



Faculdade

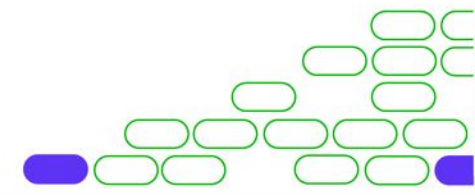


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 1. Introdução

Prof. Danilo Ferreira e Silva





Faculdade

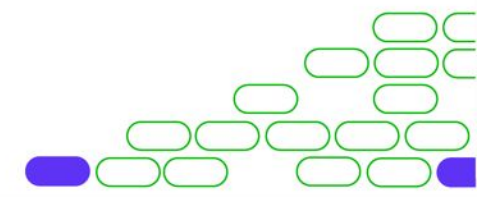


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 1.1. Visão geral do módulo**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Apresentar uma visão geral do módulo.
- ❑ Apresentar a aplicação que iremos construir ao longo das aulas.



# React II

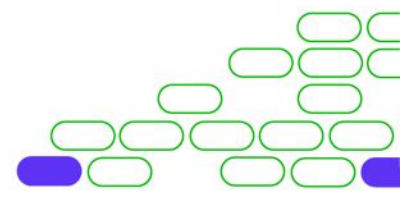
- ☐ TypeScript.
- ☐ Biblioteca de componentes (Material UI).
- ☐ Roteamento (react-router).
- ☐ Hooks avançados (useRef, useMemo, useCallback, ...).
- ☐ Context e gerenciamento de estado.
- ☐ Autenticação.
- ☐ Redux.



# Ferramentas necessárias

- ☐ Node.js.
- ☐ VSCode.
  - ☐ Extensão Prettier (opcional).
- ☐ Navegador Google Chrome.
  - ☐ Extensão React Developer Tools (opcional)





# Aplicação

Agenda React

NOVO EVENTO

Agendas

☒ Pessoal

☒ Trabalho

< > Maio de 2021

DOM	SEG	TER	QUA	QUI	SEX	SÁB
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

14:00 Reunião com o chefe

08:30 Consulta ortopedista

Entrega do projeto

09:00 Natação

08:00 Entrega do relatório

10:00 Reunião com o chefe

12:00 Almoço

14:00 Reunião com o chefe

16:00 Almoço

18:00 Reunião com o chefe

20:00 Almoço

22:00 Reunião com o chefe

24:00 Almoço

26:00 Reunião com o chefe

28:00 Almoço

30:00 Reunião com o chefe

32:00 Almoço

34:00 Reunião com o chefe

36:00 Almoço

38:00 Reunião com o chefe

40:00 Almoço

42:00 Reunião com o chefe

44:00 Almoço

46:00 Reunião com o chefe

48:00 Almoço

50:00 Reunião com o chefe

52:00 Almoço

54:00 Reunião com o chefe

56:00 Almoço

58:00 Reunião com o chefe

60:00 Almoço

62:00 Reunião com o chefe

64:00 Almoço

66:00 Reunião com o chefe

68:00 Almoço

70:00 Reunião com o chefe

72:00 Almoço

74:00 Reunião com o chefe

76:00 Almoço

78:00 Reunião com o chefe

80:00 Almoço

82:00 Reunião com o chefe

84:00 Almoço

86:00 Reunião com o chefe

88:00 Almoço

90:00 Reunião com o chefe

92:00 Almoço

94:00 Reunião com o chefe

96:00 Almoço

98:00 Reunião com o chefe

100:00 Almoço

# Back End

- ☐ Aplicação Node.js disponibilizada junto com o código fonte.
- ☐ Antes de executar:
  - ☐ `npm install`
- ☐ Executar (**sem autenticação**)
  - ☐ `npm start -- noauth`
- ☐ Executar (**com autenticação**)
  - ☐ `npm start`



# Próxima aula

- ❑ TypeScript.



**XP**e







Faculdade

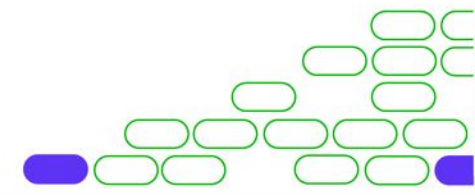


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 2. TypeScript

Prof. Danilo Ferreira e Silva





Faculdade

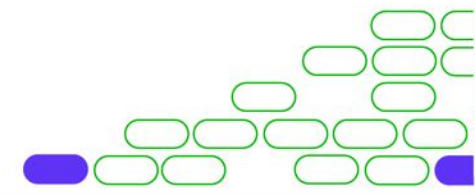


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 2.1. Introdução a TypeScript

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Introduzir a linguagem TypeScript.





# TypeScript

TypeScript é uma extensão da linguagem JavaScript que adiciona tipagem estática à linguagem.

<https://www.typescriptlang.org/>



# Vantagens

1

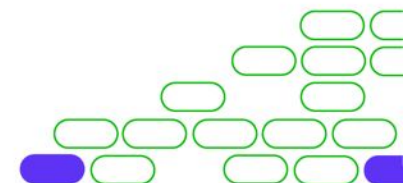
Verificação de erros em tempo de compilação.

2

Maior facilidade para manter o código.

3

Melhor suporte ferramental (autocomplete).



# Próxima aula

- ❑ Criar uma aplicação React em TypeScript.





Faculdade

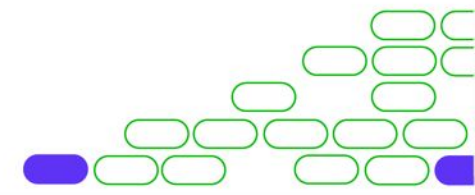


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 2.2. Aplicação React em TypeScript

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Aprender como desenvolver uma aplicação React usando Typescript.





# Próxima aula

- ❑ Definir tipos para descrever os dados manipulados pela aplicação.





Faculdade

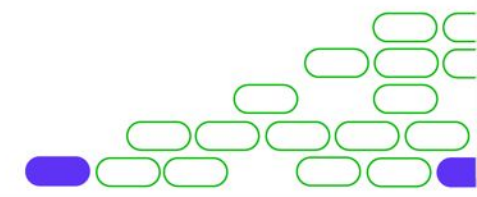


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 2.3. Definição de tipos para os dados da aplicação**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Definir tipos para descrever os dados manipulados pela aplicação.



