

Boas-vindas!

Esteja confortável, pegue uma água e se acomode em um local tranquilo que já começamos.

Como você **chega**?

1



2



3



Esta aula será

- gravada

Apresentação da equipe



Diogo Magliano
professor



Carlos Rafael
tutor

Atividade de integração

Quebra-gelo

Vamos nos conhecer?

O objetivo deste espaço é que vocês conheçam uns aos outros e discutam coisas que são do interesse de vocês.



Duração: 10 minutos



CODERHOUSE

CARÔMETRO

Quem são os estudantes Coder?

Preencha o forms que está no QR Code ou link. Ele nos ajudará a formatar as próximas aulas e tentar direcionar melhor para os interesses de vocês.





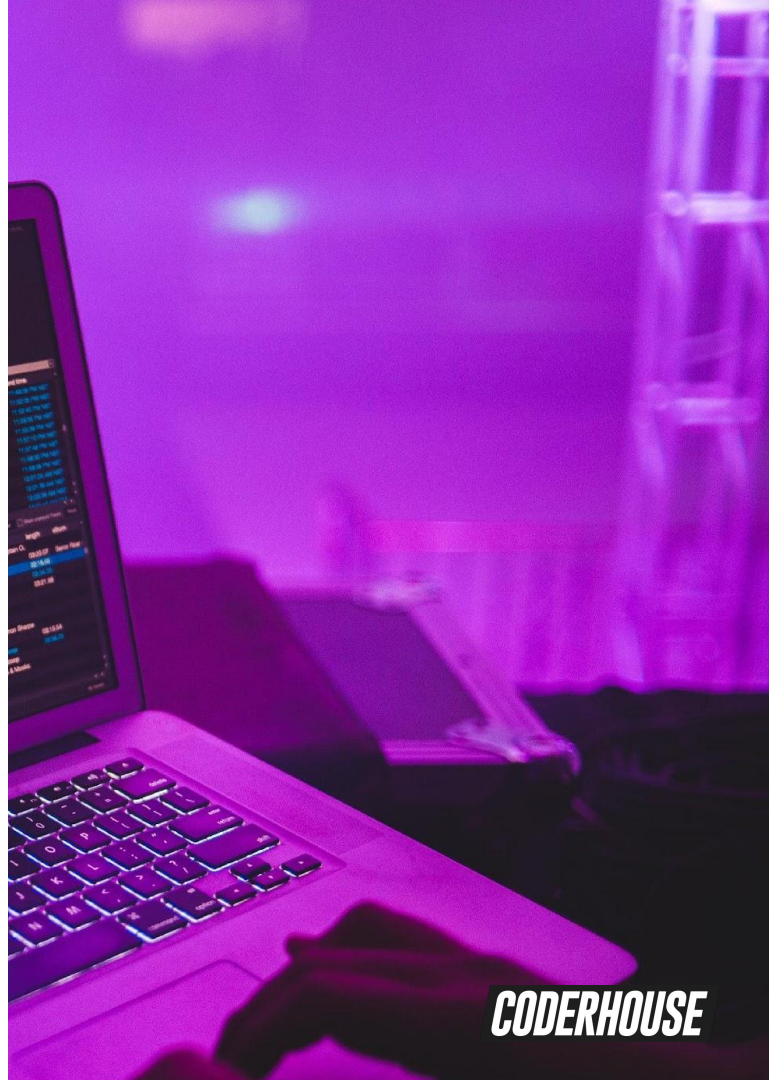
O que você precisa saber
Antes de começar

Acordos e compromissos

ACORDOS E COMPROMISSOS

Ferramentas

- ✓ Pegue **água, chá e café**.
- ✓ No **Drive** você terá todos os **arquivos e slides**.
- ✓ Todas as aulas serão **gravadas e compartilhadas**.
- ✓ **Não fique com dúvidas!** Procure no chat do Zoom o [TUTOR] disponível na aula e envie suas perguntas!
- ✓ Não se esqueçam de **avaliar as aulas**.



