

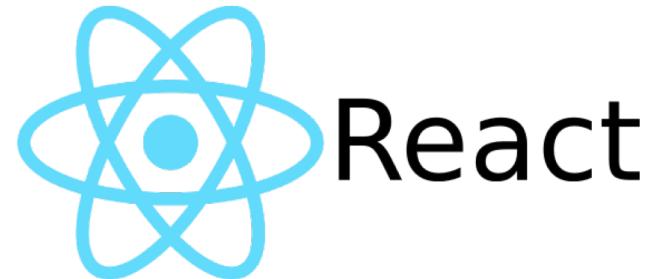
**INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE**
Campus Natal - Central

Introdução à React

Prof. Fellipe Aleixo (*fellipe.Aleixo@ifrn.edu.br*)

React.js

- Código aberto
- Framework de desenvolvimento Web baseado em **Javascript**
- Foco apenas na definição da UI
- Entrou em cena em maio 2013
- Mantido pelo **Facebook**, **Instagram** e uma comunidade de desenvolvedores e empresas



Introdução à React

1. Componentes
 - a. JSX – linguagem de descrição Javascript/HTML
 - b. Composição de componentes
 - c. Propriedades de componentes
2. Manipulação de eventos
3. Ciclo de vida dos componentes

COMPONENTES

Descrevendo Componentes com JSX

- Acrônimo para Javascript XML
- Utilizada para descrever componentes de UI
- Similar à HTML (nomes das *tags*)

Componentes React

- As aplicações são criadas através de componentes – **composição** e **modularização**

```
import React, { Component } from 'react';
```

```
class Title extends Component {
  render() {
    return (
      <h1> Hello World! </h1>
    );
  }
}
```

Composição de Componentes

- Propõem composição ao invés de herança
- Construção de componentes complexos a partir de componentes mais simples
- Composição também permite a criação de variações de componentes existentes

Composição de Componentes

```
import React, { Component } from 'react';
class NewsItem extends Component {
  render() {
    return (
      <div className="news-item">
        <Image />
        <Title />
        <Byline />
        <Description />
      </div>
    );
  }
}
```

Composição de Componentes

```
import React, { Component } from 'react';

class WarningTitle extends Component {
  render() {
    return (
      <div style={{ border: '1px solid red' }}>
        <Title />
      </div>
    );
  }
}
```

Props e PropTypes

- Torna a saída dos componentes dinâmica

```
import React, { Component, PropTypes } from 'react';
export default class Title extends Component {
  render() {
    return (
      <h1>{this.props.titleText}</h1>
    );
  }
}
Title.propTypes = {
  titleText: PropTypes.string
};
```

Props e PropTypes

- Além de propriedades do tipo *string*, podemos ter propriedades de outros tipos simples

```
Title.propTypes = {  
  titleText: PropTypes.string.isRequired,  
  highlighted: PropTypes.bool,  
  fontSize: PropTypes.number  
};
```

Props e PropTypes

```
import React, { Component, PropTypes } from 'react';
export default class Title extends Component {
  render() {
    return (
      <h1 style={{
        backgroundColor: this.props.highlighted ? 'yellow' : 'white',
        fontSize: `${this.props.fontSize}px` }}>
        {this.props.titleText}
      </h1>
    );
  }
}
```

Passando Propriedades

- Passadas quando o componente é utilizado

```
import React, { Component } from 'react';
import Title from './Title';
export default class NewsItem extends Component {
  render() {
    return (
      <div className="news-item">
        <Image />
        <Title titleText="Hello World!"
              highlighted={true}
              fontSize={18} />
        ...
    )
  }
}
```

Tipos de Propriedades

- Podem ser de tipos complexos

```
propTypes: {  
    simpleArray: PropTypes.array, //Arrays  
    simpleObject: PropTypes.object, //Objetos  
    arrayOfNumbers: PropTypes.arrayOf(PropTypes.number),  
    complexObject: PropTypes.shape({  
        id: PropTypes.number,  
        name: PropTypes.string  
    })  
}
```

Valores Padrão de Propriedades

- Valores utilizados quando não especificados

...

```
Title.propTypes = {  
  titleText: PropTypes.string.isRequired,  
  highlighted: PropTypes.bool,  
  fontSize: PropTypes.number  
};
```

```
Title.defaultProps = {  
  highlighted: false,  
  fontSize: 18  
};
```

Props.children

- Representam conteúdo passados entre as *tags*

```
render() {  
  return (  
    <h1 style={{  
      backgroundColor: this.props.highlighted?'yellow':'white',  
      fontSize: `${this.props.fontSize}px`  
    }} >  
      {this.props.children}  
    </h1>  
  );  
}
```

Manipulação de
EVENTOS

Manipulação de Eventos

- Para programar a resposta aos eventos temos:
 - *Event handlers*
 - *Event listeners*

```
document.querySelector('form').addEventListener('click', validate);
```

```
function validate () {  
    alert('The form is valid!');  
}
```

