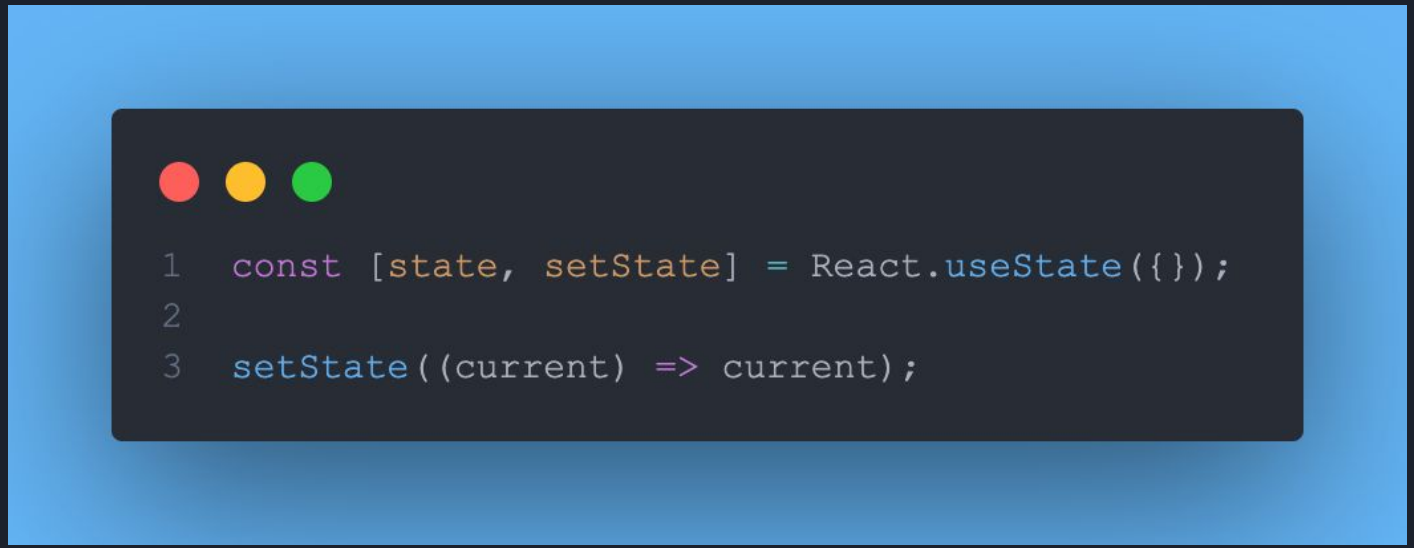
React Hooks

Prof. Guilherme Alves

O que são hooks?

Os hooks são métodos que nos permitem definir estado e realizar o controle do ciclo de vida em componentes funcionais. Foram introduzidos na versão 16.8 da biblioteca, tornando os componentes funcionais a principal recomendação da comunidade. Os hooks devem ser utilizados somente no nível superior de uma função e nunca devem ser utilizados dentro de blocos.

useState


const [state, setState] = React.useState({});
setState((current) => current);

useEffect

```
1  React.useEffect(() => {
2      // execução de requisições assíncronas
3      (async () => {})();
4
5      // executa a função logo antes de desmontar o componente
6      return () => {};
7  }, []); // quando vazio executa uma única vez e se não passar executa sempre
```

useRef

```jsx
const Component = () => {
    const refUncontrolled = React.useRef();

    const handlerClick = () => {
      const { value } = refUncontrolled.current;
      refUncontrolled.current.focus();
    }

    return (
      <React.Fragment>
        <input ref={refUncontrolled} />
        <button onClick={handlerClick}>click</button>
      </React.Fragment>
    );
};
```

useContext

```jsx
const Ctx = React.createContext({});

const useCtx = () => {
  const context = React.useContext(Ctx);

  return { ...context };
};

const ProviderCtx = ({ children }) => {
  const [state, setState] = React.useState({});

  const change = (value) => setState({ value });

  return (
    <Ctx.Provider value={{ ...state, setState, change }}>
      { children }
    </Ctx.Provider>
  );
};
```

useReducer

```
const Component = () => {
    const [state, dispatch] = React.useReducer((state, action) => {
      const { type, payload } = action;

      switch (type) {
        case 'increment':
          return state + 1;
        case 'decrement':
          return state - 1;
        default:
          return state;
      }
    }, 0);

    const decrement = () => dispatch({ type: 'decrement' });
    const increment = () => dispatch({ type: 'increment' });

    return (
      <React.Fragment>
        {state}
        <button onClick={decrement}>-</button>
        <button onClick={increment}>+</button>
      </React.Fragment>
    );
};
```