CODERHOUSE

ACORDOS E COMPROMISSOS

Ambiente

- ✓ Encontre o seu espaço e crie o momento oportuno para **desfrutar do aprendizado**.
- ✓ Evite dispositivos e aplicativos que possam **roubar sua atenção**.
- ✓ Câmeras abertas sempre que possível.
- ✓ Todas as aulas serão **uma construção conjunta!**

#DemocratizandoAEducação

CoderBolsa

Requisitos para certificação

- Ter 85% de presença nas aulas ao vivo
- Entregar os desafios dentro do prazo e no formato determinados
- Aprovação do projeto final com nota igual ou superior a 7

Não cumpri os requisitos, e agora?

- A certificação só é emitida com o cumprimento dos requisitos acima
- Entre em contato com o time de Experiência do estudante.

 WhatsApp



estudantes@coderhouse.com

Passando a limpo

Desafios e Entregas

DESAFIOS E ENTREGAS

O que são?



Desafios obrigatórios

Desafios de **entrega compulsória** para beneficiários da Coderbolsa. Devem ser inseridos na plataforma em até 7 dias após a aula.



Entregas do Projeto Final

Etapas do projeto final com **entrega compulsória** para beneficiários da Coderbolsa. Devem ser inseridos na plataforma em até **7 dias** após a aula.

DESAFIOS E ENTREGAS

O que são?



Desafios complementares

Desafios que complementam a aprendizagem. São optativos, mas contam ponto no Top 10 quando inseridos na plataforma.



Atividades em sala

Ajudam a por em prática os conceitos e a teoria vista em aula. Não devem ser inseridos na plataforma.

DESAFIOS E ENTREGAS

Projeto final

O Projeto Final se constrói a partir das **entregas parciais** que se realizam aula a aula. Vai sendo criado à medida que o estudante vai inserindo as entregas na plataforma..

O objetivo é que cada estudante possa utilizar seu **Projeto Final como parte do seu portfólio pessoal**.

Deve ser inserido na plataforma e você terá até **20 dias** corridos **a partir da finalização do curso para inseri-lo na plataforma**. Após esse período o botão de entrega é inativado.



Qual é o nosso
Projeto Final?



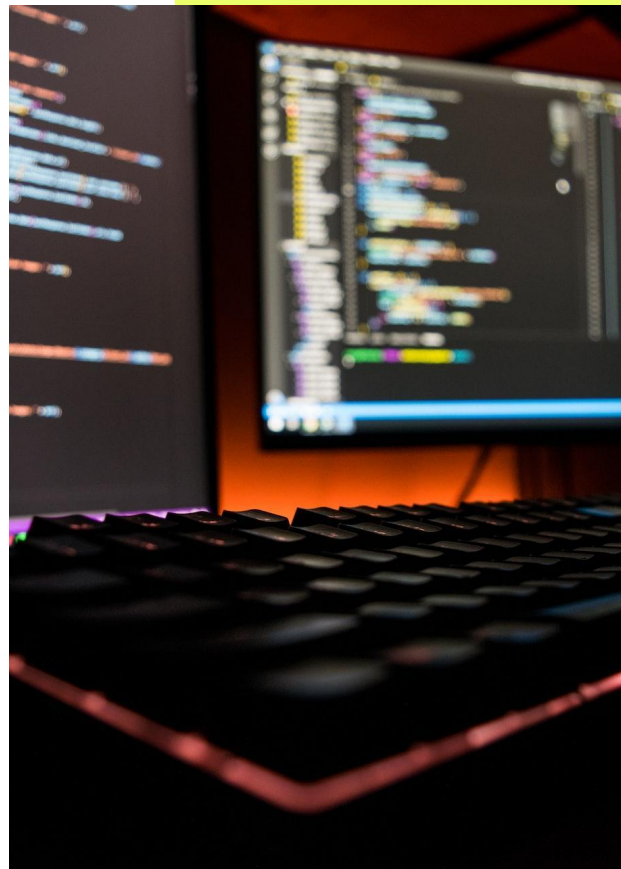
PROJETO FINAL

Descrição do projeto final

Descrição:

Você desenvolverá o back-end de uma aplicação de e-commerce para poder vender produtos em uma categoria de sua escolha.

