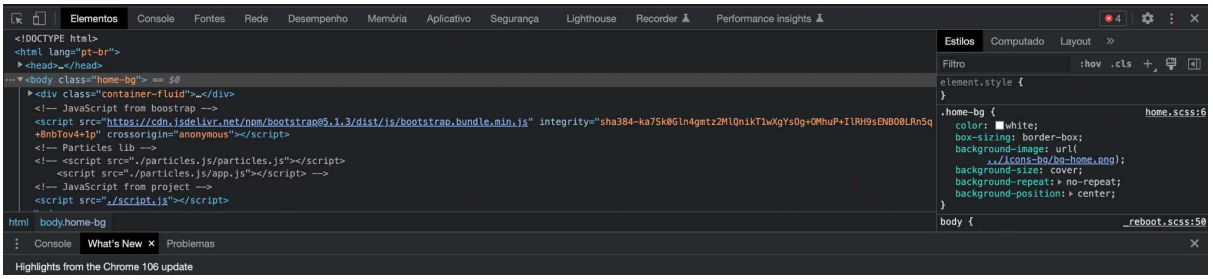
O browser (ou navegador), é uma ferramenta que provavelmente você já está habituado(a), é nele que você navega pelos sites da Web.

Porém, agora que você é um(a) programador(a) você precisa entender que existem mais coisas no seu browser do que parece.

Se você for no Chrome (browser que indico), você pode clicar no menu superior em “ver”, depois em “desenvolvedor” e por fim em “ferramentas para desenvolvedores”.

> *Essas instruções podem variar um pouco dependendo do seu sistema operacional.*


Quando você fizer isso, deve ver o seguinte menu:



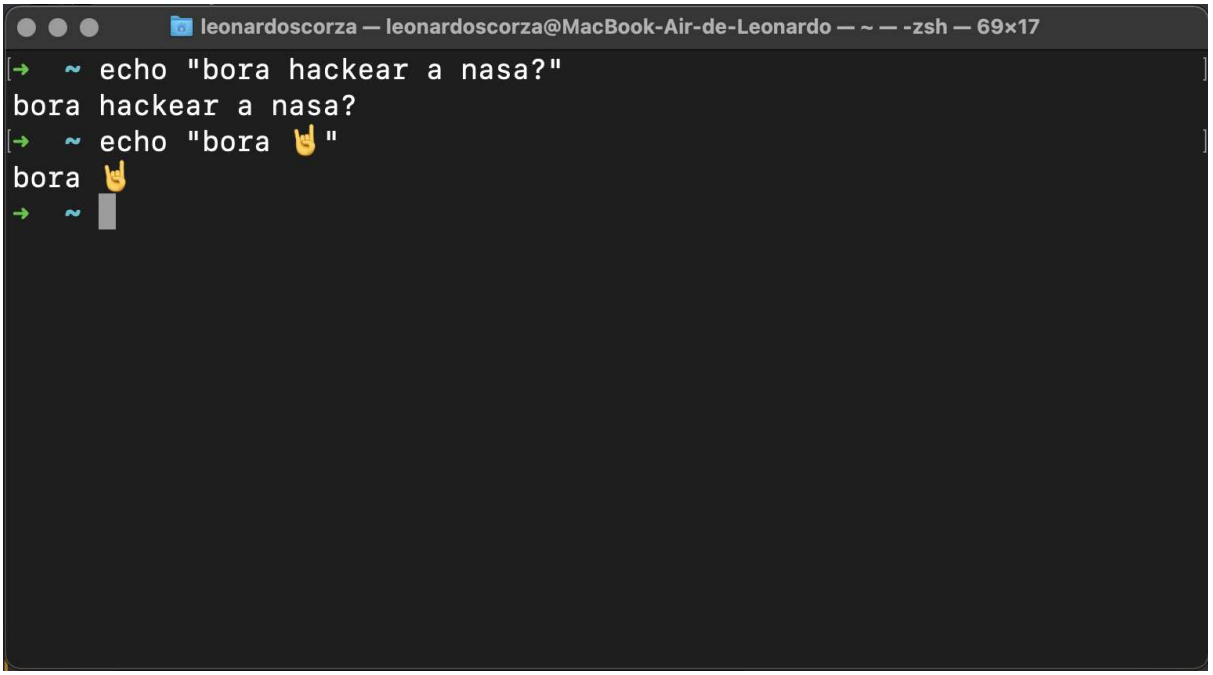
Nele você consegue ver o código fonte do site que está navegando, consegue dar comandos javascript, ver o que foi e o que está sendo baixado e realizar mais ações úteis no dia a dia de programação.

Mas calma, você vai ver o uso prático desse menu na parte em vídeo, a minha ideia aqui é apenas te mostrar que existem ferramentas incríveis de desenvolvimento no seu browser.

*Caso você ainda não possua o Chrome, baixe por aqui: https://www.google.com/intl/pt-BR/chrome*

## O terminal (console)

É bem comum em filmes onde aparecem programadores ou hackers, que eles estejam trabalhando em telas pretas com “letras pequenas”.




Geralmente quem vê esse tipo de cena, imagina que programação seja algo complexo, para alguns poucos malucos que tem coragem de entrar na área, mas isso não está nem perto de ser verdade.

O terminal, é uma ferramenta que vem no seu sistema operacional, com ele é possível dar comandos em texto para o seu computador, exemplo:

- Criar uma nova pasta
- Verificar o uso de memória
- Rodar determinado software
- Fazer chamadas para outros computadores na rede

Nada de outro mundo né?

É bem provável que você use diariamente o terminal enquanto estiver trabalhando, inclusive, é possível abrir ele dentro do VS Code para conseguir editar seus códigos e rodar seus comandos sem precisar ficar mudando de programa.

Os sistemas operacionais Linux e MacOs, por terem uma origem em comum, têm o terminal no mesmo padrão, já o Windows, tem um terminal com comandos diferentes.

Hoje em dia, é possível usar o terminal do Linux (que é mais flexível) dentro do Windows, nós mostramos como fazer isso em detalhes dentro das aulas em vídeo, também ensinamos como usar os principais comandos.

Essas são as ferramentas que você vai usar na maior parte do tempo, então, instale elas como indicado (na parte em vídeo) e brinque um pouco com elas até se familiarizar.

# GUIA DE SOBREVIVÊNCIA
# - AOS BUGS -

***"Um problema sem solução, é um problema mal colocado".***
*Ralph Waldo Emerson*

Quando você está programando, nem tudo ocorre como esperado, erros acontecem, a dificuldade em estruturar soluções acontece e às vezes o caos se faz presente.

Mas tá tudo bem, isso é comum no dia a dia de programação, afinal, estamos sempre usando novas tecnologias e explorando os limites do que podemos fazer com os códigos.

Neste guia nós vamos falar sobre algumas atitudes que você pode/deve tomar para solucionar essas dificuldades comuns no dia a dia de um(a) Programador(a).

## Um BUG inesperado apareceu

Você está lá, tranquilamente, incluindo uma nova função no seu projeto e de repente, surge uma mensagem em vermelho sangue dizendo “Parameters not defined in line 10”.

O coração bate mais rápido, a veia da testa já fica aparente, o suor escorre e você pensa: “E agora, o que eu vou fazer?”

Calma, tá tudo bem.

**A primeira atitude que você deve tomar é:** ler a mensagem e tentar entender o seu significado, caso você não domine o básico de inglês, use o google translate (a maior parte das mensagens de erro são em inglês).

No exemplo, a mensagem quer dizer “Parâmetros não definidos na linha 10”, ou seja, verifique qual foi o código adicionado à linha 10 que está gerando esse erro.

Caso mesmo olhando o código da linha 10 (e o significado da mensagem), você não identifique o erro, **você deve tomar a atitude número dois**.

Copie a mensagem de erro e cole no Google para ver se aparecem possíveis soluções, afinal, provavelmente centenas de pessoas já devem ter passado pelo mesmo erro e colocado no Stack Overflow (site de perguntas e respostas sobre programação).

Verifique na pesquisa, se aparecem links para possíveis respostas à sua dúvida (muuuito provavelmente vai aparecer), visite esses links, traduza a página caso não compreenda por estar em inglês e vá tentando as soluções apontadas.

Provavelmente, seu problema será resolvido, se não, execute **a atitude 3** (que deve ser usada só em casos de emergência, para que você se torne um(a) programador(a) autônomo(a)): pergunte para outra pessoa (na comunidade do treinamento por exemplo).

## Não sei como programar "x" coisa

É comum, principalmente quando estamos no começo, não saber como criar determinado conjunto de códigos para realizar um trabalho. Pode ser um pouco assustador, mas é normal.