# Conclusão

- ✓ Criamos o projeto react com a opção --template typescript.
- ✓ Definimos tipos para os objetos manipulados pela aplicação.
- ✓ Vimos como o compilador nos ajuda a identificar erros.





Faculdade

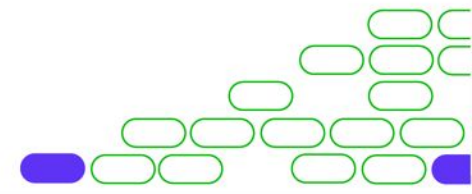


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 3. Interface com Material UI

Prof. Danilo Ferreira e Silva





Faculdade

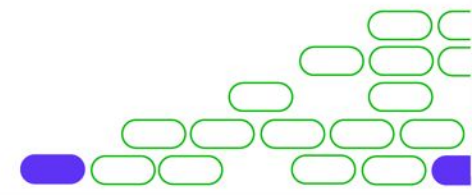


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 3.1. Material-UI**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ☐ Introduzir a biblioteca Material UI.
- ☐ Instalar a biblioteca e usar um primeiro componente.



# Material UI



MATERIAL-UI

React components for faster and easier web development.  
Build your own design system, or start with Material Design.

Biblioteca de componentes visuais baseada  
no Material Design.

<https://material-ui.com/>



# Próxima aula

- ❑ Começar a construir a interface da nossa aplicação.





Faculdade

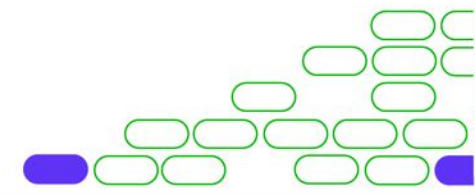


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

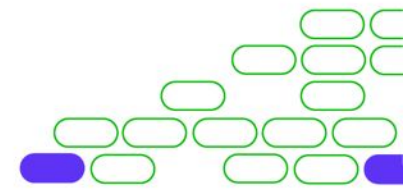
Aula 3.2. Construindo a interface (parte 1)

Prof. Danilo Ferreira E Silva



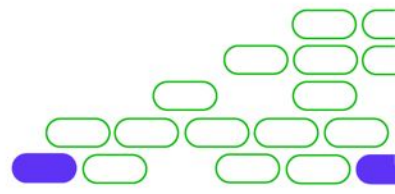
# Nesta aula

- ❑ Começar a construir a interface da aplicação.



# Próxima aula

- ❑ Continuar construindo a interface.





Faculdade

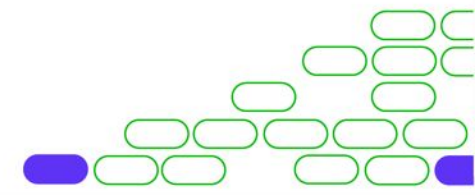


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

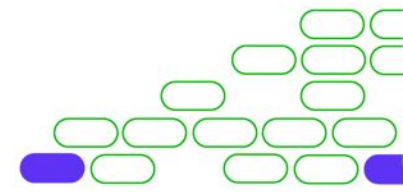
Aula 3.3. Construindo a interface (parte 2)

Prof. Danilo Ferreira E Silva



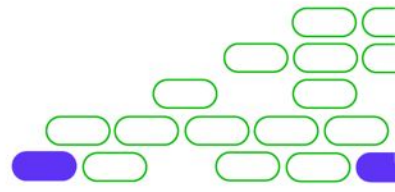
# Nesta aula

- ❑ Continuar construindo a interface.



# Próxima aula

- ❑ Continuar construindo a interface.





Faculdade

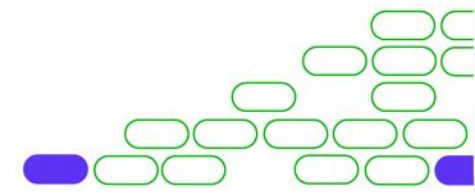


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 3.4. Construindo a interface (parte 3)

Prof. Danilo Ferreira E Silva





# Nesta aula

- ❑ Continuar construindo a interface.



# Conclusão

- ✓ Introduzimos Material UI.
- ✓ Instalamos e configuramos o mesmo.
- ✓ Construimos a interface vários componentes.
- ✓ Estilizamos componentes com o mecanismo de geração de CSS (makeStyles).





Faculdade

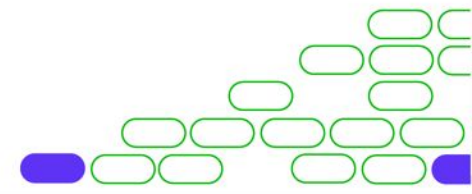


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 4. Adicionando estado e comportamento

Prof. Danilo Ferreira e Silva





Faculdade

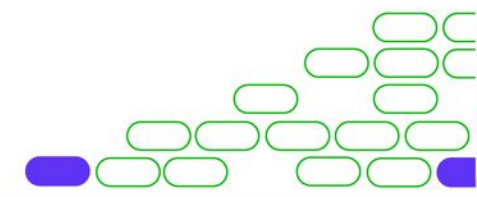


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 4.1. Gerando o calendário dinamicamente**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Introduzir a lógica para gerar o calendário.

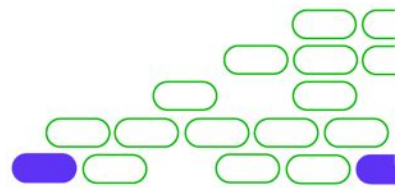


# Próxima aula

- ❑ Carregar dados do Back End.



**XP**e





Faculdade

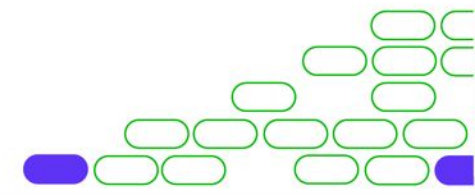


# React II

**Módulo 4. Bootcamp Desenvolvedor(a) React**

**Aula 4.2. Carregando eventos do Back End**

**Prof. Danilo Ferreira E Silva**

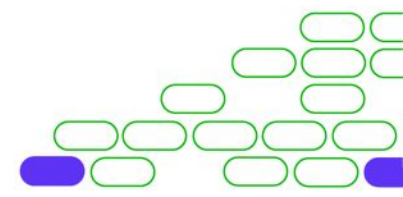


# Nesta aula

- ❑ Carregar eventos do Back End.