O servidor será baseado em um design em camadas, orientado para MVC e seu código conterá as estruturas de programação mais sólidas da linguagem ECMAScript.

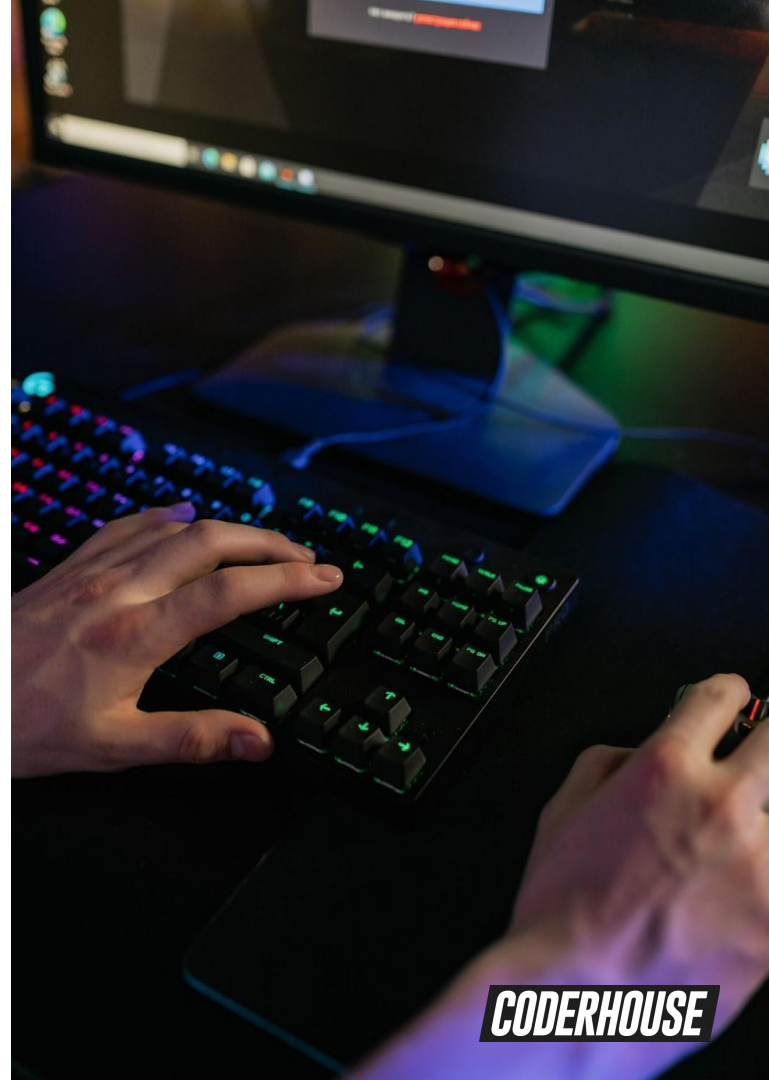


CODERHOUSE

GRUPOS

Projeto final

- ✓ O projeto final, as entregas parciais do projeto final e os desafios entregáveis **são realizados em grupo**
- ✓ Os temas são **escolhidos por vocês** com base em assuntos atuais e relevantes
- ✓ **Não é permitido fazer trabalho individual.**
Queremos simular ambientes reais do mercado de trabalho!
- ✓ Grupos são pré definidos e possuem **5 pessoas**
- ✓ Podem haver mudanças de grupos **até a aula 3, fale com seu tutor.**
- ✓ Todos os estudantes de um grupo tem o mesmo tutor
- ✓ Todos os integrantes podem divulgar o projeto como **portfólio**



Passando a limpo

Equipe de aula

Funções da equipe da **aula**.

