Dynamic Web

Introduction to React - Hooks

Built-in React Hooks

useMemo: “Memoizes” data for display. Lets you cache the result of an expensive calculation.

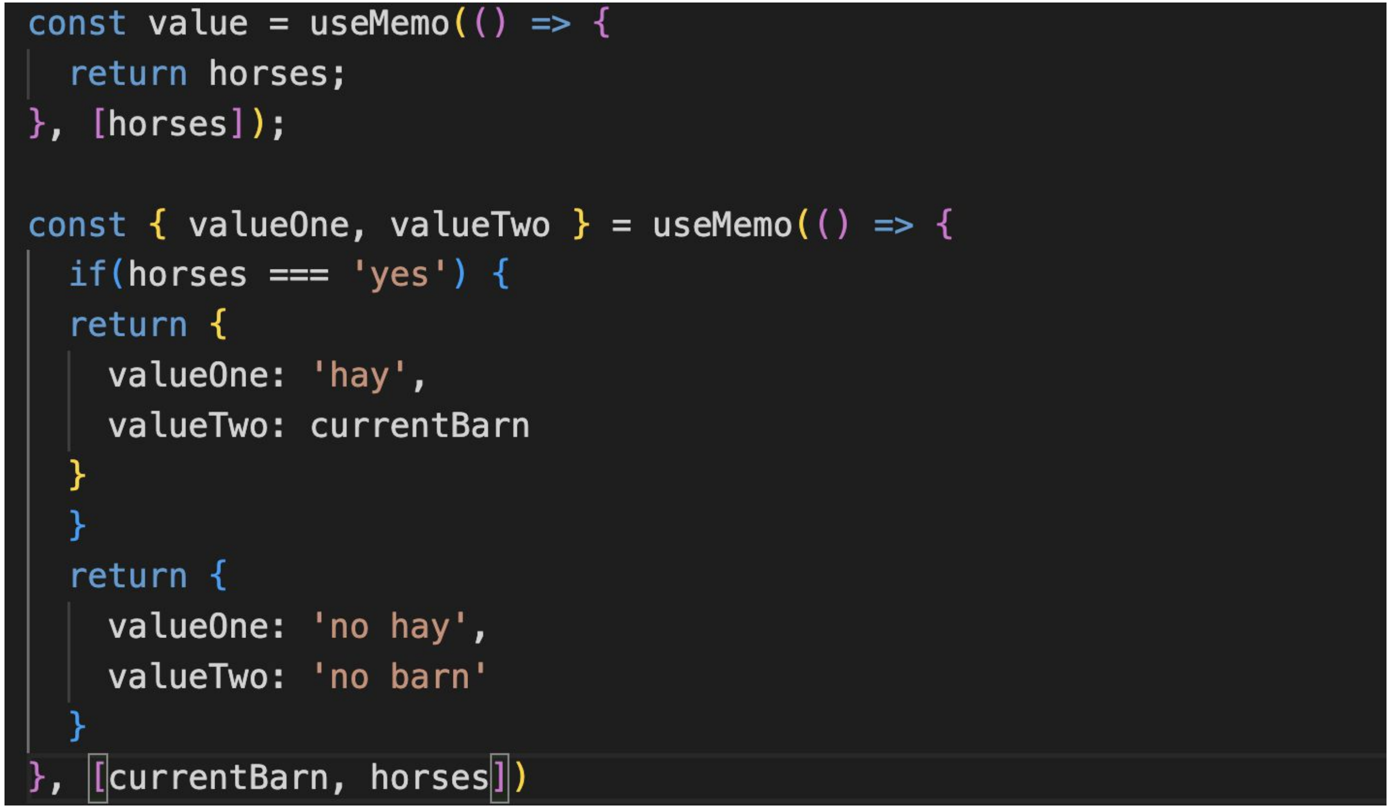
useEffect: Executes code based on a condition

useState: Stores a value in “state” for access elsewhere

useCallback: lets you cache a function definition before passing it down to an optimized component.