**XP**e



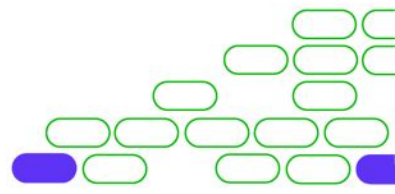


# Próxima aula

- ❑ Carregar agenda do Back End.



**XP**e





Faculdade

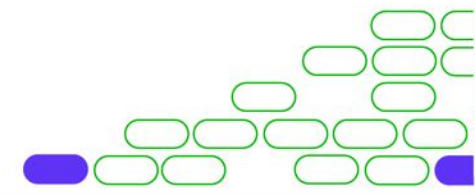


# React II

Módulo 4. Bootcamp Desenvolvedor(a) React

Aula 4.3. Carregando agendas do Back End

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Carregar agendas do Back End.

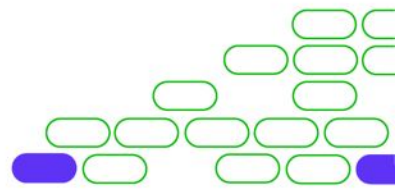


# Próxima aula

- ❑ Implementar ocultar/exibir agendas.



**XP**e





Faculdade

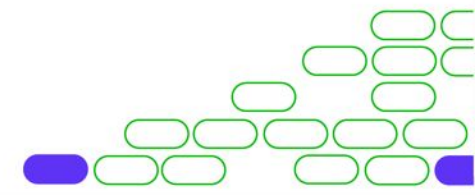


# React II

Módulo 4. Bootcamp Desenvolvedor(a) React

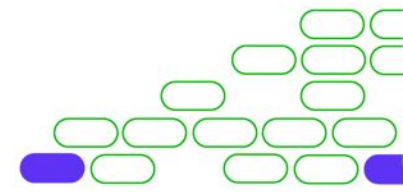
Aula 4.4. Ocultar e exibir agendas

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Implementar ocultar/exibir agendas.





# Conclusão

- ✓ Introduzimos comunicação com o Back End e estado na nossa aplicação.





Faculdade

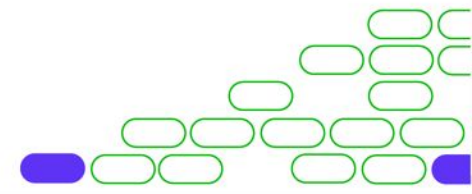


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 5. Roteamento com react-router

Prof. Danilo Ferreira e Silva







Faculdade

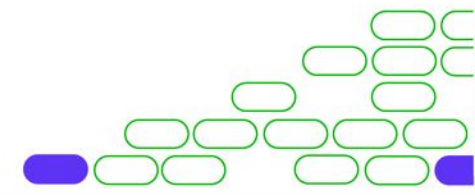


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 5.1. Introdução ao react-router

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Entender *client-side routing*.
- ☐ Introduzir o react-router.



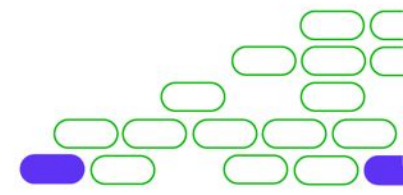
# Roteamento

## Aplicações tradicionais:

- ☐ rotas (URLs) são resolvidas pelo servidor (Back End);
- ☐ botão voltar do navegador funciona normalmente.

## Single Page Application:


- ☐ se não mudar a URL, botão voltar do navegador não funciona;
- ☐ rotas devem ser resolvidas no Front End (cliente-side);
- ☐ uso da HTML5 History API.



# Biblioteca react-router

REACT TRAINING / REACT ROUTER

GITHUB
NPM
GET TRAINING



LEARN ONCE, ROUTE ANYWHERE

## REACT ROUTER

Components are the heart of React's powerful, declarative programming model. React Router is a collection of **navigational components** that compose declaratively with your application. Whether you want to have **bookmarkable URLs** for your web app or a composable way to navigate in **React Native**, React Router works wherever React is rendering--so take your pick!

WEB
NATIVE

MacVim
File
Edit
Tools
Syntax
Netrw
Buffers
Themes
Plugin
Vimwiki
Window
Help
96% [4]
Wed Feb 1 2:41

React Router Introduction
App.js (~/.Desktop/gists/src) - VIM

```

13  .then(gists => {
14    this.setState({ gists })
15  })
16  }
17
18  render() {
19    const { gists } = this.state
20    return (
21      <Router>
22        <Root>
23          <Sidebar>
24            {gists ? (
25              gists.map(gist => (
26                <SidebarItem key={gist.id}>

```

Gists!
Assistir ma...
Compatibil...
localhost:3000/g/3a90a4c294e6a67018738d2...

Adult.m3u
[no description]
[no description]
Rimworld output log p...
[no description]
[no description]
[no description]

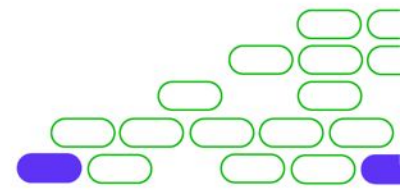
Welcome
3a90a4c294e6a67018738d2e851c42dc





# Próxima aula

- ❑ Adicionar roteamento na aplicação.





Faculdade

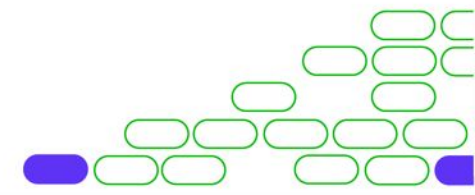


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 5.2. Adicionando roteamento (parte 1)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Adicionar react-router no projeto.
- ☐ Implementar a navegação na agenda com roteamento.

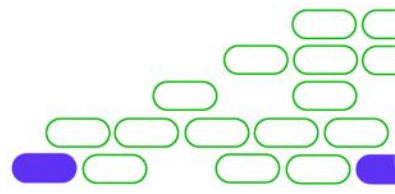


# Próxima aula

- ❑ Concluir o roteamento.