Professor

- ✓ Planejar/preparar as aulas
- ✓ Liderar as aulas ao vivo
- ✓ Corrigir os projetos finais

Tutores

- ✓ Quem te acompanha durante toda a jornada do curso
- ✓ Responsável por marcar presenças
- ✓ Corrigem os desafios entregáveis e as entregas intermediárias do projeto

O que não é função do tutor?

- Realizar **mentorias individuais ou em grupo** com seus tutorandos, aprofundando em pontos da aula;
- Responder **imediatamente** uma pergunta enviada assincronamente;
- Corrigir desafios **complementares**;
- Resolver questões administrativas, que envolvem plataforma ou Coderbolsa – isso é tratado com o time de Experiência do Estudante :)

Passando a limpo

Feedbacks

Qual a importância de preencher o **feedback**?

É habilitado após a aula

- Através dele é possível avaliar professor, tutor, conteúdo, plataforma e chat.
- Permite que possamos agir enquanto a experiência está acontecendo.
- Soma pontos no Top 10.

Passando a limpo


Canais de comunicação

Canais de comunicação

Professores, tutores e colegas: chat da plataforma

Para dúvidas sobre conteúdo, grupos, desafios, prazos e materiais disponibilizados.

Time de Experiência do estudante Coderhouse:

 WhatsApp

 estudantes@coderhouse.com

Para problemas com a plataforma, CoderBolsa, justificativas de ausência, mudança de turma, cancelamentos.



Break

5 minutos e voltamos!





Break

10 minutos e voltamos!



