



DATAPREV



UMA BIBLIOTECA JAVASCRIPT PARA CRIAR INTERFACES DE USUÁRIO

Ricardo Glodzinski
Analista de Tecnologia da Informação

Tiago de Andrade Freire
Analista de Tecnologia da Informação





AULA 3

PLANEJAMENTO

- Rotas com React Router
- Uso de CSS nos componentes React
- Usando SASS como pré-compilador de CSS
- Integração com outras bibliotecas
 - Bootstrap
 - Material-UI
 - Starscream-Theme
 - Dtp-React-Components
 - DSGov
- Integração com Web Components
- Atividades assíncronas: exercícios

- O React Router é uma coleção de componentes de navegação.
- O pacote é disponibilizado pelo NPM com o nome react-router-dom.
- Comando de instalação:

```
cd ../sua-app  
npm install react-router-dom --save
```



ROTAS

REACT ROUTER – ROTEAMENTO BÁSICO

```
import React from "react";
import { BrowserRouter as Router } from "react-router-dom";

import Menu from "../Layout/Menu";
import Routes from "../routes";

const App = () => {
  return (
    <Router>
      <React.Fragment>
        <Menu />
        <Routes />
      </React.Fragment>
    </Router>
  );
};

export default App;
```

```
import React from 'react';
import { Link } from "react-router-dom";

const Menu = () => {
  return (
    <nav>
      <ul>
        <li>
          <Link to="/">Painel</Link>
        </li>
        <li>
          <Link to="/sobre">Sobre</Link>
        </li>
        <li>
          <Link to="/usuarios">Usuários</Link>
        </li>
      </ul>
    </nav>
  );
};

export default Menu;
```



Hands on!

ROTAS

REACT ROUTER – ROTEAMENTO BÁSICO

```
import React from 'react';
import { Switch, Route } from 'react-router-dom';

import Sobre from "../Pages/Sobre";
import Painel from "../Pages/Painel";
import Usuarios from "../Pages/Usuarios";

const Routes = () => {
  return (
    <Switch>
      <Route path="/sobre" component={Sobre} />
      <Route path="/usuarios" component={Usuarios} />
      <Route path="/" component={Painel} />
    </Switch>
  );
};

export default Routes;
```



Hands on!

ROTAS

REACT ROUTER – ROTEAMENTO ANINHADO

```
import React from "react";
import { Link, useRouteMatch, Route, Switch } from "react-router-dom";
import Topico from "../Topico";

const Topics = () => {
  let match = useRouteMatch();

  return (
    <div>
      <h2>Tópicos</h2>
      <ul>
        <li>
          <Link to={`/${match.url}/componentes`} >Componentes</Link>
        </li>
        <li>
          <Link to={`/${match.url}/props-vs-state`} >Props vs. State</Link>
        </li>
      </ul>
      <Switch>
        <Route path={`/${match.path}/:topicoId`} >
          <Topico />
        </Route>
        <Route path={match.path}>
          <h3>Selecione um tópico.</h3>
        </Route>
      </Switch>
    </div>
  );
};

export default Topics;
```



Hands on!

ROTAS

REACT ROUTER – ROTEAMENTO ANINHADO

```
import React from "react";
import { useParams } from "react-router-dom";

const Topico = () => {
  let { topicId } = useParams();
  return <h3>Tópico selecionado: {topicId}</h3>;
};

export default Topico;
```

- Com os exemplos apresentados é possível iniciar a criação de rotas em uma aplicação React.
- Outros exemplos podem ser verificados na documentação oficial do React Router: <https://reactrouter.com/web/example/>



Hands on!

ESTILIZAÇÃO E CSS

FOLHAS DE ESTILOS

- É possível importar e utilizar folhas de estilos prontas.
- Para aplicar o CSS importado, passar o nome da classe CSS no atributo className do elemento.



Hands on!

```
import React from "react";

import "./estilos.css";

const UsoDeCss = () => {
  return (
    <div className="wrapper">
      <header>
        <div className="section">Cabeçaho</div>
      </header>
      <div className="content">
        <div className="section">Conteúdo</div>
      </div>
      <footer>
        <div className="section">Rodapé</div>
      </footer>
    </div>
  );
};

export default UsoDeCss;
```

ESTILIZAÇÃO E CSS

FOLHAS DE ESTILOS

- É comum para classes do CSS dependerem de props ou o state do componente.
- Pacote [classnames](#) pode auxiliar no tratamento de quais classes CSS devem ser aplicadas, ou não, de acordo com as props.

```
render() {  
  let className = 'menu';  
  if (this.props.isActive) {  
    className += ' menu-active';  
  }  
  return <span className={className}>Menu</span>  
}
```

```
import classnames from "classnames";  
  
render() {  
  
  const { isActive = true } = this.props;  
  
  return <span className={classnames("menu",  
    { "active": isActive })}>Menu</span>  
}
```



ESTILIZAÇÃO E CSS

CSS INLINE

- É possível utilizar CSS inline, passando os valores no atributo ***style*** dos elementos.
- Não é recomendado que seja a principal forma de estilização de componentes de um sistema. Na maioria dos casos, className deve ser usado para referenciar classes definidas em um arquivo de estilo CSS externo.
- O atributo style aceita um objeto JavaScript com propriedades em camelCase ao invés de uma string CSS.

ESTILIZAÇÃO E CSS

CSS INLINE

```
const divStyle = {  
  color: 'blue',  
  backgroundImage: 'url(' + imgUrl + ')',  
};  
  
function HelloWorldComponent() {  
  return <div style={divStyle}>Hello World!</div>;  
}
```

- Perceba que estes estilos não são auto prefixados. Para serem compatíveis com navegadores antigos você precisa fornecer as propriedades de estilos correspondentes:

```
const divStyle = {  
  WebkitTransition: 'all', // perceba o 'W' maiúsculo aqui  
  msTransition: 'all' // 'ms' é o único prefixo de fornecedor minúsculo  
};  
  
function ComponentWithTransition() {  
  return <div style={divStyle}>Isto deve funcionar em diferentes navegadores</div>;  
}
```


ESTILIZAÇÃO E CSS

CSS INLINE

- Chaves de style são em camelCase com o intuito de serem consistentes com o acesso de propriedades que são nós do DOM através do JS (ex. `node.style.backgroundImage`).
- Prefixos de fornecedor (vendor prefixes) diferentes de ms devem começar com a letra maiúscula. É por isso que WebkitTransition tem um “W” maiúsculo.
- O React vai acrescentar automaticamente um sufixo “px” para determinadas propriedades numéricas de inline style.
- Para usar unidades diferentes de “px”, especifique o valor como uma string com a unidade desejada. Ex: `height: '100vh'`.
- Nem todas as propriedades de estilos são convertidas para strings pixel. Algumas permanecem sem unidade (exemplo: `zoom`, `order`, `flex`). Uma lista completa com as propriedades sem unidade pode ser vista [aqui](#).

ESTILIZAÇÃO E CSS

CSS-IN-JS

- Refere a um padrão onde o CSS é definido utilizando JavaScript no lugar de arquivos externos.
- Esta funcionalidade não faz parte do React, mas é fornecida por bibliotecas de terceiros.
- Comparação das bibliotecas de CSS-in-JS [aqui](#).
- Não iremos fazer exemplos práticos usando CSS-in-JS pois existem muitas bibliotecas, cada uma com sua diferente forma de uso.
- Utilizar `className` resolve a maiorias das questões de CSS no React.

ESTILIZAÇÃO E CSS

SASS

- O suporte a SASS está disponível por padrão, no create-react-app, que utilize react-scripts $\geq 2.0.0$.
- Para utilizá-lo, basta instalar a biblioteca como dependência do projeto. O react-scripts já irá prover o suporte e configurações.

```
cd ../sua-app  
npm install node-sass --save
```



Hands on!

