HOOKS

What are Hooks?

Hooks are special functions that let you 'hook into' React features. In a sense, a hook "listens" for changes in order to trigger a response. Hooks give us the ability to use state, manage side-effects and use other features within functional components.

State

State is a variable where one can store and preserve property values that are pertinent to the application between function calls or for function components between re-renders.

Two kinds of state:

1. *Component state (useState)*
2. *Global State (Context, Redux)*

State does not persist after re-loading the page.

Side-effects

A "side effect" is anything that affects something outside the scope of the function being executed

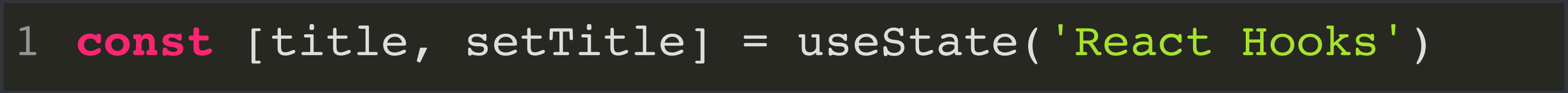
To manage these side-effects we will use the useEffect hook

Why Use Hooks

- Wasn't built into React originally
- Originally used Classes and Lifecycles
- Hard to reuse certain logic between components
- Complex components hard to understand
- Classes are confusing

useState Hook syntax

useState hook is a function that takes an optional default value as an argument and returns an array with two indexes



```
const [title, setTitle] = useState('React Hooks')
```

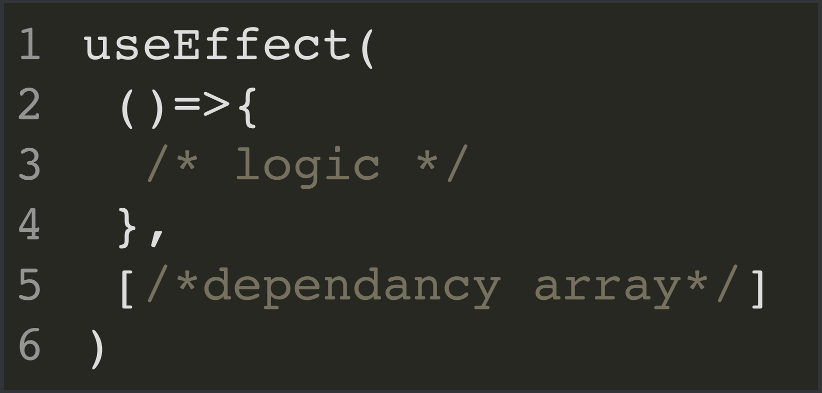
- **First index:** ***current state***
- **Second index:** ***function to update the state***

By convention we destructure these two values when defining our useState Hook.

useEffect Hook

function that handles asynchronous operations

Syntax: useEffect(func, array)



```
useEffect(
 ()=>{
  /* logic */
 },
 [/*dependancy array*/]
)
```

Great for data fetching, handling subscriptions and manually changing the DOM, & updating based on state or props

useContext Hook

Gives us a way to consume have global state from the Context API without using Redux.

NOT A REPLACEMENT FOR REDUX

Hook Rules

1. **Only Call Hooks from React Function Components**
2. **Hooks must always be called in the same order**
3. **Don’t call Hooks inside loops, conditions, or nested functions.**
4. **useEffect Hook MUST return a function**
5. **no async before the useEffect callback function**