**XP**e







Faculdade

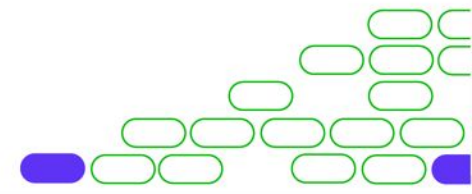


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 5.3. Adicionando roteamento (parte 2)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Concluir a implementação do roteamento.



# Conclusão

- ✓ Englobar a aplicação com componente **Router**.
- ✓ Usar componentes **Switch** e **Route** para exibir conteúdo de acordo com a rota.
- ✓ Usar componente **Link** para navegação.
- ✓ Usar hook **useParams** para obter parâmetros da rota.
- ✓ Também podemos usar o hook **useHistory** para obter informações sobre a rota ou manipular o histórico programaticamente.





Faculdade

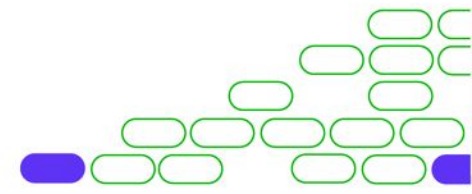


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 6. Modularizando a aplicação

Prof. Danilo Ferreira e Silva





Faculdade

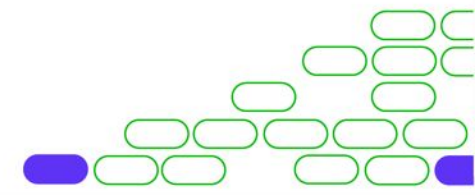


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 6.1. Modularizando a aplicação (parte 1)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Melhorar a modularização da aplicação, ou seja, “quebrar” nossa aplicação em componentes menores.



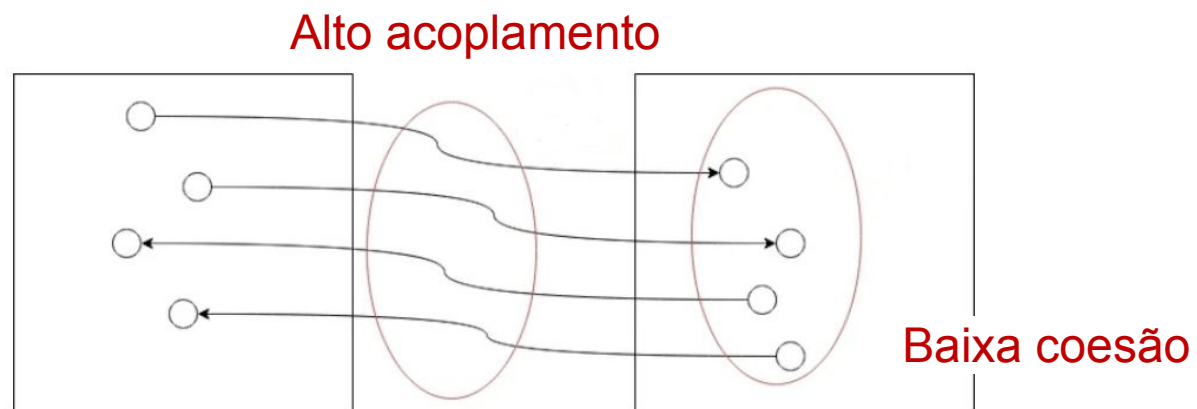
# Estratégias básicas de subdivisão

- ☐ Se o estado é usado apenas no componente, mantenha-o nele.
- ☐ Se o estado é usado em vários componentes, “suba” para um componente “pai” em comum, e passe para os “filhos” via props.
- ☐ Além de dados, podemos passar *call-backs* via props.



# Acoplamento e coesão

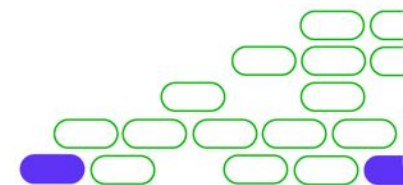
**Ruim:**



**Bom:**



Fonte: <https://shopify.engineering/shopify-monolith>



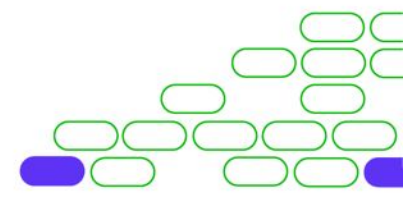


# Próxima aula

- ❑ Continuar a modularização.



**XP**e





Faculdade

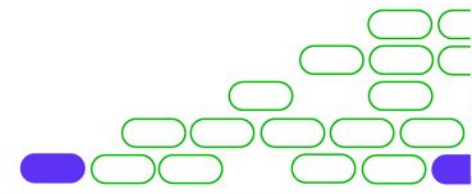


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 6.2. Modularizando a aplicação (parte 2)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Continuar a modularização.



# Conclusão

- ✓ Componentes menores são mais fáceis de entender e manter.
- ✓ Estratégia básica: passar props do “pai” para o “filho”.
- ✓ Saber onde dividir é a grande questão.





Faculdade

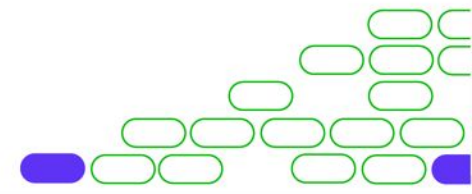


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 7. Acesso direto ao DOM com useRef

Prof. Danilo Ferreira e Silva





Faculdade

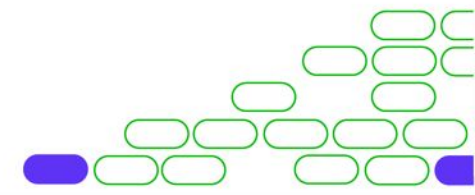


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 7.1. Hook useRef

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Introduzir o hook **useRef**.
- ☐ Entender como ele pode ser usado para acessar diretamente o DOM.



# useRef

```
const minhaRef = useRef(valorInicial);  
console.log(minhaRef.current); // imprime o valor
```

- ❑ Retorna um objeto que armazena um valor. Este objeto persiste durante toda a vida do componente.
- ❑ Mas o componente **não é renderizado** ao mudar o valor!

