



# FIAP

GRADUAÇÃO

45697056



TDS

# Front-End Design Engineering

Prof. Alexandre Carlos [profalexandre.jesus@fiap.com.br](mailto:profalexandre.jesus@fiap.com.br)

Prof. Luís Carlos [lsilva@fiap.com.br](mailto:lsilva@fiap.com.br)





# Instalando bibliotecas

45697056



Para esta aula vamos aprender a usar a biblioteca [react-icons](#). que vamos conhecer hoje. Vamos instalar o Next e logo após a biblioteca.

\* vamos instalar o react-icons:

```
npm install react-icons --save
```





# Projeto com algumas novidades


45697056



Na aula de hoje vamos consolidar os conhecimentos já adquiridos e conhecer algumas novas características do Next e Typescript.

A primeira coisa que vamos nos ater é limpar o componente `page.tsx` e `globals.css`

```
export default function Home() {  
  return (  
    <>  
    </>  
  );  
}
```

src > app >  globals.css > ...

```
1  
2  
3  
4
```






# Projeto com algumas novidades

45697056



Neste exemplo vamos fazer toda a nossa estilização no arquivo `globals.css`, então vamos começar pelo reset da página.

```
src > app >  globals.css > ...  
1  *{  
2    margin: 0;  
3    padding: 0;  
4    box-sizing: border-box;  
5    font-family: Arial, Helvetica, sans-serif;  
6  }  
7
```





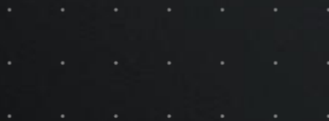
# Projeto com algumas novidades

45697056



A ideia é criar uma aplicação para criar pequenas anotações de tarefas. Vamos começar criando dentro da pasta **components** um Componente chamado **ListaTarefas.tsx**. Por enquanto ela vai ficar só com um parágrafo de teste.

```
1  export default function ListaTarefas(){
2
3
4      return(
5          <div>
6              <h1>Lista de Tarefas</h1>
7          </div>
8      )
9  }
```





# Projeto com algumas novidades

Teremos componentes que servirão como post'it de lembretes de tarefa. Crie um componente chamado **Tarefa.tsx**.

```
1 export default function Tarefas(){
2
3
4   return(
5     <div className="tarefa">
6       <h2>Relatório de Vendas</h2>
7       <p>Para: Departamento de vendas</p>
8       <p>Fazer: Levantar valores deste mês.</p>
9     </div>
10  )
11 }
```

```
.tarefa{
  background-color: #ffb;
  border: 2px solid #333;
  box-shadow: 5px 5px 5px #333;
  padding: 20px;
  text-align: center;
  width: 350px;
  height: 200px;
  margin: 10px;
  position: relative;

  h2,p{
    padding-bottom: 10px;
  }

  button{
    position: absolute;
    top: 10px;
    right: 10px;
  }
}
```

Não esqueça de fazer sua estilização no `globals.css`



# Projeto com algumas novidades

45697056  
■ ■ ■

Voltando para o `ListaTarefas.tsx`, vamos organizar para receber várias tarefas. Vamos testar com 3 tarefas para começar.

```
1  import Tarefa from "../Tarefa";
2
3
4  export default function ListaTarefas(){
5
6
7      return(
8          <div className="lista-tarefa">
9              <Tarefa/>
10             <Tarefa/>
11             <Tarefa/>
12          </div>
13      )
14  }
15
```

```
.lista-tarefa{
width: 100%;
min-height: 85vh;
padding: 20px;
background-color: #ccc;
display: flex;
flex-wrap: wrap;
justify-content: space-evenly;
}
```







# Projeto com algumas novidades

A ideia é criarmos novas tarefas, então vamos criar um state em `ListaTarefas` para guardar os dados das tarefas. Vamos criar o state e um map para carregar a tela.

```
export default function ListaTarefas(){  
  
  const [tarefas, setTarefas] = useState([  
    {titulo:'Lista de Pagamentos',  
      setor:'Dep. Vendas',  
      descricao:'Levantar os valores das vendas'},  
    {titulo:'Planilha de Salários',  
      setor:'Dep. Pessoal',  
      descricao:'Gerar Planilhas'}  
  ])  
  
  return(  
    <div className="lista-tarefa">  
      {tarefas.map((t,i)=>(  
        <Tarefa key={i} {...t}/>  
      ))}  
    </div>  
  )  
}
```



# Projeto com algumas novidades

45697056

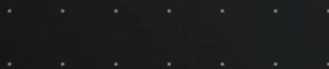


Crie um arquivo chamado `types.ts`. Vamos precisar destes types no projeto

```
export type TarefaProps={
  titulo:string;
  setor:string;
  descricao:string
}

export type RemoveProps = {
  remove: (num: string) => void;
}

export type ActionsProps = {
  add:(e:React.FormEvent<HTMLFormElement>)=>void;
  digit:(event:React.ChangeEvent<HTMLInputElement>)=>void;
}
```





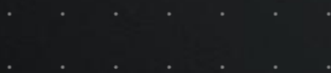
# Projeto com algumas novidades

45697056



Para funcionar precisamos receber os props em **Tarefa** para carregar os dados na tela.

```
export default function Tarefa({titulo, setor, descricao}:TarefaProps){  
  
  return(  
    <div className="tarefa">  
      <h2>{titulo}</h2>  
      <p>Para: {setor}</p>  
      <p>Fazer: {descricao}</p>  
    </div>  
  )  
}
```