Um exemplo, imagine que você queira criar uma barra de navegação usando Bootstrap (Bootstrap é um framework que ajuda a criar interfaces web responsivas), mas você não faz ideia de como começar a sua barra, o que você faz?

Bom, como você está usando neste exemplo um framework popular (bootstrap), a primeira coisa é ir até a documentação do bootstrap (quase toda tecnologia tem uma documentação online) e verificar nessa documentação se tem as instruções para criar uma barra de navegação.

Caso você não encontre, você deve pesquisar no google "barra de navegação com bootstrap", existe uma grande chance de que você encontre tutoriais em texto e vídeo demonstrando como criar essa barra.

Se você não encontrar, você pode simplesmente pesquisar em inglês "navigation bar bootstrap", provavelmente você vai encontrar muitos

tutoriais ensinando como fazer (vale a pena, mesmo que você tenha que traduzir a página).

Em último caso, você pode simplesmente pedir um direcionamento para alguém mais experiente (na comunidade do OneBitCode por exemplo), explicando detalhadamente o que quer fazer e pedindo instruções de que caminho tomar para conseguir realizar a tarefa.

Com o tempo, fica natural pesquisar dessa forma, inclusive, programadores com muitos anos de experiência continuam pesquisando, afinal de contas, programação não é sobre decorar e saber tudo e sim sobre **saber encontrar as respostas**.

## Meu projeto está o caos

Apenas para termos uma comunicação precisa, quando eu digo “meu projeto está um caos”, eu quero representar aquele momento onde as coisas mais ou menos funcionam, mas você tem dificuldades em adicionar novas features (funções) no seu código porque é difícil de compreender como as coisas estão funcionando.

Quando isso acontece, a primeira coisa que você precisa fazer é respirar, relaxar e pensar o seguinte: “Eu tenho tempo de ajustar essa bagunça agora?”.

Se a resposta for não, é bom você arranjar um tempo assim que possível para arrumar a bagunça, mas se a resposta for sim, aí é hora de botar a mão na massa.

Primeiro, o caos se instala geralmente por que grande parte dos **projetos nascem sem planejamento**, ou seja, você não tinha uma definição em texto (ou visual) do objetivo e das features que deveriam estar no seu código.

Só com o plano na cabeça, fica fácil tudo virar uma bagunça, então é fundamental usar técnicas de planejamento de projeto antes de começar (Nos APPs completos do treinamento, eu ensino como planejar de forma profissional).

Caso você não tenha feito o planejamento e já esteja no meio do projeto, vale a pena dar um passo atrás, replanejar e depois ajustar o projeto para seguir o plano.

Caso você tenha um plano e mesmo assim o seu projeto virou um caos, provavelmente é porque os códigos não foram bem estruturados, **aí é hora da famosa refatoração**.

**Refatoração** é quando você pega seus códigos bagunçados e reescreve eles de forma organizada, existem várias técnicas para fazer isso (falamos sobre no treinamento), mas gostaria de já deixar como recomendação o livro "**Refatoração: Aperfeiçoando o Design de Códigos Existentes**".

Caso você se sinta realmente perdido(a) na hora de replanejar ou de refatorar, é nessa hora que você busca o aconselhamento de algum programador(a) mais experiente para te dar um direcionamento.

Lembrando, pedir aconselhamento é bom, mas tentar sozinho antes, vai te dar mais autonomia e te ajudar a se desenvolver mais rapidamente.

Se você masterizar esses três pontos, como reagir quando Bugs aparecem, como pesquisar quando não sabe como desenvolver algo e como organizar seu projeto quando o caos reina, você vai chegar em um novo nível na programação.

Então, sempre que tiver algum desses problemas, volte nessa parte do guia, leia com calma e implemente o que eu disse, tudo que eu indiquei é fruto de mais de 10 anos de experiência lutando com essas adversidades.

# HTML
# - O ESSENCIAL -

## O que é e para que serve o HTML?

Sigla para HyperText Markup Language — Linguagem de Marcação de Hipertexto —, o **HTML** é o componente base da web. Isso quer dizer que ele permite a construção de websites e a inserção de novos conteúdos, como imagens e vídeos, por meio dos hipertextos.

O HTML é essencial para todas as páginas web, ele permite que nós estruturemos nossas páginas (usando as famosas tags </>), colocando textos, botões, campos para digitar informações e etc tudo de forma organizada.

**Um breve contexto histórico da criação do HTML vindo do Wikipedia:**

“Tim Berners-Lee (físico britânico) criou o HTML original (e outros protocolos associados como o HTTP), numa estação NeXTcube, usando o ambiente de desenvolvimento NeXTSTEP. Na época, a linguagem não era uma especificação, mas uma coleção de ferramentas para resolver um problema de Tim: a comunicação e disseminação das pesquisas entre ele e o seu grupo de colegas. A sua solução, combinada com a então emergente internet pública (que tornar-se-ia a Internet), ganhou atenção mundial.”

O HTML foi evoluindo com o tempo, se tornou mais abrangente e organizado, hoje estamos na versão 5, por isso temos o módulo HTML 5 (para diferenciar de outras versões).

Um erro comum é dizer: HTML é uma linguagem de programação, ele não é, é uma linguagem de marcação, já que serve para estruturar as páginas e não “programar” elas.

Um exemplo de código HTML:

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="utf-8" />
        <title>OneBitCode</title>
    </head>
    <body>
      <p>Olá Mundo</p>
    </body>
</html>
```

Esse é um código simples, que mostraria na tela a frase: Olá mundo.

Mas não se preocupe, ensinamos do zero o uso do HTML detalhadamente na parte em vídeo, aqui eu quero apenas lhe apresentar alguns detalhes úteis.

## Quando usar HTML?

Usamos HTML em todas as páginas Web.

## Requisitos para usar

Você não precisa de nada especial para usar o HTML, basta poder editar um arquivo .html para inserir as tags de marcação desejadas e um browser, onde você possa abrir o documento criado.

# Links importantes

**Documentações para consulta:**

- https://developer.mozilla.org/pt-BR/docs/Web/HTML
    - Aqui você encontra as definições e exemplos de uso de cada TAG de forma detalhada
- https://www.w3schools.com/tags
    - Aqui também é possível encontrar a definição das tags e outros exemplos de uso

# Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do HTML.

1. Criar um formulário de pesquisa com os campos (de texto, número, select e etc) que o usuário deve responder e um botão para submeter a pesquisa.
2. Criar uma página estilo Wikipedia documentando algum conhecimento que você possui (com textos e imagens), por exemplo, como tocar violão.

# CSS
# - O ESSENCIAL -

## O que é e para que serve o CSS?

Cascading Style Sheets é um mecanismo para adicionar estilo a um documento web.

Você acabou de ver sobre o html, que é a estrutura da nossa página. O css é o visual da nossa página. Enquanto o html vai servir para você colocar uma imagem, um botão, um texto, com o css você vai definir o tamanho da imagem, definir a cor do botão e também qual a fonte que você quer no texto.

Se trouxermos um exemplo de uma casa, você pode pensar que o html vai ser a estrutura da casa. Apenas com html a casa fica funcional, você até consegue morar lá, porém não vai ser agradável visualmente. Agora, se você coloca uma pintura, uma decoração, a casa já ganha outro estilo, e é isso o que o css faz.

O css assim como o html, não é uma linguagem de programação. O css é uma linguagem de estilização, sendo focada apenas nisso.

## Quando usar CSS?

O css é usado em todas as páginas modernas, se não for usado o css é usado variações dele, como o SASS por exemplo, que você vai estudar sobre logo logo.

Como já foi dito, o css é o estilo da sua casa, você não quer ter uma casa feia e sem graça, certo? Então você usa o css em toda página que fizer. Porém, seu uso não é obrigatório em todo elemento da página. Você pode deixar textos, cores e outros itens com o padrão inicial.

Nós temos até momentos que não precisam de css, usando por exemplo o bootstrap, que vamos ver sobre mais para frente. Então o css é usado de acordo com a necessidade nos elementos, mas toda página vai ter o seu css, mesmo que seja pouco.

## Requisitos para usar

Assim como o html, você não precisa de nada especial para usar. Utilizando um editor de texto, você vai criar um arquivo .css e vai modificar ele de acordo com o que quiser. Para modificar os elementos na página, nós usamos os identificadores dentro do html (Classes de ids).

## Links importantes

Recomendo que você sempre use a documentação para entender alguns pontos importantes e superar as dificuldades que aparecerem.

**Documentações para consulta:**

- https://developer.mozilla.org/en-US/docs/Web/CSS (Documentação oficial)

- https://www.w3schools.com/cssref/ (Documentação de apoio)

## Ideias de projetos para praticar

Com o css e o html, você já tem um poder grande em mãos, é possível criar algumas páginas legais para treinar e fixar o conteúdo.

1. Página de login do instagram é uma página de login bem simples, onde apenas usando html e css simples você consegue um belo resultado.

2. Estilizar a página inspirada no wikipedia que você criou para treinar html. Com css agora você consegue colocar cores, fontes diferentes e deixar ela mais bonita.

3. Página de login da netflix. Você vai precisar estudar um pouco mais para aplicar alguns conceitos nela, mas vai ser um belo treino para você.

# FLEX-BOX E GRID
# - O ESSENCIAL -

## O que é e para que serve o Flex-Box e Grid?

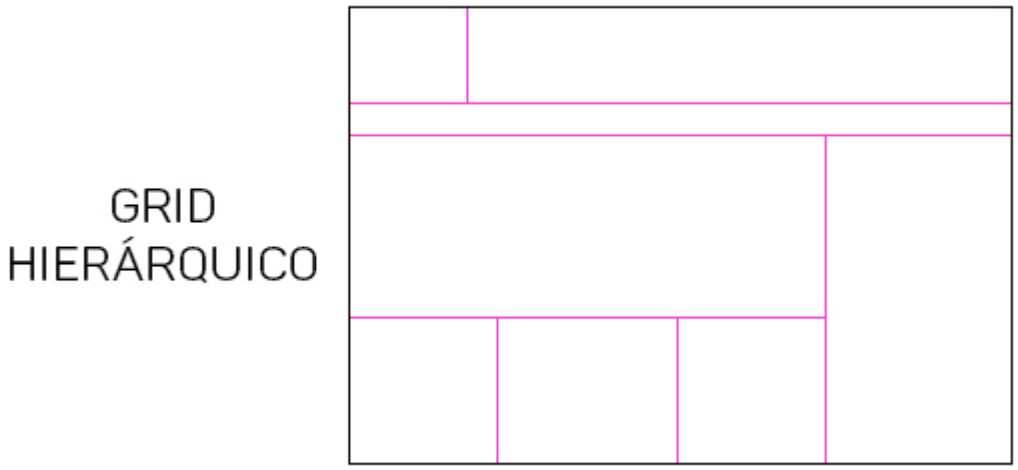
O flex-box e o grid são ferramentas dentro do css de forma nativa, ou seja, não é necessário instalar nada para usar. Elas servem para organizarmos os layouts e as páginas.

Ao utilizar o flexbox, nós conseguimos deixar coisas centralizadas, alinhadas e também responsivas (uma página responsiva é aquela que vai se adaptar a todos os dispositivos de forma horizontal, sem prejudicar o layout) de forma muito mais fácil.

O layout padrão das páginas é vertical, onde nós vamos ter um item abaixo do outro, seja ele uma div, um texto, uma imagem (posição de coluna), entre outros. Com o flexbox, podemos deixar esses itens em linha, fazendo eles ficarem um ao lado do outro. E é desse jeito que podemos deixar os layouts ainda mais bonitos, com imagens e outros elementos lado a lado.


O grid é algo parecido, porém com outro estilo. Ele também é um organizador de layout, porém, ao invés de termos os itens em forma de linha, aqui vamos ter linhas e colunas, para fazer uma divisão do layout.

O grid é bastante utilizado como o exemplo da imagem abaixo, colocando colunas e linhas, onde podemos por exemplo, criar a divisão de um blog com o que está abaixo.




É muito normal usarmos o grid e o flexbox juntos, às vezes um dentro do outro. Isso é bastante visto no curso. Eles têm basicamente o mesmo objetivo, que é organizar os layouts, mas na prática são diferentes.

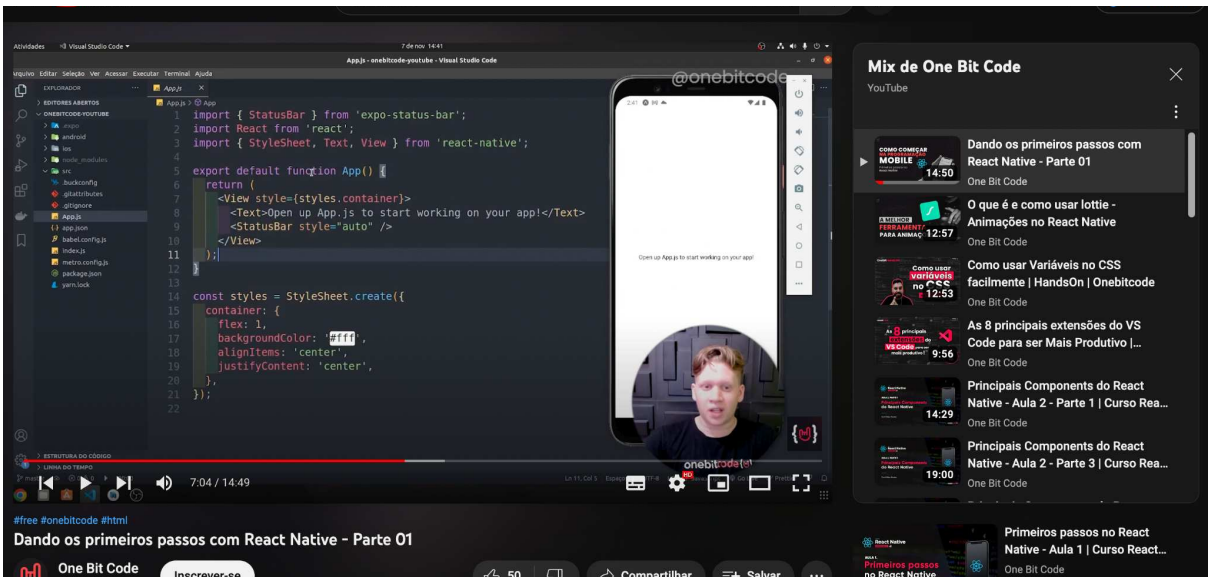
## Quando usar CSS?

Nós vamos usar eles em diferentes momentos. Começando com o flexbox, nós podemos usar eles em momentos onde é ideal que tenhamos objetos um ao lado do outro, de forma organizada, fazendo com que eles fiquem se adaptando com o tamanho da tela.

Um exemplo, é a barra do youtube, onde temos barra de pesquisa, perfil, etc...


Além disso, nós podemos usar ele quando queremos imagens ou textos um ao lado do outro, fazendo essa mesma adaptação, como por exemplo a página da netflix.


O grid por exemplo, pode também ser usado no exemplo do youtube, fazendo com que a gente tenha uma coluna para o vídeo e uma para os vídeos recomendados, vídeos da playlist, etc.




Nós vamos usar de acordo com a necessidade de uso, podendo usar um dentro do outro, juntos.

## Requisitos para usar

Para usar eles, você precisa saber sobre css e html o suficiente para montar layouts simples e legais.

É necessário bastante atenção e prática para usar eles com mais profundidade, pois eles são um grande passo no css.

Até pegar a ideia e perfeição deles, você vai precisar praticar muito, então quando surgirem as primeiras dificuldades, não se preocupe, é normal. Você não deve desistir ou desanimar, lembre-se que todos nós já passamos por isso com flexbox e grid, e você vai agradecer por não desistir. o/

## Links importantes

**Documentações para consulta:**

- Flexbox: https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox

- Grid: https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Grid_Layout/Basic_Concepts_of_Grid_Layout

## Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do Flexbox e grid.

1. Página inicial da netflix: Você não precisa fazer todas as funcionalidades que tem lá, mas é um ótimo exemplo para você seguir, pois temos muitas organizações complexas que precisam do flexbox e grid.

2. Um blog simples: Você pode colocar nesse blog um sidebar, um header bem dividido e um footer, pode ser algum autoral ou o exemplo que temos no curso. É importante demais a prática, se quiser um desafio ainda maior, um clone simples do layout do youtube seria excelente.

# SASS
# - O ESSENCIAL -

## O que é e para que serve o SASS?

Sass (Syntactically Awesome StyleSheets, ou Folhas de Estilo Sintaticamente Incríveis) é uma linguagem de extensão css, onde você pode ter funcionalidades que não temos no css normal.