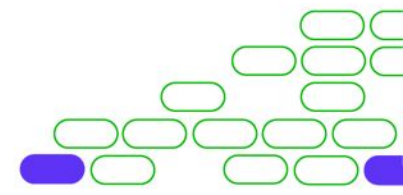




# Para que useRef?

- ❑ Por exemplo, guardar referências para o DOM.

```
function MeuComponente() {  
  const inputEl = useRef(null);  
  
  const onClick = () => inputEl.current.focus();  
  
  return <>  
    <input ref={inputEl} type="text" />  
    <button onClick={onClick}>Dá foco!</button>  
  </>;  
}
```



# Próxima aula

- ❑ Implementar o formulário de criar eventos.





Faculdade

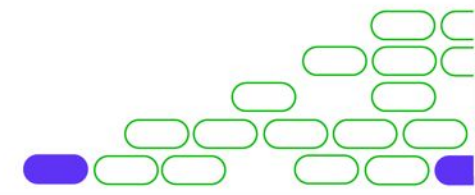


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 7.2. Implementando o formulário de criar evento (parte 1)

Prof. Danilo Ferreira E Silva



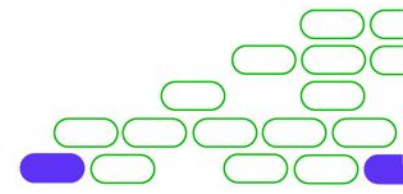
# Nesta aula

- ❑ Implementar o Dialog e formulário de criar eventos.



# Próxima aula

- ❑ Continuar a implementação do formulário.





Faculdade

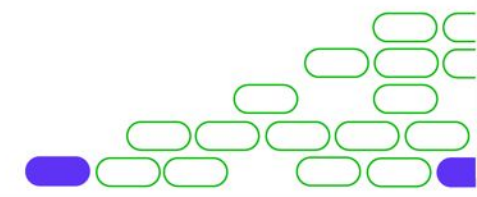


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 7.3. Implementando o formulário de criar evento (parte 2)

Prof. Danilo Ferreira E Silva



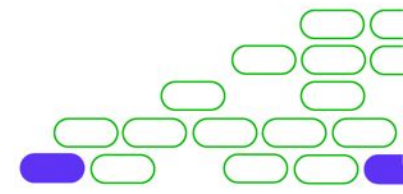
# Nesta aula

- ❑ Continuar a implementação do formulário de criar eventos.



# Próxima aula

- ❑ Continuar a implementação do formulário.







Faculdade

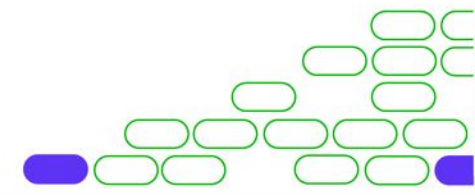


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 7.4. Implementando o formulário de criar evento (parte 3)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Continuar a implementação do formulário de criar eventos.



# Próxima aula

- ❑ Finalizar a implementação salvando dados no Back End.





Faculdade

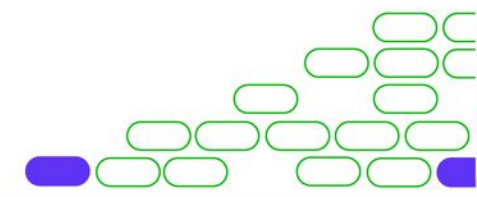


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 7.5. Salvar dados no Back End**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Finalizar a implementação salvando dados no Back End.

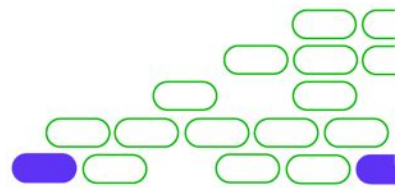


# Próxima aula

- ☐ Validar formulário.



**XP**e





Faculdade

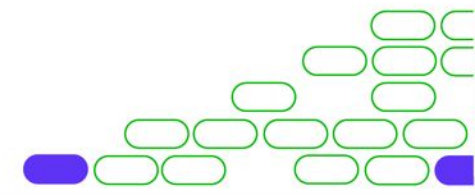


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 7.6. Validar formulário

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Validar o formulário.
- ☐ Dar foco no campo com erro.



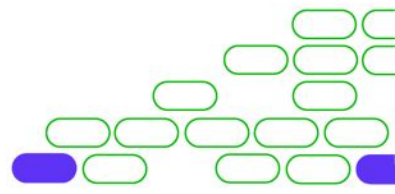


# Próxima aula

- ❑ Editar eventos.



**XP**e





Faculdade

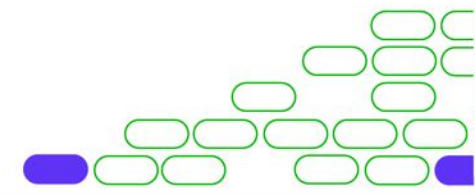


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 7.7. Editar eventos**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Concluir a implementação do formulário e dar suporte a editar eventos.



# Conclusão

- ✓ Normalmente o código React é declarativo, mas quando precisamos acessar o DOM diretamente, podemos usar **useRef** / **ref**.





Faculdade

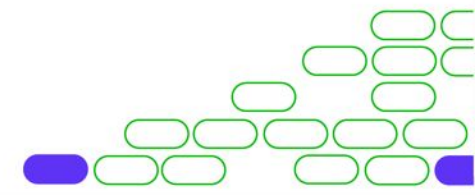


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 8. Autenticação

Prof. Danilo Ferreira e Silva





Faculdade

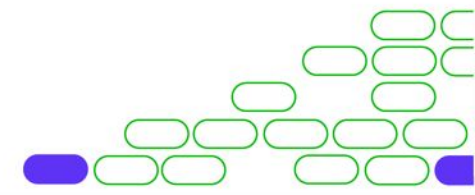


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 8.1. Lidando com autenticação**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Preparar a aplicação para lidar com a autenticação.



# Fluxo de autenticação

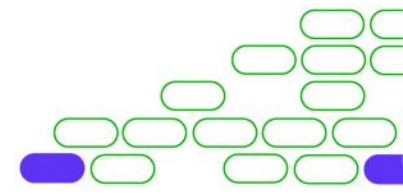
- ❑ Depende do Back End, mas tipicamente:
  - ❑ Usuário autentica via login e senha e estabelece uma sessão via cookie, ou obtém um código de acesso, com uma determinada expiração.
  - ❑ As demais requisições enviam o cookie ou código de acesso.