Aula 01. BACKEND

Princípios de programação Back End

Objetivos da aula

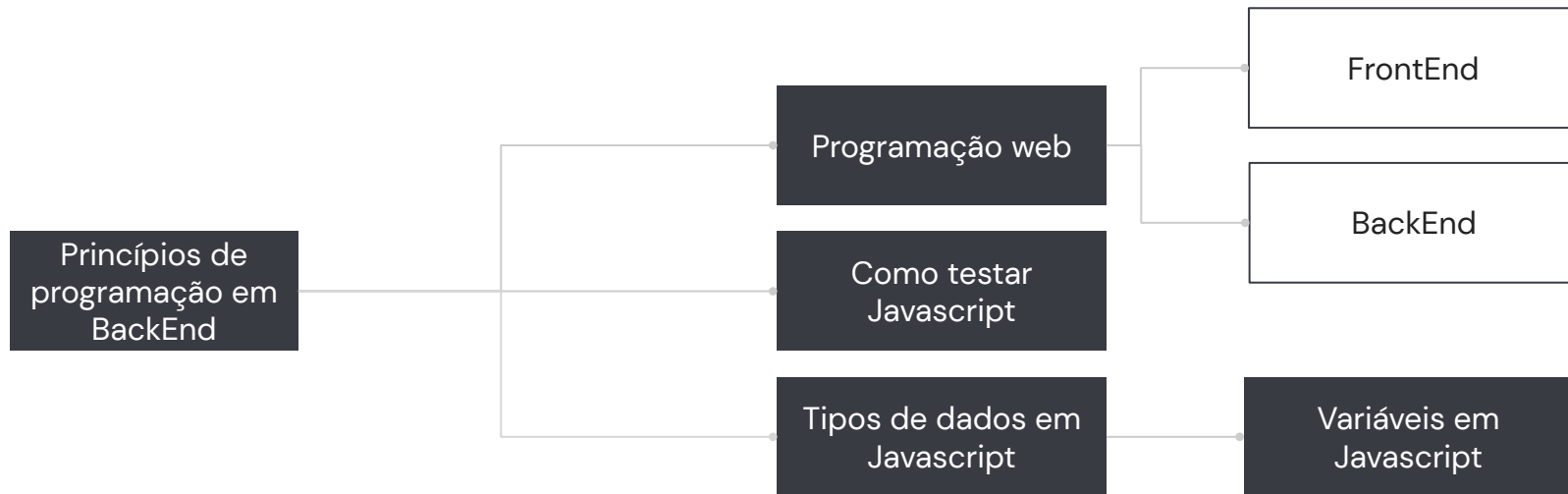


Conhecer as diferenças entre programação Front end e Back end



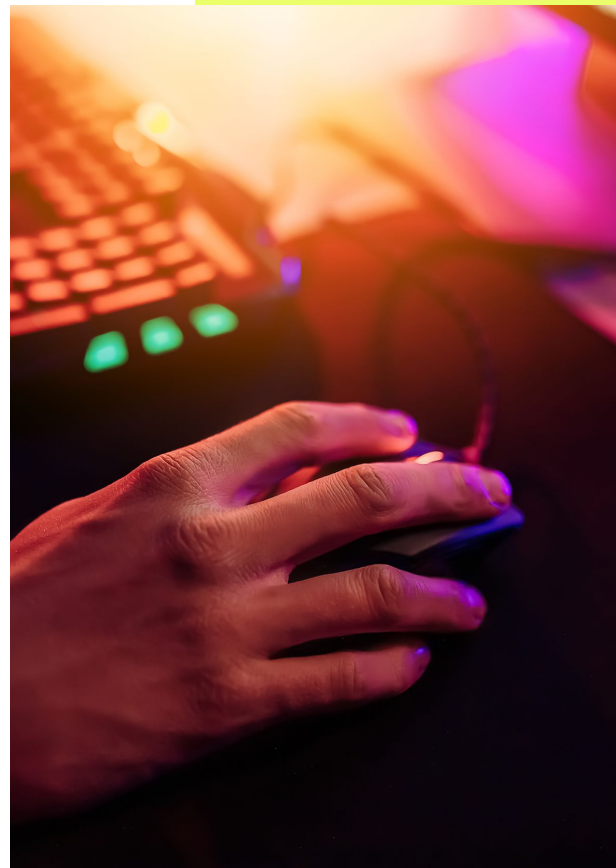
Familiarizar-se com os fundamentos da programação usando Javascript e o MERN Stack

MAPA DE CONCEITOS



O que é programação web?

- ✓ Quando falamos de desenvolvimento web, nos referimos à construção e manutenção desses sites que podem ser encontrados na World Wide Web (www) e suas redes derivadas.
- ✓ Originalmente era baseado apenas no desenvolvimento de páginas estáticas. No entanto, com o passar do tempo, estes exigiram mais dinamismo, ao ponto de começarem a desenvolver não só páginas estáticas, mas também websites completos ou mesmo aplicações web.



Os dois lados da mesma moeda

À medida que as necessidades de desenvolvimento web crescem, surge a necessidade de começar a separar o desenvolvimento em dois aspectos importantes:

FrontEnd e BackEnd.

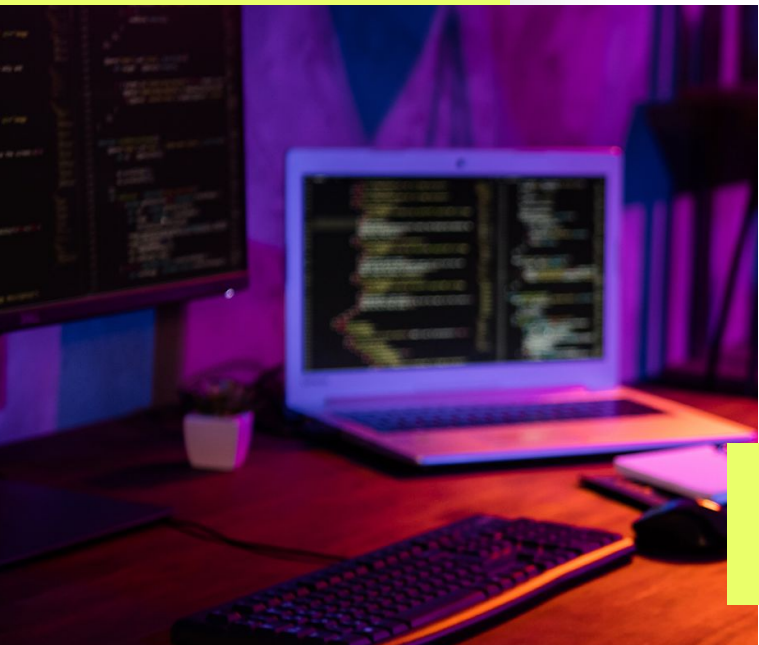
FrontEnd x BackEnd

O **FrontEnd** inclui toda a parte visual e interação direta com o usuário. Sua finalidade é dar a melhor impressão ao usuário sobre a página ou aplicativo, por exemplo:

- ✓ Imagens
- ✓ Cores
- ✓ Botões
- ✓ Interações

O **BackEnd** inclui toda a parte lógica e de gestão da informação. Ou seja, a base lógica da nossa página ou aplicativo. O usuário **NÃO** sabe o que está por trás disso. Exemplos:

- ✓ Armazenamento de informações
- ✓ Cálculos complexos
- ✓ Os servidores onde as páginas estão publicadas
- ✓ Gestão de informações em geral



- ✓ Enquanto o **FrontEnd** é focado na **experiência do usuário**, torná-la o mais apresentável e amigável possível!
- ✓ O **BackEnd** possui uma **estrutura interna** para que tudo funcione **corretamente**. Não deixe o usuário vê-lo!

FrontEnd x BackEnd



Muitas vezes ouvimos outras pessoas dizerem “*Eu odeio BackEnd*” ou “*FrontEnd é chato*”. Isso tem polarizado a comunidade de desenvolvimento, pensando que o mundo estaria melhor se um dos dois não existisse.

FALSO!

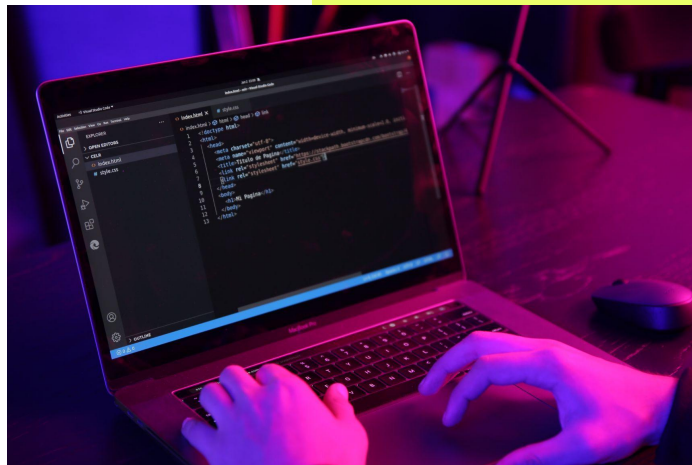
Ambos precisam um do outro para poder estruturar páginas e aplicativos. Dessa forma, poderemos resolver problemas do mundo real com soluções tecnológicas.

Temos que fazer as pazes! 💛

Stack MERN

O que é um “Stack”?

- ✓ São algumas tecnologias que, juntas, nos darão a **possibilidade de desenvolver sistemas completos**, pela sua máxima compatibilidade.
- ✓ Poderíamos dizer que é a forma como o **FrontEnd e o BackEnd fazem as pazes**, já que trabalham juntos. **Neste curso trabalharemos no MERN Stack.**



Componentes do **MERN Stack**



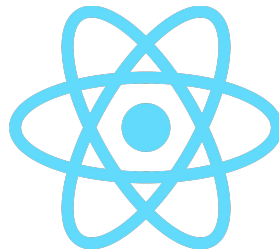
MongoDB

Base de dados não
relacional



ExpressJS

Framework para
criar servidores em
NodeJS



ReactJS

Biblioteca para
desenvolver
interfaces de
usuário



NodeJS

Interpretador que
permite execução
de códigos
JavaScript

Como testar Javascript?

Cliente web

Usamos no console do navegador, não precisamos instalar nada, pois todo o mecanismo fica no navegador

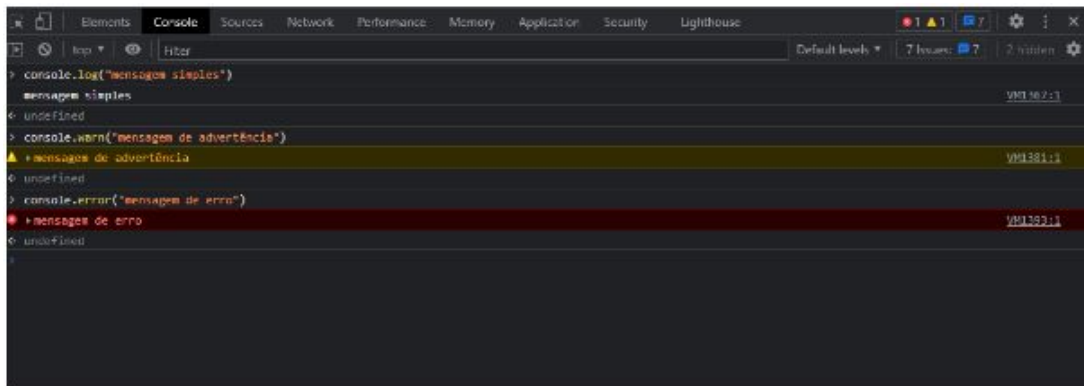
Node JS

Devemos instalá-lo, pois é usado fora do navegador, para que possamos escrever Javascript diretamente de nosso computador. Este é o que usaremos ao longo do curso.

Cliente web

Ele é usado em qualquer navegador. Alguns dos comandos que você mais usará e verá ao longo de uma depuração em um cliente web são:

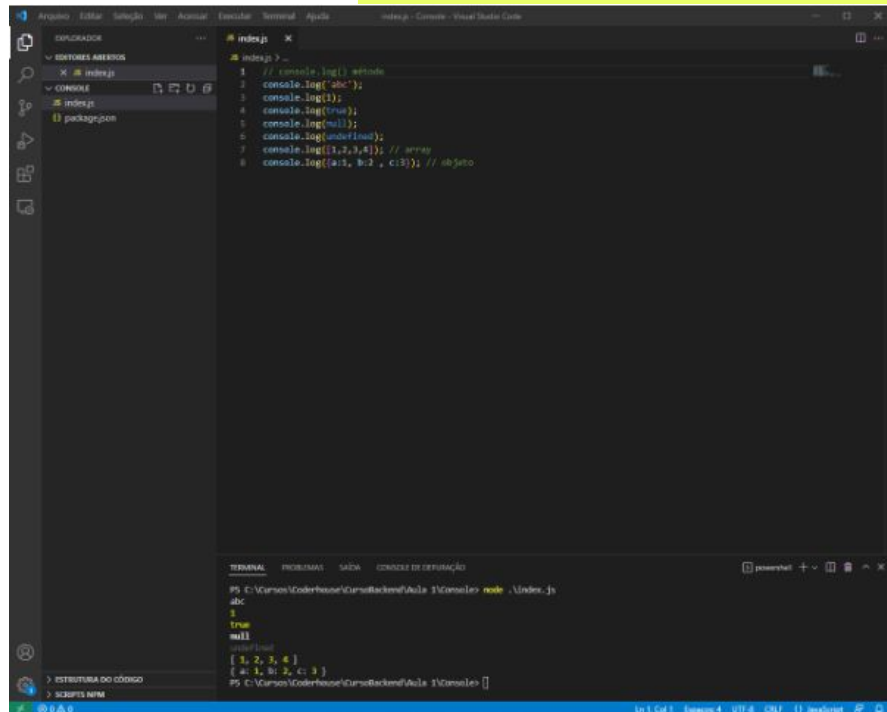
- ✓ `console.log("texto")` mostrará um texto simples
- ✓ `console.warn("texto")` mostrará uma advertência
- ✓ `console.error("texto")` mostrará um erro
- ✓ `console.clear()` limpa o console para evitar poluição de código



Node JS

Quando trabalhamos no backend, não temos navegador, por isso precisamos de alguma tecnologia que nos permita executar o código Javascript, sem precisar abri-lo no navegador. É quando utilizamos o Node js, para isso, é preciso levar em consideração:

- ✓ O Node JS construirá um ambiente completo para testar nossas funções.
- ✓ Ele deve ser executado a partir de uma CLI para poder visualizar o progresso do código.
- ✓ Normalmente, usaremos a CLI do Visual Studio Code.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a file named `index.js`. The main editor area displays the content of `index.js`:

```
1 // console.log() método
2 console.log('abc');
3 console.log(1);
4 console.log(true);
5 console.log(null);
6 console.log(undefined);
7 console.log([1,2,3,4]); // array
8 console.log({a:1, b:2, c:3}); // objeto
```

The Terminal pane at the bottom shows the output of the code execution:

```
P5 C:\Curso\CoderHouse\CursoBackend\Aula 1\console node .\index.js
abc
1
true
null
[1,2,3,4]
{a:1, b:2, c:3}
```



PARA PENSAR

Quando você usaria um cliente da Web e quando usaria o Node js ao usar o Javascript?

Um substitui o outro ou podemos considerá-los complementares?

Compartilhe no chat suas opiniões

Tipos de dados em JavaScript

Tipo de dado

É o atributo que especifica a classe de dados que a variável armazena.

Especifica com o que vamos trabalhar, para que o computador reconheça quais operações pode fazer com ele.

Tipo Primitivo: Incluem strings de texto (String), variáveis booleanas cujo valor pode ser verdadeiro ou falso (Boolean) e números (Number). Além disso, existem dois tipos primitivos especiais que são nulos e indefinidos. A cópia é por valor.

Tipo de Objeto: Incluem objetos (Object), arrays (Array) e funções. A cópia é para referência.

		Nome	Descrição
Tipos de dados em JavaScript	Tipos primitivos	String	Cadeias de textos
		Number	Valores numéricos
		Booleans	True/false
		Null	Tipo especial, contém null
		Undefined	Tipo especial, contém undefined
	Tipos objeto	Tipos predefinidos de Javascript	Date (data) RegExp (expressões regulares) Error (dados de erro)
		Tipos definidos pelo programador/usuário	Funções simples Classes
		Arrays	Série de elementos que formam uma matriz ou vetor. Vamos considerá-lo um objeto especial que necessita de métodos
		Objetos Espaciais	Objeto Global
			Objeto protótipo
			Outros

Variáveis em JavaScript

Agora que entendemos quais são os tipos de dados, é importante entender onde os dados serão armazenados. **Uma variável é um espaço de memória reservado pelo computador para armazenar dados.**

Como o próprio nome diz, **uma variável pode mudar seu valor se o programa precisar.** Isso permite a reutilização de uma única variável, para os casos que forem necessários.

Perguntas?

Como foi a aula?

1

Que bom

O que foi super legal na aula e podemos sempre trazer para as próximas?

2

Que pena

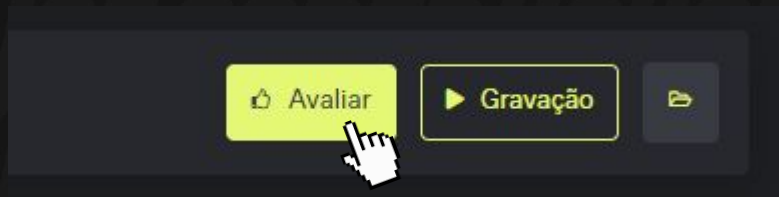
O que você acha que não funcionou bem e precisamos melhorar?

3

Que tal

Qual sugestão deveríamos tentar em próximas aulas?

O que você achou da aula?



Seu feedback vale pontos para o Top 10!! 😎



Deixe sua opinião!

1. Acesse a plataforma
2. Vá na aula do dia
3. Clique em **Avaliar**

Resumo da aula de hoje

- ✓ Relação e diferença entre FrontEnd e BackEnd
- ✓ Fazendo as pazes: Stack MERN
- ✓ Como testar Javascript
- ✓ Tipo de dados de Javascript
- ✓ Variáveis de Javascript



**Obrigado por estudar
conosco!**