# Projeto com algumas novidades

Começaremos criando uma função em `ListaTarefas`, para inserir uma tarefa nova, por enquanto só para validar a função.

```
45697056
■ ■ ■

const addTarefa = ()=>{

  const ntarefa = {
    titulo: 'Planilha de Salários',
    setor: 'Dep. Pessoal',
    descricao: 'Gerar Planilhas'
  }

  setTarefas([...tarefas, ntarefa])

}

return(
  <div className="lista-tarefa">
    {tarefas.map((t,i)=>(
      <Tarefa key={i} {...t}/>
    ))}
    <button onClick={addTarefa}>Adicionar</button>
  </div>
)
```



# Projeto com algumas novidades

45697056



Criaremos um formulário para inserir novas tarefas, crie um componente chamado `FormTarefas.tsx`.

```
import { ActionsProps, TarefaProps } from "@/types"

export default function FormTarefas({titulo,setor,descricao,add,digit}:TarefaProps&ActionsProps){

  return(
    <div className="form-tarefa">
      <form onSubmit={add}>
        <div>
          <input name="titulo" placeholder="Título"
            value={titulo} onChange={digit} />
        </div>
        <div>
          <input name="setor" placeholder="Setor"
            value={setor} onChange={digit} />
        </div>
        <div>
          <input name="descricao" placeholder="Descrição"
            value={descricao} onChange={digit} />
        </div>
        <button type="submit">Adicionar</button>
      </form>
    </div>
  )
}
```



# Projeto com algumas novidades

Para o formulário funcionar temos que criar um state para nova tarefa, mudar nossa função de adicionar tarefa, criar um para captura das informações do formulário e passar tudo por props.

```
export default function ListaTarefas(){

  const [tarefas, setTarefas] = useState<TarefaProps[]>([])

  const [tarefa, setTarefa] = useState({
    titulo: '',setor: '',descricao: ''
  })

  const addTarefa = (e:React.FormEvent<HTMLFormElement>)=>{
    e.preventDefault()
    setTarefas([...tarefas, tarefa])
    setTarefa({titulo: '',setor: '',descricao: ''})
  }

  const captura = (e: React.ChangeEvent<HTMLInputElement>): void => {
    const {name,value} = e.target
    setTarefa({ ...tarefa, [name]: value });
  }

  return(
    <div className="lista-tarefa">
      <FormTarefas {...tarefa} add={addTarefa} digit={captura}/>
    </div>
  )
}
```



# Projeto com algumas novidades

Agora um pouco de estilização no formulário, só para o vermos melhor.

```
.form-tarefa{
  width: 100%; padding: 20px; background-color: #993;

  form{
    width: 50%; margin: auto;

    input, textarea{
      padding: 5px; width: 80%; margin-bottom: 5px;
    }

    textarea{
      height: 80px;
    }

    button{
      padding: 10px; background-color: #3c3; color: #fff;
    }
  }
}
```



# Projeto com algumas novidades

Vamos fazer uma função para remover a tarefa. Em `ListaTarefas`, crie a função `remover tarefa` e passe como propriedade pelo `props`.

Estamos usando o título para ele conseguir referenciar o componente a ser excluído, enquanto não temos o id.

```
const removeTarefa = (tit:string)=>{
  let lista = tarefas
  lista = lista.filter(t=> t.titulo !== tit)
  setTarefas(lista)
}

return(
  <div className="lista-tarefa">
    <FormTarefas {...tarefa} add={addTarefa} digit={captura}/>
    {tarefas.map((t,i)=>(
      <Tarefa key={i} remove={removeTarefa} {...t}/>
    ))}
  </div>
)
}
```





# Projeto com algumas novidades

45697056



Temos que criar um botão para usar a função no componente **Tarefa**.

```
export default function Tarefa({titulo, setor, descricao,remove}:TarefaProps&RemoveProps){  
  
  return(  
    <div className="tarefa">  
      <button onClick={()=>remove(titulo)}>X</button>  
      <h2>{titulo}</h2>  
      <p>Para: {setor}</p>  
      <p>Fazer: {descricao}</p>  
    </div>  
  )  
}
```





# Projeto com algumas novidades

45697056

■ ■ ■

Vamos usar agora a biblioteca de ícones que instalamos no início. O react-icons junta várias libs de ícones e nos deixa usá-los como componentes. Escolha os ícones no endereço:

<https://react-icons.github.io/react-icons>

```
import { RemoveProps, TarefaProps } from "@types";
import { FaTrashAlt as X } from "react-icons/fa";

export default function Tarefa({titulo, setor, descricao,remove}:TarefaProps&RemoveProps){

  return(
    <div className="tarefa">
      <button onClick={()=>remove(titulo)}><X/></button>
      <h2>{titulo}</h2>
      <p>Para: {setor}</p>
      <p>Fazer: {descricao}</p>
    </div>
  )
}
```



# Projeto com algumas novidades

Agora vamos criar uma página principal e um menu para nosso projeto. Vamos começar com a página inicial, crie um componente chamado `Home.jsx`.

```
import styled from "styled-components"
```

```
const DivHome = styled.div`  
  width:100%;  
  height:85vh;  
  background: #bbf;  
  h1, p{padding:20px;}
```

```
const Home = ()=>{  
  
  return(  
    <div>  
      <h1>Página principal do projeto</h1>  
      <p>Projeto para controle de Tarefas.</p>  
    </div>  
  )  
}
```



# Exercício

45697056



Crie um projeto React chamado exercício-lava-rapido, limpe o componente page.tsx.

Faça uma aplicação para um Lava Rápido, onde você tenha uma tela e possa criar os pedidos de cada carro, contendo Marca, Modelo e Placa.

Ao terminar cada carro deve ser possível excluir a anotação do quadro.

Deve ser utilizado o React-Icons para estilização do exercício.



# DUVIDAS





Copyright © 2015 - 2021 Prof. Luís Carlos S. Silva  
Prof. Alexandre Carlos de Jesus

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).