# Back End da nossa aplicação

<b>GET /auth/user</b>	Obtém informações do usuário logado se houver sessão, caso contrário retorna erro 401.
<b>POST /auth/login</b>	Verifica e-mail/senha e cria uma sessão no Back End. O id da sessão é armazenado em um cookie HttpOnly.
<b>POST /auth/logout</b>	Destrói a sessão no Back End.

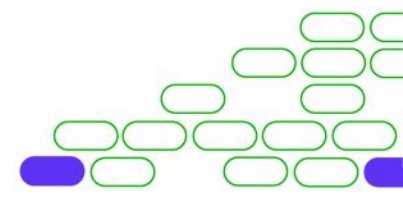


# Próxima aula

- ❑ Criar a tela de login.



**XP**e





Faculdade

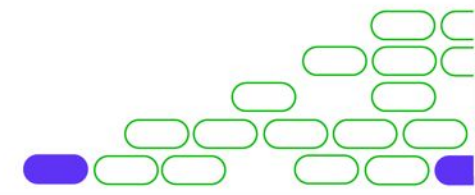


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

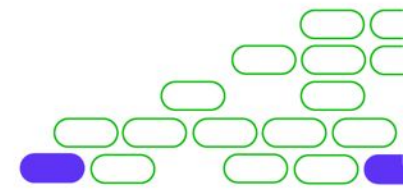
Aula 8.2. Criando a tela de login

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Criar a tela de login.

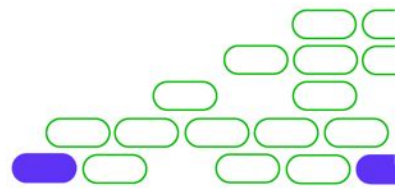


# Próxima aula

- ☐ Criar sessão no Back End.



**XP**e





Faculdade

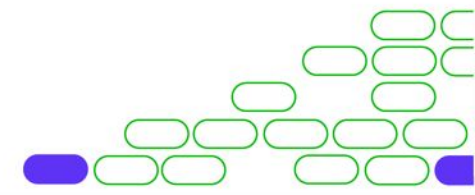


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

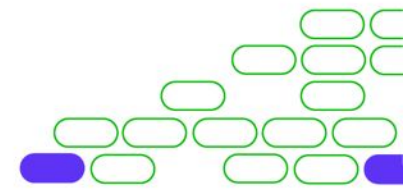
**Aula 8.3. Criando sessão no Back End**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ☐ Criar sessão no Back End.

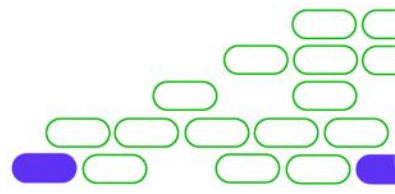


# Próxima aula

- ❑ Implementar *logout*.



**XP**e







Faculdade

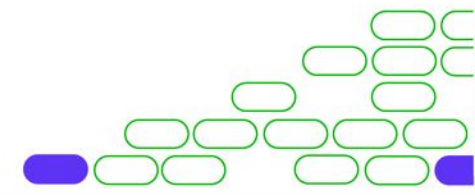


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 8.4. Adicionando logout

Prof. Danilo Ferreira E Silva

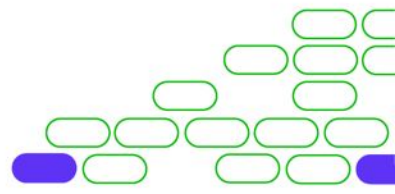


# Nesta aula

- ❑ Implementar *logout*.



**XP**e



# Conclusão

- ✓ A forma de autenticar depende do Back End.
- ✓ Caso utilize cookies de sessão, não esqueça de incluir a configuração *credentials* ao fazer fetch.
- ✓ Trate as respostas das requisições adequadamente (por exemplo, status 401).





Faculdade

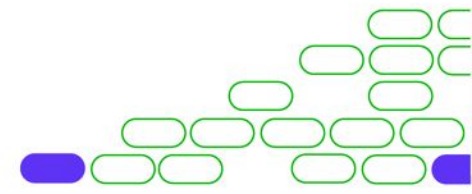


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 9. Context

Prof. Danilo Ferreira e Silva





Faculdade

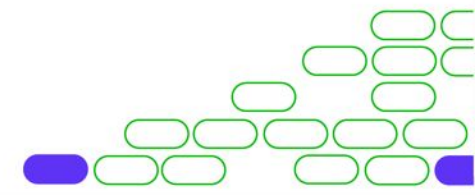


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 9.1. A API Context (parte 1)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Compartilhar informações entre componentes via API Context.



# A API Context

- ❑ API que permite compartilhar dados entre componentes da aplicação sem precisar repassá-los via props explicitamente.

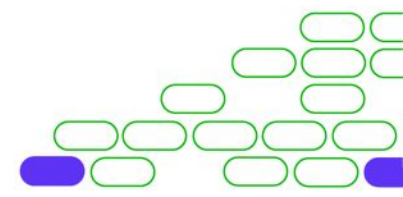


# Próxima aula

- ❑ Continuar a implementação.



**XP**e







Faculdade

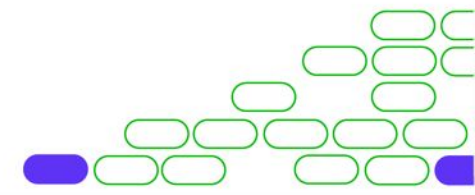


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

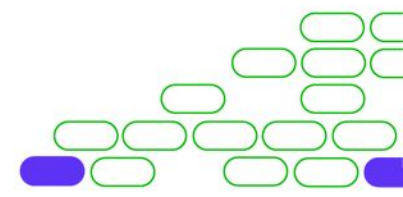
Aula 9.2. A API Context (parte 2)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Continuar a implementação.



# Conclusão

- ✓ Considere usar Context quando uma informação precisar ser repassada por vários níveis de hierarquia da árvore de componentes.
- ✓ Crie um contexto com **createContext**, forneça um valor com seu **Provider**, e obtenha o valor com **useContext**.