Manipulação de Eventos

- Em HTML:

```
<form onsubmit="validate ()">  
  ...  
</form>
```

- Em JSX:

```
<form onSubmit={validate}>
```

Manipulação de Eventos

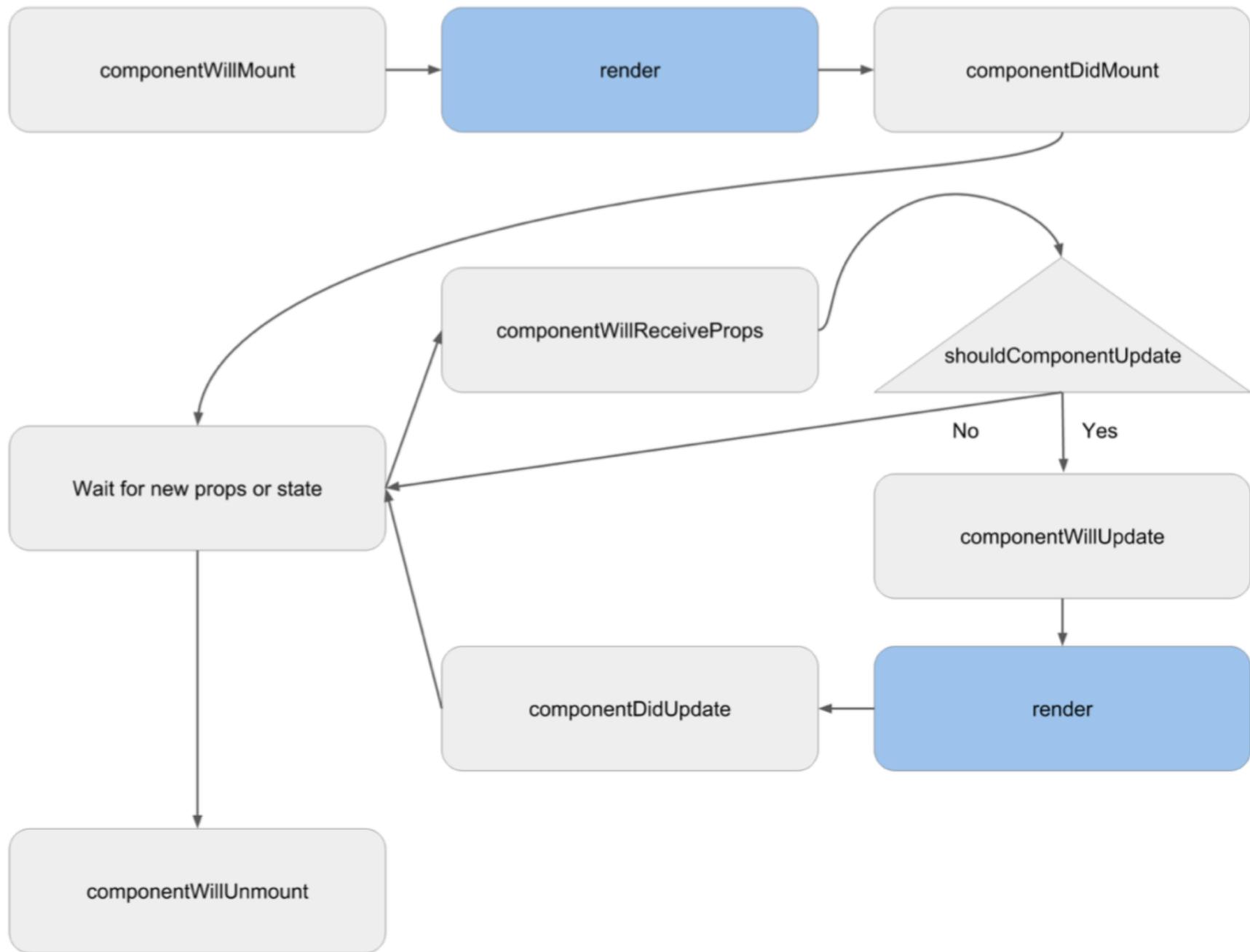
```
import React, { Component, PropTypes } from 'react';
import Title from './Title';

export default class NewsItem extends Component {
  onClick() {
    alert('You've clicked on ${this.props.titleText}');
  }
  render() {
    return (
      <div className="news-item"
        onClick={this.onClick.bind(this)}>
      ...
    
```

```
...
    <Image />
    <Title highlighted>
        {this.props.titleText}
    </Title>
    <Byline />
    <Description />
</div>
);
}
}

NewsItem.propTypes = {
  titleText: PropTypes.string.isRequired
};
```

CICLO DE VIDA



Mais Informações

- <https://reactjs.org>
- <https://redux.js.org/basics/usage-with-react>
- <https://medium.com/reactbrasil>