ESTILIZAÇÃO E CSS

SASS

```
.sass-button {  
  &.flat {  
    border: none;  
  }  
  &.small {  
    font-size: xx-small;  
  }  
  &.large {  
    font-size: x-large;  
  }  
}
```

```
import React from "react";  
import classNames from "classnames";  
  
import "./estilos.scss";  
  
const SassButton = props => {  
  const { children, flat, small, large } = props;  
  const classes = classNames(  
    "sass-button",  
    { "flat": flat },  
    { "small": small },  
    { "large": large }  
  );  
  return <button className={classes}>{children}</button>;  
};  
  
export default SassButton;
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

BOOTSTRAP

- É o framework mais popular do mundo para a criação de sites responsivos.
- Possui pacote disponível para instalação via NPM:

```
cd ../sua-app  
npm install bootstrap --save
```

```
// Importe o CSS do Bootstrap e, opcionalmente, o CSS do tema do Bootstrap no início do seu arquivo  
// src/index.js
```

```
import 'bootstrap/dist/css/bootstrap.css';
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

BOOTSTRAP

- O Bootstrap disponibiliza os seus estilos em formato SASS. Isso permite que ele seja customizado, alterando, por exemplo, variáveis de cores, tamanhos, etc.
- A referência completa das variáveis que pode ser consultada [aqui](#).
- É necessário já ter configurado suporte a SASS na aplicação.
- Criar um arquivo onde serão realizadas as customizações. Sugestão: *src/assets/styles/bootstrap-custom.scss* e o importar em **src/App.js**.



Hands on!

```
// Sobrescrever os valores padrão das variáveis do Bootstrap
$body-bg: #000;

// Importar o SCSS do Bootstrap
@import '~bootstrap/scss/bootstrap.scss';
```

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

BOOTSTRAP

- Embora não seja necessário utilizar nenhuma biblioteca de componentes para integrar o Bootstrap ao app React, geralmente é mais fácil do que tentar agrupar os plugins Bootstrap jQuery.
- A sugestão é utilizar o CSS do Bootstrap em conjunto com bibliotecas de componente React, como o [React Bootstrap](#) e [Reactstrap](#).

```
cd ../sua-app  
npm install react-bootstrap reactstrap --save
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

BOOTSTRAP

```
import React from "react";
import { Button as ReactstrapButton } from "reactstrap";
import Button from "react-bootstrap/Button";
import Container from "react-bootstrap/Container";
import Row from "react-bootstrap/Row";
import Col from "react-bootstrap/Col"

const Bootstrap = () => {
  return (
    <Container>
      <Row>
        <Col>
          <ReactstrapButton color="danger">Reactstrap Button</ReactstrapButton>
        </Col>
        <Col>
          <Button>React Bootstrap Button</Button>
        </Col>
      </Row>
    </Container>
  );
};

export default Bootstrap;
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

MATERIAL-UI

- É um biblioteca de componentes que segue o Google Material Design.
- Possui pacote disponível para instalação via NPM:

```
cd ../sua-app  
npm install @material-ui/core --save
```

```
<!-- Carregar a fonte padrão Roboto em public/index.html -->  
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap" />
```

```
import React from "react";  
import { Button } from "@material-ui/core";  
  
const MaterialUI = () => <Button color="primary">Botão do Material-UI</Button>;  
  
export default MaterialUI;
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

STARSCREAM-THEME

- É uma biblioteca CSS própria da Dataprev, construída e mantida pelo time de Frontend da COPD. Implementa os componentes definidos no Starscream-Design.
- Possui pacote disponível para instalação via NPM/GIT:

```
cd ../sua-app  
npm install git+https://www-scm.prevnet/copd/starscream-theme.git --save
```