O sass é principalmente utilizado na organização e separação de arquivos grandes, fazendo com que fique fácil e simples de dar manutenção.

## Quando usar SASS?

O sass é usado quando você quer funcionalidades extras. No nosso módulo de sass você consegue descobrir as funcionalidades que temos no sass e decidir se o projeto vai precisar ou não.

Você não necessariamente precisa usar SASS, mas ao utilizar, você vai ter em vários pontos uma produtividade maior, fazendo com que o projeto fique mais rápido de criar, atualizar ou dar manutenção.

Existem atualmente diversos projetos que utilizam sass, então ter conhecimentos sobre ele é essencial.

## Requisitos para usar

Para usarmos sass, precisamos entender o css e manipular arquivos .scss ou .sass

Quando você termina de manipular um sass, você vai “traduzir” ele para css, chamamos isso de transpilação.

Ou seja, criamos em sass, porém como o browser entende somente css, transformamos nosso arquivo sass em css antes de utilizá-lo no browser.

## Links importantes

**Documentações para consulta:**

- https://sass-lang.com/documentation/ - Documentação oficial

**Extensões úteis do VS Code:**

- Sass compiler
- Live sass - Tanto essa como a de cima servem para você transpilar para o css de forma rápida.
- Sass - Extensão necessária para escrever em sass

## Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do SASS.

1. Sass, é um css com mais detalhes para melhorar a sua produtividade, então você pode fazer as mesmas páginas indicadas na parte de css (Login Instagram, home Netflix), porém em SASS.

# BOOTSTRAP
# - O ESSENCIAL -

## O que é e para que serve o bootstrap?

Bootstrap é um framework web com código-fonte aberto para desenvolvimento de componentes de interface e front-end para sites e aplicações web, usando HTML, CSS e JavaScript, baseado em modelos de design para a tipografia, melhorando a experiência do usuário em um site amigável e responsivo.

Usando o Bootstrap, você não precisa criar cada elemento, você pode utilizar o que já está pronto e criar apenas o essencial para suas páginas, tendo mais produtividade.

Ou seja, você encontra elementos prontos no Bootstrap como botões, barra de navegação, carousel, etc., dessa forma, ele vai servir para agilizar o desenvolvimento, fazendo com que a criação de uma página, seja bem mais rápida e fácil.

## Quando usar bootstrap?

Nós usamos bootstrap para agilizar o desenvolvimento, pois você não precisa criar tudo do zero, podendo reaproveitar diversos elementos.

O bootstrap pode ser usado em toda aplicação que você criar, fazendo você ter muito menos dor de cabeça e muito mais agilidade.

Além disso, se você quiser dar um “ar de desenvolvimento autoral” para a aplicação, você pode mudar cores, tamanhos, fontes, etc., dos elementos já prontos, acelerando a criação e estilização das suas páginas.

## Requisitos para usar

Você precisa saber html e css, não precisa de javascript para usar bootstrap pois o javascript já vem pronto.

Sabendo bem o HTML, você já pode criar diversas coisas, e usando flexbox e grid que tem dentro do bootstrap, agilizará ainda mais o seu processo de criação.

Ou seja, seguindo todos os passos até aqui, você consegue uma ótima base para usar bootstrap.

## Links importantes

**Documentações para consulta:**

- https://getbootstrap.com/ - Documentação oficial

**Extensões úteis do VS Code:**

- Bootstrap snippets

## Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do bootstrap.

1. Página de login do Facebook Com o Bootstrap você já consegue fazer projetos mais completos de forma mais rápida, então podemos começar com a página de login do Facebook, onde há várias informações e itens para a utilização do Bootstrap.

2. Site de viagem com carrossel e informações de passeios para serem feitos.

3. Site institucional Você pode se inspirar em um site de uma empresa e criar a sua própria versão como uma empresa fictícia, isso com certeza dará mais visibilidade no seu portfólio.

# JAVASCRIPT
# - O ESSENCIAL -

## O que é e para que serve o Javascript?

O javascript é uma linguagem de programação usada principalmente na web, porém que também pode ser usada em outros contextos. Ele é o que permite que os sites e aplicativos tenham comportamento dinâmico, como mudar o tema da página de claro para escuro, fazer com que um menu abra ou feche, ou ainda criar, remover e alterar elementos na tela.

Uma linguagem de programação nada mais é do que um conjunto de códigos que contém instruções para o computador. Ao receber essas instruções o computador irá entender o que deverá fazer, e com isso nós podemos criar uma grande variedade de interações em nossos sites e aplicativos.

Considerando tudo isso, o javascript, e as linguagens de programação em geral, são diferentes das linguagens de marcação (HTML) e de estilização (CSS).

O javascript foi criado em 1995 por Brendan Eich para ser uma linguagem capaz de funcionar no navegador Netscape, um dos primeiros navegadores web e que era muito popular na época. O motivo para a criação do javascript era justamente permitir que as páginas HTML, que até aquele ponto eram apenas documentos estáticos, pudessem permitir mais interações por parte dos usuários, que naquele momento eram apenas leitores. Com o passar dos anos o javascript evoluiu e se transformou em um padrão utilizado por todos os navegadores e toda a web, se consolidando como uma das linguagens mais populares do mundo.

Comparado às outras linguagens de programação, dizemos que o javascript é uma linguagem de alto nível, o que significa que ele é de fácil entendimento e muito mais próximo da linguagem humana, ou seja, que um humano utiliza para se comunicar. Do lado contrário, as linguagens de programação de baixo nível são aquelas que utilizam códigos mais próximos do que o hardware utiliza para se comunicar, ou seja, são as linguagens mais próximas do que o computador entende e que normalmente são mais difíceis de se trabalhar. Isso faz do javascript uma ótima opção para dar os primeiros passos na programação.

## Quando usar Javascript?

Utilizamos o javascript quando precisamos de comportamento dinâmico em nossas páginas web, quando queremos criar uma interação que não é possível apenas com HTML e CSS.

Imagine que você está desenvolvendo um site institucional para uma empresa e em uma determinada área do site você deseja colocar uma galeria de fotos. No entanto, você não quer apenas uma galeria com várias fotos fixas, mas uma apresentação de slides em movimento, semelhante ao que temos em softwares como o PowerPoint, por exemplo.

Para conseguir isso, somente HTML e CSS não serão o suficiente, e é nessa hora que entra o javascript. Ele permitirá que você crie uma galeria com imagens em movimento, efeitos de transição entre as imagens, um botão para pausar e continuar a apresentação e tudo mais que sua imaginação quiser. Isso é apenas um exemplo bem pequeno de como e quando podemos usar o javascript. Considerando o quanto os sites de hoje buscam ser mais atrativos e interessantes para prender a atenção dos usuários, você irá utilizar muito o javascript.

Um outro momento em que o javascript é essencial é quando estamos criando um aplicativo para a web.

## Requisitos para usar

Assim como no HTML e no CSS, utilizar o Javascript é tão simples quanto criar um arquivo com a extensão .js, incluir esse arquivo no código HTML através da tag <script> e pronto. Também é possível escrever código Javascript diretamente dentro de uma tag <script>, porém isso não é recomendado e é considerada uma má prática, pois deixará nosso código mais desorganizado e difícil de ser entendido.

Vale destacar aqui que o javascript possui muito potencial, então nós podemos usá-lo para poucas coisas simples ou para coisas incrivelmente complexas. Com isso, é muito importante que para utilizar o javascript o programador tenha sólidos conhecimentos de lógica de programação, conheça bem a linguagem e os seus principais comandos e tenha uma boa

noção sobre algoritmos, esses são os fundamentos que vão permitir construir coisas cada vez mais elaboradas com a linguagem.