Faculdade

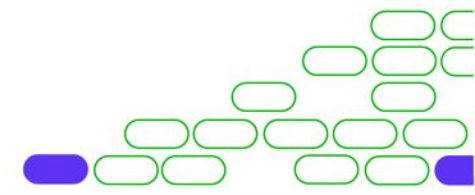


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 10. Otimização de aplicações

Prof. Danilo Ferreira e Silva





Faculdade

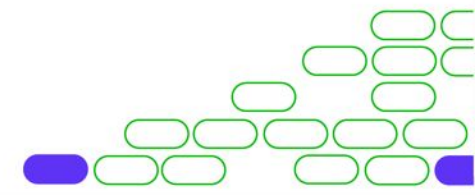


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 10.1. useMemo, useCallback e React.memo (parte 1)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Entender como evitar executar código repetidas vezes sem necessidade em uma aplicação, usando `useMemo`, `useCallback` e `React.memo`.



# Renderização de componentes

- ❑ A renderização consiste em gerar o virtual DOM, compará-lo com o DOM, e aplicar as mudanças necessárias.
- ❑ Por padrão, toda vez que ocorre qualquer mudança de estado em um componente ele é renderizado novamente, **assim como toda a subárvore abaixo dele.**



# Renderização de componentes

- ❑ Normalmente isso não é um problema, pois a renderização é rápida, mas podemos ter problemas de performance com componentes muito grandes.





# O que podemos fazer?

- ❑ **useMemo**: Hook que computa um valor apenas quando houver modificações em suas dependências.
- ❑ **React.memo**: Função que, dado um componente, retorna uma versão otimizada do mesmo que só renderiza quando algum prop recebido for alterado.

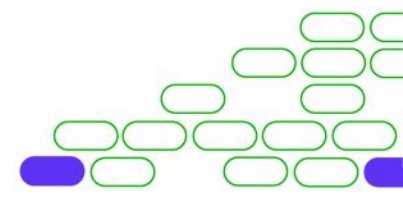


# Próxima aula

- ❑ Continuar a otimização.



**XP**e





Faculdade

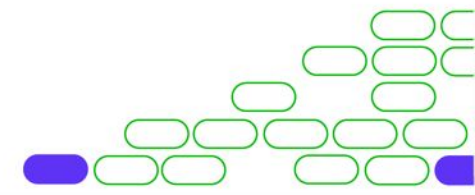


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

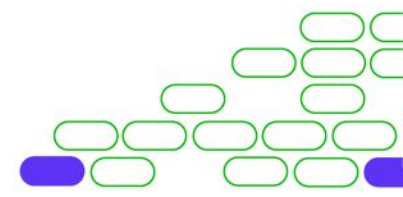
Aula 10.2. useMemo, useCallback e React.memo (parte 2)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Continuar a otimização.



# Conclusão

- ✓ Execute código “pesado” apenas quando necessário, por meio da **useMemo**.
- ✓ Use **React.memo** para evitar renderizações desnecessárias.
- ✓ Cuide para que as props de componentes com **React.memo** sejam estáveis, usando **useMemo** e **useCallback** quando necessário.





Faculdade

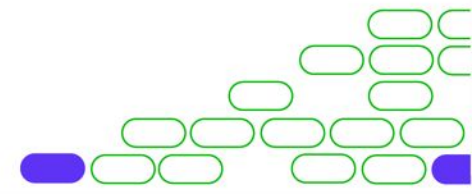


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 11. useReducer

Prof. Danilo Ferreira e Silva





Faculdade

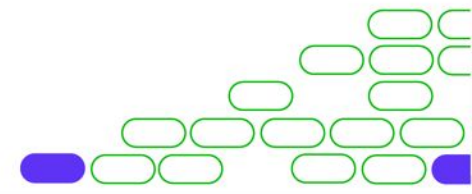


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 11.1. Usando useReducer (parte 1)**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Introduzir o hook **useReducer**.
- ❑ Utilizá-lo para gerenciar o estado do componente CalendarScreen.



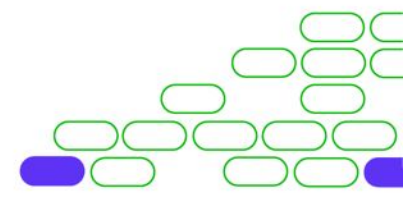


# Próxima aula

- ❑ Continuar a implementação.



**XP**e





Faculdade

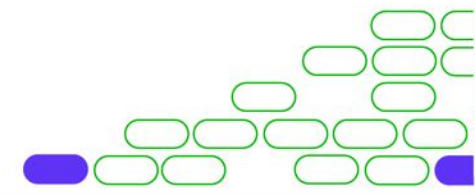


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 11.2. Usando useReducer (parte 2)**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Continuar a implementação.



# Próxima aula

- ❑ Continuar a implementação.



**XP**e





Faculdade

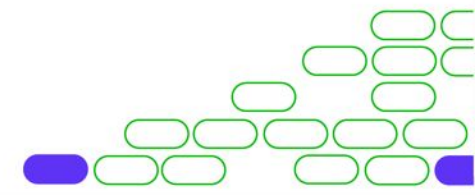


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 11.3. Usando useReducer (parte 3)

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Concluir a implementação.



# Conclusão

- ✓ **useReducer** é uma alternativa ao **useState** quando temos transições de estado mais complexas e desejamos maior controle.





Faculdade

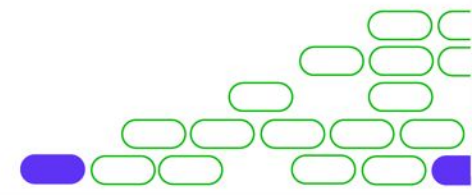


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 12. Hooks customizados

Prof. Danilo Ferreira e Silva







Faculdade

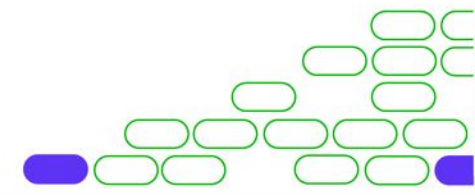


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 12.1. Como criar nossos hooks

Prof. Danilo Ferreira E Silva



# Nesta aula

- ☐ Aprender a criar nossos próprios hooks.