```
<body class="nome-do-tema"></body>
```

- Os temas disponíveis são: default-theme, inss-theme, mtb-theme, anvisa-theme, rfb-theme e dsgov-theme.
- O dsgov-theme é uma adaptação ao novo sistema de Design do Governo Federal. Não é uma implementação completa do DS Gov.



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

DTP-REACT-COMPONENTS

- O Dtp-React-Components é uma biblioteca de componentes React que implementa os componentes do Starscream.
- Deve ser utilizada em conjunto com o Starscream-Theme, que foi apresentado na seção anterior. Os ícones utilizados são da [LIBICONS](#).
- Possui pacote disponível para instalação via NPM/GIT:

```
cd ../sua-app  
npm install git+https://www-scm.prevnet/copd/dtp-react-components.git --save  
npm install git+https://www-scm.prevnet/copd/libicons.git --save
```

- Importar o CSS do tema do Starscream-Theme que deseja utilizar e a LIBICONS.



Hands on!

```
import "starscream-theme/dist/css/default.min.css";  
import "libicons/libicons/style.css";
```

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

DTP-REACT-COMPONENTS

```
import React from "react";
import { DtpHeader, DtpContent, DtpContentContainer, DtpPageHeader, DtpButton } from "dtp-react-components";
import "starscream-theme/dist/css/default.min.css";
import "libicons/libicons/style.css";

const DtpReactComponents = () => {

  const appInfo = {
    firstName: "Treinamento",
    lastName: `React`,
    owner: "Coordenação de Gestão de Padrões de Desenvolvimento - COPD"
  };

  const homeHeader = <DtpPageHeader title="Página Inicial" />;

  return (
    <div className="default-theme">
      <DtpHeader homeUrl="/" appInfo={appInfo} />
      <DtpContent>
        <DtpContentContainer pageHeader={homeHeader}>
          <DtpButton>Botão do Dtp-React-Components</DtpButton>
        </DtpContentContainer>
      </DtpContent>
    </div >
  );
};

export default DtpReactComponents;
```



Hands on!

INTEGRAÇÃO COM OUTRAS BIBLIOTECAS

DS GOV

- O DS Gov é o novo Design System do Governo Federal Brasileiro.
- É uma iniciativa do Serpro mas a Dataprev está negociando para que seja possível o desenvolvimento de um trabalho colaborativo.
- Site oficial: <http://dsgov.estaleiro.serpro.gov.br/>
- Atualmente em fase BETA (29/07/2020).
- Disponibiliza apenas CSS e JavaScript puro. Não possui bibliotecas de componentes baseadas em stacks específicas (React, Angular, Vue, etc).
- Há uma iniciativa na Dataprev/COPD para a criação de WebComponents que encapsulem os componentes do DS Gov, sendo assim possível sua utilização em stacks específicas. Atualmente em fase de POC (29/07/2020).

INTEGRAÇÃO COM WEB COMPONENTS

- Os WebComponents são um conjunto de APIs que permitem criar novas tags HTML personalizadas, reutilizáveis e encapsuladas para uso em páginas e aplicativos web.
- Abaixo o exemplo de uso do WebComponent <paper-badge>:

```
cd ../sua-app  
npm install --save @polymer/paper-badge
```

```
import React from "react";  
import "@polymer/paper-badge/paper-badge.js";  
  
const WebComponents = () => {  
  return (  
    <div>  
      <div style={{ display: "inline-block" }}>  
        <span>Caixa de Entrada</span>  
        <paper-badge label="3"></paper-badge>  
      </div>  
    </div>  
  );  
};  
  
export default WebComponents;
```



Hands on!



Obrigado!

Ricardo Glodzinski

Analista de Tecnologia da Informação

ricardo.glodzinski@dataprev.gov.br

Tiago de Andrade Freire

Analista de Tecnologia da Informação

tiago.freire@dataprev.gov.br

Agosto de 2020



www.facebook.com/dataprevtecnologia



[@dataprev](https://twitter.com/dataprev)



[DATAPREV](https://www.linkedin.com/company/dataprev)