## Links importantes

**Documentações para consulta:**

- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript

  Essa é a documentação do Javascript pela MDN. Aqui é onde você encontrará as definições de todos os recursos do javascript, com exemplos e casos de uso de cada um. Ele é como um grande manual de instruções, então você não precisa ler do início ao fim, mas sim consultar sempre que quiser entender como algum recurso funciona ou quando precisar se lembrar de como usar determinado comando.

- https://www.w3schools.com/js/

  O W3Schools é um site com vários guias, referências e tutoriais de tecnologias da web. Assim como o MDN. ele pode ser utilizado como uma documentação, consultando seus materiais quando necessário.

**Extensões úteis do VS Code:**

- Javascript (ES6) code snippets

  Snippets são pequenos atalhos em texto onde você digita o atalho, aperta Enter e um bloco maior de código é escrito automaticamente. Eles nos permitem escrever blocos grandes e repetitivos de código mais rapidamente. Essa extensão é uma coleção de vários snippets gerais para a linguagem javascript.

- Babel Javascript

  Essa é uma extensão que serve para *synax highlighting*, que é quando o editor de texto muda a cor dos comandos para que possamos visualizar melhor o código e entendê-lo mais rápida e intuitivamente. Ela expande as funcionalidades padrão do VS Code para entender melhor os comandos e recursos das versões mais modernas do javascript.

- ESLint (necessita do pacote npm eslint)

  Integra o ESLint ao VS Code para tornar o seu uso fácil e rápido. O ESLint é um *linter* para javascript. Um *linter* é uma ferramenta usada para corrigir e padronizar o nosso código para garantir que estamos seguindo as boas práticas e regras de escrita. Essas regras geralmente são muito importantes quando estamos trabalhando em equipe, mas mesmo quando estamos trabalhando sozinhos podem ser muito úteis.

## Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do Javascript.

1. Aplicativo de clima consumindo uma API pública.
2. Aplicativo de lista de tarefas
3. Galeria de imagens dinâmica
4. Calculadora
5. Site que consome uma API pública para listar seu conteúdo (GitHub API, The Movie Database, PokéAPI, etc)
6. Um jogo simples
7. Aplicativo de controle financeiro

# TYPESCRIPT
# - O ESSENCIAL -

## O que é e para que serve o Typescript?

Typescript é uma linguagem de programação que também é um *superset* do javascript, ou seja, é uma linguagem que possui todos os recursos do javascript e adiciona alguns recursos novos com a intenção de expandir e melhorar o uso normal do javascript.

O principal diferencial do typescript é a existência de um recurso chamado tipagem estática. Ela permite que o programador escreva manualmente o tipo de uma determinada variável, parâmetro ou retorno de função, garantindo que aquele valor deverá obedecer ao tipo escolhido. Com isso, as ferramentas de desenvolvimento, como o VS Code, irão conseguir entender qual é o tipo que aquele valor deveria ter e nos avisar caso estejamos tentando utilizar um tipo incorreto. A principal vantagem disso é a possibilidade de percebermos vários erros durante a escrita do código, sem precisar executá-lo para ver se há algo de errado, o que nos economiza bastante tempo.

Claro que a existência da tipagem estática não é o único atrativo do typescript. Ele possui suas próprias implementações de vários recursos que nem sequer existem no javascript, como interfaces, encapsulamento protegido, etc.

Mas todas essas vantagens também carregam uma dificuldade extra ao se trabalhar com typescript. Os navegadores web e o Node.js, principal ambiente para execução de javascript fora do navegador, não funcionam com typescript, apenas com javascript. Portanto, para que nossas aplicações escritas em typescript possam funcionar, precisamos fazer o processo de converter o código typescript para código javascript equivalente, muito parecido com o que o Babel faz. Claro que o time por trás do javascript conhece muito bem essa dificuldade e o typescript possui um compilador próprio que realiza todo o trabalho pesado por conta própria.

No fim, typescript é um recurso incrível e que aumenta ainda mais o potencial do javascript, tornando-o uma linguagem de programação ainda mais popular entre os programadores

## Quando usar Typescript?

O typescript pode ser utilizado em qualquer cenário substituindo totalmente o javascript, porém as dificuldades extras que ele trás podem ser um custo que não vale a pena em projetos mais simples. Isso porque em projetos pequenos é mais fácil controlar o fluxo dos dados, possíveis pontos de falha e as interações entre os diferentes componentes do sistema.

Portanto, são os projetos de grande escala, aplicações desenvolvidas e mantidas por vários desenvolvedores ao longo de muito tempo e que estão em constante evolução, aqueles que mais sentem os impactos positivos do typescript. Em projetos grandes assim, o typescript é a ferramenta ideal, muitas vezes viabilizando projetos que apenas com o javascript seriam muito mais custosos e complexos, pois o seu uso correto torna a estrutura de um sistema mais sólida e confiável.

## Requisitos para usar

Para utilizar o typescript você precisará de algumas coisas.

Primeiro de tudo, você precisará do Node.js e do npm, que são requisitos para se utilizar o pacote npm do typescript, que possui o compilador do typescript que irá converter o código para javascript.

Além disso, você precisa ter um entendimento da linguagem javascript, afinal, o typescript é basicamente a mesma linguagem, porém com algumas coisas a mais. Portanto, coisas como trabalhar com arrays, objetos, funções, etc, serão praticamente iguais no javascript e no typescript.

Por fim, é importante ter uma boa noção de como utilizar os recursos do typescript, como os *type aliases*, as interfaces, os tipos genéricos, *decorators*, etc, e tentar não fugir do uso deles. Existem situações onde poderá ser complicado utilizar o typescript da forma correta e o typescript possui meios para “anular” esses recursos de forma que não precisemos utilizá-los. No entanto, se começarmos a fazer isso com frequência, terá sido inútil escolher o typescript, sendo melhor se manter utilizando apenas o javascript.

# Links importantes

**Documentações para consulta:**

- https://www.typescriptlang.org/

  Site oficial da linguagem que contém a documentação completa e vários outras links úteis.

**Extensões úteis do VS Code:**

- O próprio VS Code já possui recursos muito bons para se trabalhar com typescript, que inclusive funcionam até quando estamos trabalhando somente com javascript.

# Ideias de projetos para praticar

Para praticar o uso do typescript você pode utilizar os mesmos projetos front-end e back-end que utilizaria para praticar o javascript, porém a sua atenção deve ser maior em alguns pontos:

1. Se esforçar ao máximo para anotar e respeitar todas as tipagens
2. Uso correto das interfaces é fundamental
3. Implementação dos princípios SOLID
4. Uso de padrões de desenho de software (design patterns)
5. Uso da programação orientada a objetos

Mas, para deixar algumas dicas de projetos mais convencionais você pode desenvolver:

1. API Rest que executa operações CRUD

    1. Catálogo de Produtos
    2. Plataforma de EAD
    3. Sistema de Gerenciamento de Conteúdo (CMS) Headless
2. Aplicações front-end utilizando React
    1. Sistema administrativo com dashboard e autenticação
    2. Clone do Trello
    3. Rede social

# GIT E GITHUB
# - O ESSENCIAL -

Antes de começar, precisamos entender que Git e GitHub são coisas diferentes, apesar de serem muito confundidos no começo.

Imagina que você está desenvolvendo um projeto e que ele esteja salvo na sua máquina local. Caso essa máquina pare de funcionar ou você perca o acesso a ela, você perde seu projeto inteiro! Quando se trata de projetos pessoais, isso pode ter um impacto menor, mas quando o projeto é um produto para um cliente, o prejuízo é muito maior.

