INFINUM

React Hooks

ADRIAN BIĆANIĆ

01

WHAT ARE REACT HOOKS

WHAT ARE REACT HOOKS

- Handle state, refs, lifecycles… without using class components
- Optional feature
- Backwards compatible - no breaking changes
- Classes remain supported

02

WHY REACT HOOKS

WHY REACT HOOKS

- No unnecessary wrapper components
- No combining a mix of unrelated logic inside a single lifecycle method
- Smaller functions with related logic
- Classes don’t minify well, and cause potentially unreliable hot reloading
- No binding
- No rewriting functional components

03

WHEN CAN I USE THEM

WHEN CAN I USE THEM

- Currently in alpha (React v16.8.0-alpha.1)
- Don’t require rewriting old components
- OK to try them out side by side with existing components

04

REACT HOOKS API

BASIC HOOKS

- useState
- useEffect
- useContext

CUSTOM HOOKS

ADDITIONAL HOOKS

- useReducer
- useCallback
- useMemo
- useRef
- useImperativeHandle
- useLayoutEffect
- useDebugValue

useState

```
const [state, setState] = useState(initialState);

....

setState(newState);
```

useState

```
function Counter({initialCount}) {
  const [count, setCount] = useState(initialCount);
  return (
    <>
      Count: {count}
      <button onClick={() => setCount(initialCount)}>Reset</button>
      <button onClick={() => setCount(prevCount => prevCount + 1)}>+</button>
      <button onClick={() => setCount(prevCount => prevCount - 1)}>-</button>
    </>
  );
}
```

useState

```
setState(prevState => {
  // Object.assign would also work
  return {...prevState, ...updatedValues};
});
```

useState

```
const [state, setState] = useState(() => {
  const initialState = someExpensiveComputation(props);
  return initialState;
});
```

useEffect

```
useEffect(handleSomeSideEffect);
```

useEffect

```
useEffect(() => {
  const subscription = props.source.subscribe();
  return () => {
    // Clean up the subscription
    subscription.unsubscribe();
  };
});
```

useEffect

```
useEffect(
  () => {
    const subscription = props.source.subscribe();
    return () => {
      subscription.unsubscribe();
    };
  },
  [props.source],
);
```

useEffect

```
useEffect(
  () => {
    const subscription = props.source.subscribe();
    return () => {
      subscription.unsubscribe();
    };
  },
  [props.source],
);
```

useEffect

```
useEffect(() => {
    ...

    handleSomeDataFetching();

    ...
  },
  [],
);
```

useContext

```
const context = useContext(Context);
```

BASIC HOOKS

- useState
- useEffect
- useContext

CUSTOM HOOKS

ADDITIONAL HOOKS

- useReducer
- useCallback
- useMemo
- useRef
- useImperativeHandle
- useLayoutEffect
- useDebugValue

INFINUM

Any questions?

ADRIAN.BICANIC@INFINUM.CO

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum

INFINUM

Thank you!

ADRIAN.BICANIC@INFINUM.CO

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum