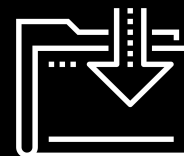
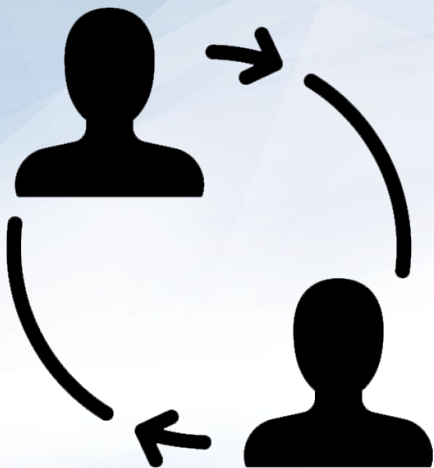




# Introduction to React Hooks

Web Development Boot Camp  
Lesson 10.3





## **Partner Activity:** Managing React State

With a partner, discuss the various methods used in React state management.

What are some of the advantages/disadvantages with these methods?

**Suggested Time:**  
5 minutes





Managing state can be difficult  
because there is no  
one-size-fits-all solution.

But there is another way.

# Comparing Ways to Manage State

---

01

Class Components with `setState()`

## Advantages

- Component and children will re-render with up-to-date data.

## Disadvantages

- Updating state from nested components can be difficult.
- Since state only flows one way, all components that need access to the state must be children of the same stateful component.

02

Functional Components with `useState()`

## Advantages

- Easier to read and debug, and no need to use `this`
- Access to Hooks

## Disadvantages

- Needs to use other Hooks to manage complex levels of state.
- Not supported by older codebases, which will still need to use class components for state.



**As of React 16.8, Facebook recommends using functional components whenever possible.**

# Introducing React Hooks

**Hooks** are functions that let you "hook into" React state and lifecycle features from stateless components.

# In This Lesson, We Will Cover Two Hooks

---

01

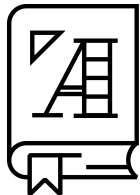
**useState:** Allows you to use state in a functional component.

02

**useEffect:** Replaces lifecycle methods like **componentDidMount** and **componentDidUpdate**.

03

**Custom Hooks:** Create your own reusable Hooks!



*Effect* is a term used to describe the result of affecting the “outside world.” This includes data fetching, subscribing to events, and making changes to the DOM.



# The Two Rules of Hooks

---

01

**Do not** call Hooks from within loops, conditionals, or nested functions.

- Hooks must always be called in the same order, like component lifecycle methods.
- This makes it possible for React to store the state of Hooks when using `useState` or `useEffect`.

02

**Do not** call Hooks from within regular JavaScript functions.

- This makes it so that all stateful logic is easy to find for the developer (you).

# <Time to Code>