Além disso, dentro de uma equipe de desenvolvimento, mais de uma pessoa pode alterar um mesmo arquivo e se não for feito o controle adequado dessas versões, tudo pode se perder.

Por isso, utilizamos os versionadores de código, que são sistemas que não apenas armazenam os projetos em um servidor remoto, mas também fazem esse controle seguro das alterações que são realizadas em todos os arquivos.

Agora, para ficar mais claro, vamos entender como o GIT e o GitHub podem ajudar a solucionar esse problema do controle de versões.

## O que é e para que serve o Git?

O Git é um software de controle de versão (versionador), que cria um servidor local na sua máquina, que vai armazenando as versões do seu projeto para que ele esteja mais seguro. Além de armazenar as versões do seu projeto, ele consegue identificar os pontos que foram alterados em cada arquivo para que, caso alguém tenha feito alteração no mesmo arquivo, vocês possam comparar as alterações e fazer o “merge” (do inglês, “mesclar”), que é a união dessas versões, de forma segura e consistente.

Temos um post no nosso Instagram (@onebitcode) explicando de forma bastante simplificada, o que é o GIT. Olha só: https://www.instagram.com/p/CelkzEsLOt6/

## O que é e para que serve o GitHub?

É uma plataforma que oferece diversos recursos além do controle de versão do seu código. Lá você consegue criar vários repositórios (que são como se fossem pastas) que vão armazenar seus projetos, deixá-los liberados para outras pessoas verem e contribuírem com eles. Você também poderá contribuir com o projeto de outras pessoas. Pode ser usada como portfólio, onde você cria seu perfil e expõe todos os seus projetos para que outras pessoas utilizem ou recrutadores possam ver seu trabalho.

## Requisitos para usar

O Git precisa ser instalado no seu computador antes de você usar, já que ele é um software de controle de versão. O GitHub é uma plataforma online, ou seja, nada precisa ser instalado. A única coisa que você vai precisar fazer, é criar uma conta.

Para nenhuma das duas ferramentas você precisa ter conhecimento prévio.

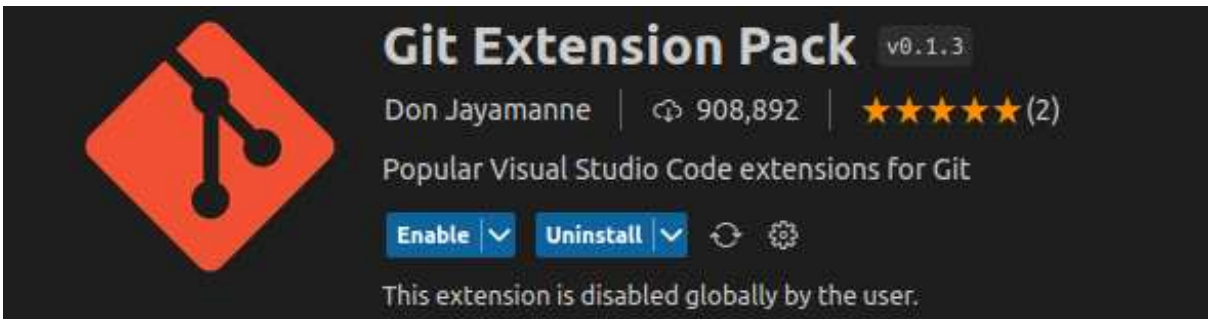
## Links importantes

**Documentações para consulta:**

- Documentação do Git https://git-scm.com/book/pt-br/v2
- GitHub - https://github.com/

**Extensões úteis do VS Code:**

Existem algumas extensões para facilitar nosso trabalho com o Git e alguns até conectam os projetos com a conta do GitHub para agilizar o processo de envio de informações. Uma das mais utilizadas que podemos citar é a Git Extension Pack, que vem com um kit de extensões úteis.




Ela vem com as seguintes extensões:

- Git History (git log) - Para ver o histórico do Git, arquivo ou uma linha específica.

- Project Manager - Para navegar facilmente entre projetos.

- GitLens - O GitLens sobrecarrega os recursos integrados do Visual Studio Code Git. Ele te ajuda a fazer uma "auditoria" no projeto, descobrindo a autoria e histórico de modificações nos arquivos, entre outras funcionalidades úteis.

- gitignore - Permite que você manipule e envie facilmente arquivos de gitignore para seu projeto.

- Open in GitHub / Bitbucket / VisualStudio.com - Facilita o gerenciamento de projetos nas plataformas de versionamento.

## Ideias para praticar

Fizemos um post no nosso Instagram (@onebitcode), indicando alguns sites onde você consegue praticar conceitos do Git, com jogos! Corre lá pra ver: https://www.instagram.com/p/CfAM0f0L1tB/

# REACT
# - O ESSENCIAL -

## O que é e para que serve o React?

O React é uma biblioteca Javascript para ajudar na criação de interfaces de usuários. Para isso ele utiliza componentes, pequenas partes reutilizáveis que juntas irão formar a interface como um todo. O foco principal do React é criar interfaces modernas e dinâmicas para atender as necessidades das aplicações web.

A criação e adoção do React ocorreram como consequência do surgimento e popularização das SPAs, ou *Single Page Applications*, que nada mais são do que aplicativos web que funcionam por completo em uma única página HTML. Dentro de uma mesma página ocorrem todas as interações com o aplicativo, o que permite usar ao máximo o poder do javascript para gerenciá-lo sem lidar com a "perda" das informações e estado atual quando se muda de página.

Entendendo tudo isso, o React é uma biblioteca que foi criada para facilitar a interação com o DOM, além de tornar fácil a criação de elementos na tela com o uso do seu sistema de componentes. Os componentes são como pequenos blocos de HTML, mas que utilizam a sintaxe JSX, que permite definir a estrutura da página utilizando um misto de javascript e XML, que é uma outra linguagem de marcação. Após escrevermos um componente, o React nos permite renderizá-lo facilmente na tela de acordo com as necessidades da aplicação.

Com o React, o dinamismo que o javascript proporciona às páginas HTML se torna ainda mais fácil de ser utilizado.

## Quando usar React?

Utilizamos o React sempre que necessitarmos de uma interface complexa e rica em interações. Geralmente esse tipo de cenário ocorre quando estamos tentando construir algum tipo de aplicação web. Coisas como editores de imagens online, aplicativos de gestão para empresas, redes sociais, etc.

Também é possível utilizar o React para sites simples, porém não é recomendado, visto que o uso do javascript para fazer a renderização da interface pode acarretar em problemas de SEO, um tema que é muito importante para sites institucionais, lojas e projetos semelhantes. Claro que existem formas de solucionar esses problemas, mas aí entra também a questão do aumento da complexidade. Utilizar o React em um site simples pode ser custoso ou trabalhoso demais para o pouco retorno que ele traria, visto que seria um projeto com pouco uso real de javascript e do dinamismo que ele proporciona.

## Requisitos para usar

Para utilizar o React você precisará do Node.js e do npm instalados, porque eles são usados para criar um projeto React. Também é possível inserir o React em um projeto sem precisar instalá-lo durante o desenvolvimento e empacotá-lo na sua aplicação através de uma CDN, mas essa não costuma ser a forma recomendada. Além disso, utilizar o React com o Node.js e o npm não é um processo difícil.

Outro requisito muito importante é o conhecimento da linguagem Javascript. O React foi desenvolvido para ser leve e objetivo. Ele cumpre bem o seu papel, mas é um pacote bem enxuto, portanto durante todo o desenvolvimento você estará trabalhando basicamente com javascript. Você estará constantemente lidando com estruturas condicionais, repetições, arrays, funções e vários outros elementos do javascript, portanto iniciar seus estudos em React sem antes ter uma base sólida em Javascript pode ser arriscado.

# Links importantes

**Documentações para consulta:**

- https://pt-br.reactjs.org/

  Site oficial do React contendo a documentação e vários outros links úteis.

**Extensões úteis do VS Code:**

- ES7+ React/Redux/React-Native snippets

  Um conjunto de snippets úteis para desenvolvimento com React e seu ecossistema.

# Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do React.

1. Lista de tarefas
2. Aplicativo de blocos de anotações
3. Aplicativo para consumir e exibir dados de uma API pública
4. Dashboard com gráficos, relatórios de dados e paginação
5. Sistema de gestão de projetos

# BANCO DE DADOS SQL
# - O ESSENCIAL -

Bancos de dados são coleções organizadas de dados. Por exemplo, quando você faz seu cadastro em uma rede social. Seu nome, endereço, data de nascimento, todas as suas informações pessoais estão armazenadas em um banco de dados para serem acessadas quando necessário.

## O que é e para que serve o SQL?

SQL (Structured Query Language) é uma linguagem de consulta de dados. Todos esses dados que são armazenados no banco, precisam ser acessados e manipulados de alguma forma. Utilizamos a linguagem SQL em bancos de dados que chamamos de bancos de dados relacionais, que armazenam os dados em tabelas que se relacionam entre si.

Então por exemplo, quando eu utilizo o comando:

```
SELECT * FROM CLIENTS
```

Eu estou dizendo, em SQL, que eu quero que o banco de dados me traga todos os dados de todos os clientes que estiverem lá.

Por isso ficou muito comum chamar os bancos de dados relacionais, de bancos de dados SQL, pois eles utilizam essa linguagem para manipulação dos dados.

## Exemplos de Bancos de dados Relacionais (SQL)

Bancos de dados gratuitos e open source:

- Postgresql
- MySQL

Banco de dados Pagos e de código proprietário:

- Oracle
- SQL Server (Possui versão gratuita e paga, mas não é Open Source)

## Requisitos para usar

Para começar a trabalhar com bancos de dados, é interessante que você tenha uma noção básica sobre as camadas das aplicações, em como e em que momento esses dados serão armazenados, mas não é obrigatório. Você pode instalar um dos bancos de dados citados anteriormente e começar a criar suas primeiras tabelas.

## Links importantes

**Documentações para consulta:**

-Postgresql: https://www.postgresql.org/

-MySQL: https://dev.mysql.com/doc/

-Oracle: https://www.oracle.com/br/database/

-SQL Server: https://www.microsoft.com/pt-br/sql-server/sql-server-downloads

No nosso instagram (@onebitcode), fizemos um post explicando o que é Bancos de Dados de uma forma bem simplificada.

Acesse aqui: https://www.instagram.com/p/CeL042VNIX-/

# Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado de Bancos de dados:

1- Criar um projeto de armazenamento para uma loja do ramo de varejo, como: Magazine Luiza, Casas Bahia, etc. Planejar os dados que serão armazenados e quais tabelas serão necessárias.

2- Planejar a estrutura de armazenamento de estabelecimentos como: pizzaria, lanchonete, supermercado. Quais dados esses sistemas precisam armazenar?

- HTML 5
    - Permite estruturar as páginas web
- CSS 3
    - Permite estilizar as páginas web
- CSS Moderno (Flexbox + GRID)
    - Facilitam o posicionamento dos elementos na tela.
- Bootstrap
    - Traz um conjunto de elementos pré prontos (botões, modals, barras de navegação e etc) para acelerar o desenvolvimento de páginas web.
    - Além de ajudar a implementar a responsividade para que as páginas se adaptem bem tanto no PC como no Mobile.
- JavaScript
    - Serve tanto para adicionar funções as páginas Web (como respostas a clicks) como permite o desenvolvimento do Backend

- Git e GitHub

    - O Git nos ajuda a versionar nossos códigos, ou seja, nos ajuda a organizar melhor (em versões) nossos códigos para trabalhar sozinho(a) e/ou em equipe

    - O Github é uma plataforma online, onde podemos subir nosso código fonte. Para deixar ele visível (para o mundo, ou para pessoas selecionadas). Ele serve também como um portfólio para o Programador(a), já que a qualidade dos seus códigos diz muito sobre você.

# NODEJS (EXPRESSJS)
# - O ESSENCIAL -

## O que é e para que serve o NodeJS?

É um ambiente de execução JavaScript, que permite executar aplicações do lado do servidor, sem depender do browser. Ele é muito utilizado, especialmente por trazer boa escalabilidade, flexibilidade e também, uma arquitetura de baixo custo.

Algumas tarefas não são suportadas diretamente por ele, como por exemplo se você quiser que sua aplicação gerencie rotas para diferentes URLs, ou utilize templates para exibir as respostas. Você terá muito trabalho para fazer isso apenas com NodeJS. Será muito mais rápido e fácil se você utilizar um framework, que é uma estrutura pronta, capaz de trazer essas e outras funcionalidades e possibilidades para você só utilizar, sem precisar ficar desenvolvendo do zero. Um exemplo de framework é o Express, que é utilizado para agilizar o desenvolvimento de quem trabalha com o Node.

## O que é e para que serve o ExpressJS?

Express é um framework (uma estrutura pronta), que permite desenvolver diversos tipos de aplicação backend, como APIs RESTfull até sites e aplicações comuns com Javascript, que podem ser fullstack.

Algumas das soluções que o Express traz para a aplicação:

- Gerenciar requisições de diferentes requisições HTTP (GET, POST, DELETE) em diferentes URLs.
- Definir configurações comuns da aplicação (localização de porta, models, etc)
- Gerenciamento de filas através de middlewares.

## Onde Utilizar o Node.js e o Express?

Como o Node é uma ferramenta que proporciona uma boa capacidade de adaptação, pode ser aplicado a diversos tipos de situações que requerem esses benefícios que citamos anteriormente.

Alguns exemplos:

- Pode ser utilizado para trabalhar em uma estrutura que possua muitas conexões concorrentes, como a estrutura de microsserviços, por exemplo.
- Quando é utilizada uma estrutura com bancos de dados NoSQL, pois utilizam uma estrutura de comunicação comum entre eles. O que proporciona maior velocidade e escalabilidade dos processos.
- Em aplicações de mensagens instantâneas.

Além dessas situações, estas são algumas empresas que utilizam NodeJs e o Express nos seus sistemas:

- Netflix
- Linkedin
- Uber
- IBM
- PayPal
- Walmart

Na verdade, precisamos dizer, que além dessas grandes empresas, a maioria das empresas no mercado utilizam alguma aplicação com NodeJs.

## Requisitos para usar

É interessante que antes de começar a estudar Node e Express, você entenda como funciona a arquitetura web e suas camadas (backend, frontend) arquitetura cliente/servidor. Quando chegar nesta parte do conteúdo pelas videoaulas, você estará pronto para compreender o uso do

Node e Express e conseguirá participar com sucesso dos desafios propostos na parte em vídeo do nosso treinamento.

## Links importantes:

**Documentações para consulta:**

- Express: https://expressjs.com/
- Node.js: https://nodejs.org/pt-br/docs/

No nosso instagram (@onebitcode), fizemos um post explicando de forma bem simplificada o que é um framework! Acesse aqui: https://www.instagram.com/p/CgXPI-0uota/

**Extensões úteis do VS Code:**

Existem muitas extensões que podem ser utilizadas, dependendo da necessidade, mas podemos te indicar essa: Express Snippets, que disponibiliza um conjunto de Snippets ou atalhos para criação de Controllers e métodos em aplicações Node.js com Express.

# Ideias de projetos para praticar

Algumas ideias de projetos simples para reforçar o aprendizado do Node/Express

1. Criar projetos de backend, como as APIs. Por exemplo, uma API para uma biblioteca ou para controlar cadastros de clientes e produtos em estoque. Veja uma explicação simplificada do que é uma API: https://www.instagram.com/p/CjLnNyAOj7p/

2. Fazer uma aplicação de mensagens instantâneas com Node e Express.

# DINHEIRO NA PROGRAMAÇÃO

**"A maneira de começar, é parar de falar e começar a fazer".**
*Walt Disney*

Depois de atingir um certo nível no seu desenvolvimento como Programador(a), é bem provável que você vá querer rentabilizar essa sua nova habilidade.

Neste capítulo eu vou falar sobre 3 aspectos que vão te ajudar nisso.

Um aviso, na parte em vídeo falamos detalhadamente sobre carreira mostrando como criar seu portfólio, aplicar para vagas, entrevistando um contador para falar como se legalizar como programador(a) PJ e uma recrutadora para falar como mandar bem em entrevistas.

Então, depois de ler esse capítulo, vá para a parte em vídeo ver o módulo especial de carreira.

## Nível para começar a trabalhar com programação

A principal pergunta que temos assim que começamos na programação é:

**Qual nível eu preciso atingir para começar a ganhar dinheiro com programação?**

Quando prestamos um serviço, estamos usando as nossas habilidades para realizar uma determinada ação que vai beneficiar quem está nos contratando.

Então a resposta mais simples é, você precisa ter aprendido o suficiente para que as suas habilidades sejam compatíveis com uma entrega de qualidade do serviço que você vai prestar.

Vou dar um exemplo: se você está oferecendo a criação de uma Landing Page (página de vendas ou de captura de emails), em geral você vai precisar ter dominado: HTML 5, CSS e JavaScript, e de preferência, Bootstrap também.

Não existe um determinado conjunto fixo de coisas que você precisa aprender para poder prestar serviços como programador(a), tudo é relativo ao serviço e as habilidades necessárias para entregar ele com qualidade.

A segunda pergunta que deve ter disparado no seu cérebro:

**Léo, como eu vou saber se consigo entregar o serviço com qualidade?**

Uma forma segura e ética de testar as suas habilidades é: Criar um projeto pessoal simulando um trabalho do mundo real (você sendo seu primeiro cliente).

No exemplo da Landing Page, visite por exemplo um site de freelancer, como o freelancer.com, veja demandas reais de criação de landing page e tente fazer de forma independente, para verificar se conseguiria entregar aquele trabalho.

Ou simplesmente, veja uma página que te agrade e tente replicar ela do zero usando as tecnologias que aprendeu.

Dessa forma, você vai conseguir identificar os pontos que precisam ser ajustados nas suas habilidades antes de oferecer esses trabalhos.

Um efeito colateral interessante, é que fazer isto vai te dar muita autoconfiança, por saber que já teve sucesso produzindo algo de qualidade que seria entregável a um cliente real.

# Como ser “contratável”

Antes de buscar jobs na programação, é importante você entender o quão contratável você está nesse momento, então me permita te guiar por essa metáfora:

Imagine que é sexta-feira à noite, você quer comer uma bela pizza mas está sem ideia de onde pedir, aí você abre um aplicativo para pedir comida e vê uma infinidade de pizzarias.

**Como você decide qual pizza escolher?**

Provavelmente vai buscar o melhor custo benefício, ou seja, a melhor pizza dentro do melhor preço, então você começa a olhar pizzaria por pizzaria.

