**REACT**

1. **CRIAÇÃO DE PROJETO (Com Vite)**
2. **Comando:** npm create vite@latest
3. **Como executar?**

* cd Projeto;
* npm i (instalar as dependências);
* npm run dev;

1. **CONFIGURAÇÕES E EXTENSÕES RECOMENDADAS**
2. **Configurar o Emmet do VsCode com** Javascript : javascriptreact
3. **Extensão:** ES7 React
4. **CRIAÇÃO DE COMPONENTES**
5. **Pasta:** components

* Em **src** criar a pasta **componentes**;
* Nessa pasta que se cria os componentes em extensão **jsx;**

1. **Padrão Primeira Letra Maiuscula com CamelCase**
2. **Sempre se criar o componente e o exporta**
3. **Ao invés de escrever o código abaixo basta digitar:** rafce (da extensão – Cria o componente automaticamente, pode apagar o import react)

**const FirstComponent = () => {**

**return (**

**<div>**

**<h2>Meu primeiro Component no REACT</h2>**

**</div>**

**);**

**}**

**export default FirstComponent;**

1. **NOMES DO HTML QUE COINCIDEM COM JAVASCIPRT**
2. **class vira className**
3. **O mesmo acontece com outros recursos, como o for**
4. **COMENTÁRIOS (no js:** // **e no jsx:** {/\* \*/}
5. **TEMPLATE EXPRESSIONS**
6. **Usar js no meio do JSX**

* { códifo em js }

**const TemplateExpression = () => {**

**const name = "Guilherme Terra";**

**const obj = {**

**age: 20,**

**job: "Desenvolvedor"**

**}**

**return (**

**<div>**

**<p>A soma é: { 2 + 2 }</p>**

**<p>Olá! me chamo { name }</p>**

**<p>Tenho { obj.age } anos e sou um { obj.job }</p>**

**</div>**

**)**

**}**

**export default TemplateExpression**

1. **Evento de Clique**
2. **Propriedade onClick**

* onClick = { algumaCoisa } ;
* Alguma coisa normalmente é função;
* Note q usa { }, então quando pressionado ele roda o JS;

**const Event = () => {**

**const handleClick = () => {**

**console.log("Clicou no segundo btn!");**

**}**

**return (**

**<div>**

**<div>**

**<button onClick={ () => { console.log("Ocorreu um clique no btn!") } }>Clique Aqui!</button>**

**<button onClick={ () => { handleClick() } }>Clique Aqui 2!</button>**

**</div>**

**</div>**

**)**

**}**

**export default Event**

1. **RENDERIZAÇÃO SIMPLES**
2. **Uso de If para controler o que será carregado na tela**

**const Event = () => {**

**const handleClick = () => {**

**console.log("Clicou no segundo btn!");**

**}**

**const renderingSomething = ( x ) => {**

**if( x ) {**

**return <h1>Texto de Renderização 1</h1>;**

**}else{**

**return <h1>Texto de Renderização 2</h1>;**

**}**

**}**

**return (**

**<div>**

**<div>**

**<button onClick={ () => { console.log("Ocorreu um clique no btn!") } }>Clique Aqui!</button>**

**<button onClick={ () => { handleClick() } }>Clique Aqui 2!</button>**

**{ renderingSomething(true) }**

**{ renderingSomething(false) }**

**</div>**

**</div>**

**)**

**}**

**export default Event**

1. **IMAGENS**
2. **Abordagens**

* Através de public;
* Através de assets;

1. **Public**

* Põe imagem em public e chama ela com src=”/nomeDaImagem”;

1. **Assets (+ comum)**

* Põe imagem em Assets;
* Pra chamar é necessário importar;
* Import nomeQueVcQuiser from “chaminhoDaimagem”;
* Src = { nomeQueVcQuiser };
* Permite reutilização;
* Usa JS pra meio que criar uma variável que vai guardar o caminho;

1. **HOOKS**
2. **O que são Hooks?**

* São Recursos do React;
* Caracterizado pro começar com a palavra **use;**
* Já existem alguns prontos e podemos criar também;
* **Hooks** precisam ser importados;
* **Custom Hook =** É um Hook que vc mesmo cria (É só uma nomenclatura de mercado);

1. **useState**

* É o Hook mais comum e mais usado;
* Permite gerenciar o estado de dados;
* É como se fosse um getter e setter;
* **Quando se usa variáveis ele não re-renderiza / atualiza os componentes após a troca;**
* O Hook resolve isso;

1. **OBSERVAÇÃO PROPRIA: TEM PROBLEMAS DE SINCRONIA NA VELOCIDADE DO USESTATE E CONSOLE.LOG**