# Conclusão

- ✓ Hooks customizados permitem reutilizar código e modularizar a aplicação.
- ✓ Basta criarmos um função **useNomeHook**.
- ✓ Devemos seguir as mesmas regras de hooks usadas em componentes.





Faculdade

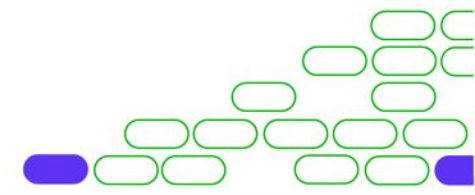


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 13. Class components

Prof. Danilo Ferreira e Silva





Faculdade

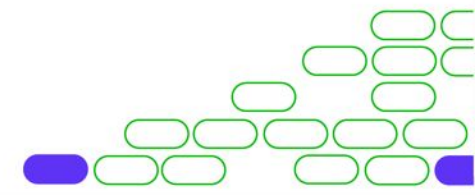


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 13.1. Class components

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Entender como criar Class Components e para que eles servem.



# Class Components

- ❑ Antes do React 16.8, a única forma de definir componentes contendo estado era por meio de classes.
- ❑ O estado do componente é armazenado na propriedade **state**, e atualizado via **setState**.
- ❑ A exibição do componente é implementada no método **render**.



# Class Components

```
class Counter extends React.Component<{}, {counter: number}> {  
  constructor(props: {}) {  
    super(props);  
    this.state = {counter: 0};  
  }  
  
  increment() {  
    this.setState({ counter: this.state.counter + 1 })  
  }  
  
  render() {  
    return <div>  
      <div>{this.state.counter}</div>  
      <button onClick={() => this.increment()}>Increment</button>  
    </div>;  
  }  
}
```





# Ciclo de vida do componente

- ❑ Apenas Class components podiam interagir com o ciclo de vida de componentes, sobrescrevendo os métodos:
  - ❑ `componentDidMount()`
  - ❑ `componentWillUnmount()`
  - ❑ `componentDidUpdate(prevProps, prevState)`



# Próxima aula

- ❑ Converter um dos componentes da aplicação para Class Component.





Faculdade

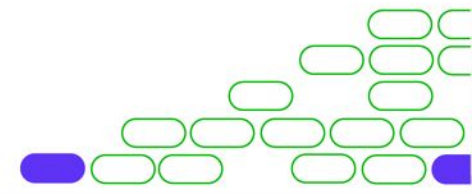


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 13.2. Convertendo um componente para Class Component

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Converter um dos componentes da aplicação para Class Component.



# Conclusão

- ✓ Não precisamos usar Class Components em componentes novos, mas é importante conhecê-los para entender o código legado.



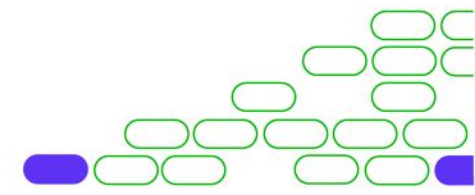


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Capítulo 14. Redux

Prof. Danilo Ferreira e Silva





Faculdade

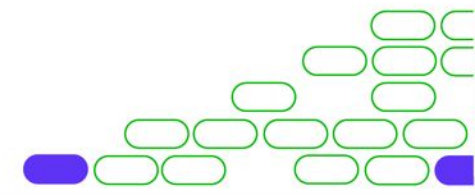


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 14.1. Introdução ao Redux

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Conhecer a biblioteca Redux, que nos ajuda a gerenciar o estado de nossa aplicação:

<https://redux.js.org/>.





# A arquitetura redux

- ❑ O estado global da aplicação é centralizado no **store**. Existe uma *única fonte da verdade*.
- ❑ O estado é imutável. A única forma de alterá-lo é emitindo uma **action**, um objeto que representa uma ação.
- ❑ Dado o estado atual e uma **action**, Um novo estado é calculado pela função **reducer**. Tal função não possui nenhum tipo de efeito colateral.

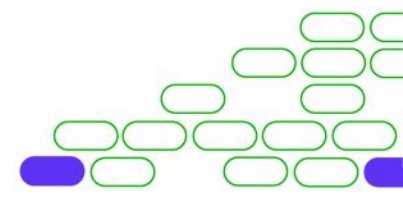


# Próxima aula

- ❑ Começar a implementar a aplicação.



**XP**e





Faculdade

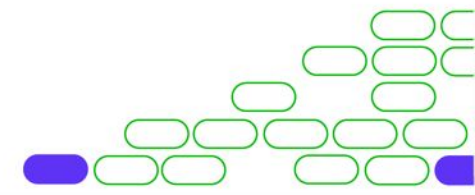


# React II

**Módulo 3. Bootcamp Desenvolvedor(a) React**

**Aula 14.2. Implementar a aplicação usando Redux**

**Prof. Danilo Ferreira E Silva**



# Nesta aula

- ❑ Começar a implementar a aplicação usando Redux.



# Próxima aula

- ❑ Utilizar o reducer e store em um componente React.





Faculdade

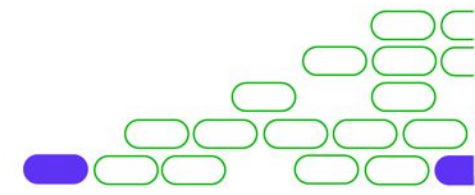


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 14.3. Integrar Redux com React

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Utilizar o reducer e store em um componente React.



# Próxima aula

- ❑ Utilizar o `@redux/toolkit` para criar o reducer.







Faculdade

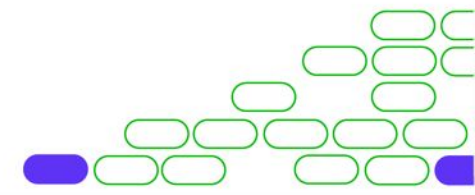


# React II

Módulo 3. Bootcamp Desenvolvedor(a) React

Aula 14.4. Reducer com o Redux Toolkit

Prof. Danilo Ferreira E Silva



# Nesta aula

- ❑ Utilizar o `@redux/toolkit` para criar o reducer.



# Conclusão

- ✓ Construimos uma aplicação usando Redux.
- ✓ Apesar de uma maior complexidade inicial, o Redux oferece algumas vantagens, principalmente quando queremos separar a lógica da exibição.
- ✓ Os recursos mais modernos do React tornam o Redux menos atrativo.