Em algumas, as fotos são mal feitas, a descrição pouco atraente, com reviews suspeitos e o preço mais ou menos, outras, tem belas fotos, uma descrição interessante, bons reviews e um preço legal.

Você seleciona as melhores, depois para filtrar entre elas, você manda algumas mensagens pelo aplicativo para tirar algumas dúvidas, por exemplo, se é possível tirar a cebola (quem gosta de cebola afinal?).

Provavelmente, você vai escolher uma pizzaria que se apresente de forma interessante, que tenha respondido de forma adequada, que tenha bons reviews, que tenha as características que você deseja e com um preço dentro do orçamento.

De certa forma, também é assim que funciona a seleção de uma pessoa para uma equipe, o departamento de RH analisa vários perfis no LinkedIn (ou banco de talentos), vê quais deles se apresentam de forma interessante, quais tem as características desejadas (habilidades), quais tem bons reviews (depoimentos de colegas de trabalho) e depois chamam essas pessoas para o processo seletivo.

No processo seletivo eles vão verificar se as características anunciadas no perfil dos candidatos eram reais, colocando as habilidades deles à prova.

Depois, analisam quem se saiu melhor, seja em testes técnicos ou comportamentais (baseado nas respostas dadas, assim como no caso da pizzaria) e depois finalizam a contratação do(a) que tem o melhor custo-benefício.

A pergunta que você deve fazer então é: **Como ser um(a) programador(a) contratável (assim como a pizzaria vencedora)?**

Você precisa:

- Desenvolver as habilidades (como está fazendo nesse treinamento)
- Se apresentar de forma atraente (ensinamos na parte em vídeo como fazer isso no LinkedIn e GitHub)
- Se preparar para ter um bom desempenho no processo seletivo (damos dicas importantíssimas sobre isso na parte em vídeo, no módulo de carreira)

Então eu te faço a seguinte pergunta: "Você seria a pizzaria escolhida na Sexta-Feira a noite?" Se não, o que você vai fazer agora para se tornar mais atraente para o mercado?

## CLT é um caminho, mas não é o único

O mais típico, é os programadores dominarem as habilidades necessárias e entrarem em uma empresa para prestar serviços como CLT (ou PJ em um formato semelhante ao CLT), mas esse não é o único caminho.

Então aqui eu gostaria de abordar algumas formas alternativas de ganhar dinheiro com programação que podem abrir sua mente para novas realidades.

### Freelancer

Depois de CLT, ser programador(a) Freelancer é a segunda forma mais conhecida de ganhar dinheiro com programação, nesta modalidade você:

Realiza serviços de programação de curta ou média duração sem um contrato de trabalho (a longo prazo) para pessoas e empresas.

Normalmente, nesses trabalhos você vai criar sites, web Apps ou melhorar sistemas já existentes.

É uma forma de ganhar dinheiro interessante, porque você estipula o seu preço e tempo de entrega junto com o cliente, podendo ser mais lucrativo do que o trabalho fixo.

**Prós**:

- Mais liberdade (trabalhe no seu horário)
- Projetos diferentes sempre (o que expande sua visão)
- Relação mais horizontal (o cliente é seu cliente, não seu chefe)

**Contras**:

- Menos segurança (se não conseguir freelas, não ganha dinheiro)
- Trabalho mais isolado (em geral, não vai ter colegas de trabalho)
- Precisa de habilidades extras (como aquisição de clientes, autogestão e etc)

Eu trabalhei alguns anos como freelancer enquanto viajava como nômade digital, para mim, foi uma experiência incrível que me ajudou a evoluir muito.

O freelancer também pode servir como uma renda extra caso você já tenha um trabalho fixo, então eu realmente recomendo que você tente.

## Venda de templates / plugins

Uma modalidade pouco explorada no Brasil, é a criação e venda de templates prontos em sites como http://themeforest.net.

Imagine o seguinte: você quer criar um site lindão para o seu novo portfólio mas está sem tempo ou não tem as habilidades que precisaria, o que você faz?

Busca por um template pré pronto, compra por alguns dólares e depois edita ele como quiser. Bem útil né?

Tem pessoas fazendo 100k reais por mês com vendas de templates, que podem ser templates para landing pages, blogs e etc.

Uma observação, um aluno do OneBitCode, relatou que fez 300k reais com a venda de um template simples em um ano! Sim, é algo totalmente possível.

Como citei acima, o Theme Forest é o maior ecossistema para venda desse tipo de produto, então se você tem boas ideias e já domina bem o Frontend, vale a pena investir na pesquisa (do que já existe) e no desenvolvimento de algo atraente para vender por lá.

**Prós**

- Possibilidade de uma rentabilidade mais alta por ser escalável
- Você desenvolve no seu tempo
- Seus templates viram um portfólio para você

**Contras**

- Você precisa se destacar da concorrência
- Não existe garantia nenhuma de rentabilidade

Minha recomendação, vale a pena analisar e investir um tempo nisso.

## Criação de um SAAS

Primeiro, vamos definir o que é um SAAS (Software as a Service), o SASS é um software que permite que os usuários realizem determinado serviço útil (resolve um problema).

Um exemplo, o Trello é um SAAS, nele é possível organizar tarefas. Ele tem um plano gratuito, mas também tem uma versão mais completa paga que gera muito retorno.

Um exemplo de SAAS brasileiro é a Hotmart, que é uma plataforma que facilita a venda de infoprodutos (como este treinamento) através do seu software (WebAPP + Mobile).

Inclusive, os fundadores da Hotmart, eram programadores e isso os possibilitou criar esse SAAS bilionário.

Mas nem só de bilhões vivem os SAAS, não é necessário que o seu serviço seja gigantesco para que você consiga lucrar e viver dele.

Até mesmo uma ferramenta simples de edição de imagens, uma ferramenta de gestão de finanças ou qualquer coisa assim, se bem feita e com um marketing adequado, pode te dar a liberdade que procura.

**Prós**

- Uma vez desenvolvido, a quantidade de trabalho pode diminuir muito
- Possibilidade alta de lucro
- Você trabalha no seu tempo
- Na pior das hipóteses, a ferramenta vira um super portfólio para você

**Contras**

- Sem garantia alguma de lucro (inclusive, pode gerar gastos)
- Pode precisar de habilidades extras (Marketing e etc)

Eu recomendo que você fique mais atento(a) às suas necessidades do dia a dia e analise se existe alguma ferramenta que poderia facilitar a sua vida, caso você não encontre uma satisfatória, pesquise se outras pessoas também precisam dessa ferramenta, caso sim, invista um tempo em desenvolvê-la.

Estas são apenas algumas formas diferentes de ganhar dinheiro com programação, existem outras que eu expliquei na parte em vídeo, vale a pena ver.

Agora que você já tem uma visão geral do que precisa para começar a ganhar dinheiro com programação, como ser mais contratável (assim como a pizzaria vencedora) e que CLT não é o único caminho, recomendo que você vá para a parte em vídeo, para se aprofundar em tudo isso.

# A CONQUISTA

***"O que faz de alguém um vencedor não é apenas cruzar a linha de chegada, mas também o caminho percorrido até à vitória".***

Parabéns por chegar até aqui Programador(a), você enfrentou os desafios do caminho e conseguiu finalizar o guia Full Stack JavaScript do OneBitCode com sucesso 🎉

Tenho certeza que você evoluiu demais na jornada, tanto no guia como também no treinamento em vídeo (caso ainda não tenha finalizado ele, foque suas energias em finalizar).

Se você finalizou os dois, já está pronto para usar as suas novas habilidades de programação para dominar o mundo...digo, contribuir com o mundo 😅

Uma observação importante, esse guia pode e deve ser relido sempre que necessário, é natural absorvermos qualquer coisa ainda melhor na segunda leitura.

## Continue evoluindo

Como você sabe, a programação está em constante evolução e agora que você já tem uma base forte (graças ao guia e ao treinamento em vídeo), não pare de evoluir.

Aqui tem algumas sugestões para se especializar com o OneBitCode:

- Treinamento de testes automatizados (especialidade altamente pedida no mercado)
- Treinamento de docker (especialidade altamente pedida no mercado)
- Livro Programador Nômade (como viver de freelancer enquanto viaja o mundo)

- 30 desafios de JavaScript (para aumentar seu domínio sobre lógica de programação)

Aproveite que você tem desconto especial por sem aluno(a) e acrescente esses conhecimentos extras para se destacar no mercado de trabalho.

## 2 dicas importantes

Agora que você finalizou o treinamento, mantenha seu portfólio constantemente atualizado, sempre adicionando novos projetos pessoais e melhorias aos projetos já disponíveis nele.

Isso vai te ajudar a alcançar oportunidades cada vez melhores, então, mantenha-se em “movimento”.

Outro ponto importante, agora que você já desenvolveu os seus conhecimentos, uma forma incrível de reforçar eles é “devolvendo para comunidade”.

Ou seja, esteja na comunidade do OneBitCode e ajude os outros alunos com as dúvidas, isso vai te ajudar a se destacar no mundo da programação, vai manter seus conhecimentos afiados e vai te dar um sentimento de satisfação e felicidade por poder ajudar.

## Hora da glória

É sempre importante comemorarmos as nossas vitórias, isso nos ajuda a nos manter dedicados e buscando sempre melhorar.

Tire uma foto ou um print da imagem a seguir e poste nas suas redes sociais marcando o *@onebitcode* para mostrar a sua nova conquista ao mundo:

Obrigado por confiar no OneBitCode e parabéns pelo grande progresso, continuamos juntos na comunidade e em outros treinamentos.

***Desejamos que a habilidade de programar te leve mais perto de todos os seus objetivos***.

Equipe OneBitCode.

onebitcode