https://react.dev/reference/react/hooks

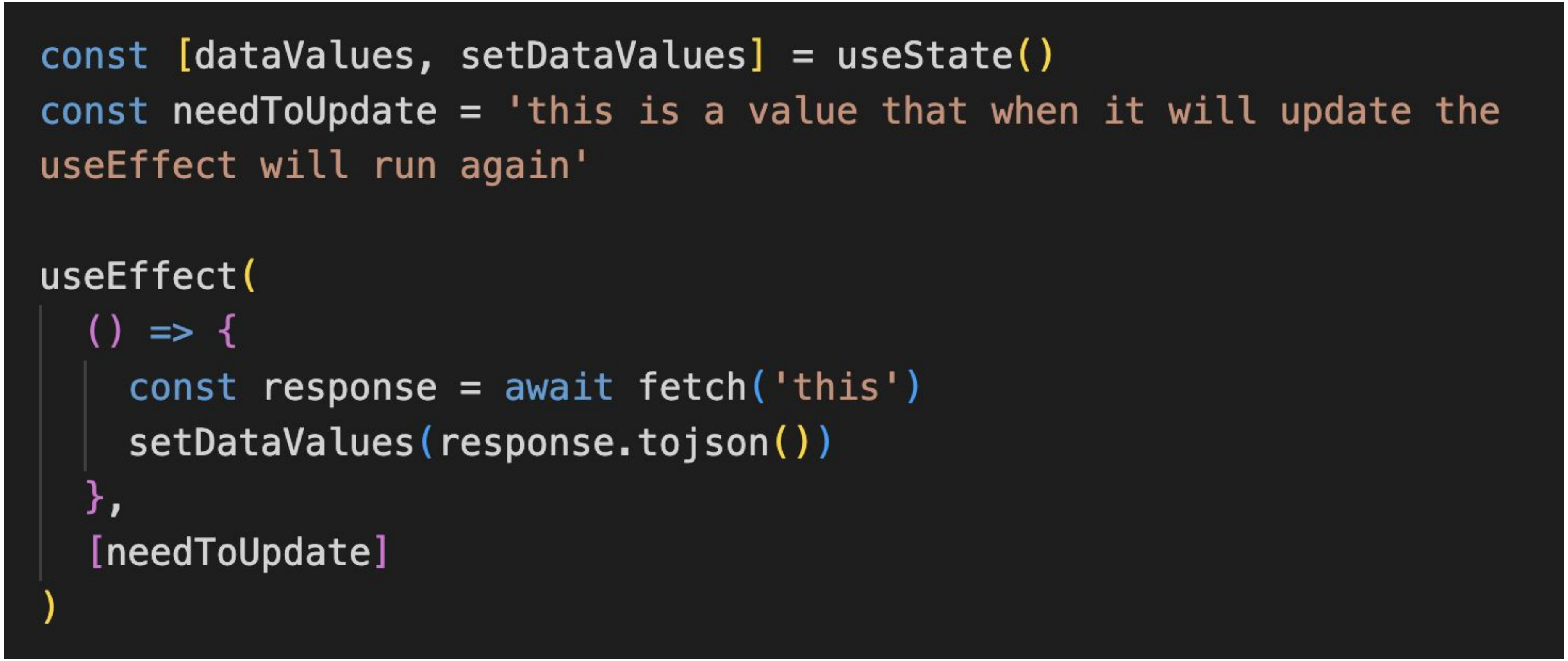
useMemo



```
const value = useMemo(() => {
  return horses;
}, [horses]);

const { valueOne, valueTwo } = useMemo(() => {
  if(horses === 'yes') {
  return {
    valueOne: 'hay',
    valueTwo: currentBarn
  }
  }
  return {
    valueOne: 'no hay',
    valueTwo: 'no barn'
  }
}, [currentBarn, horses])
```

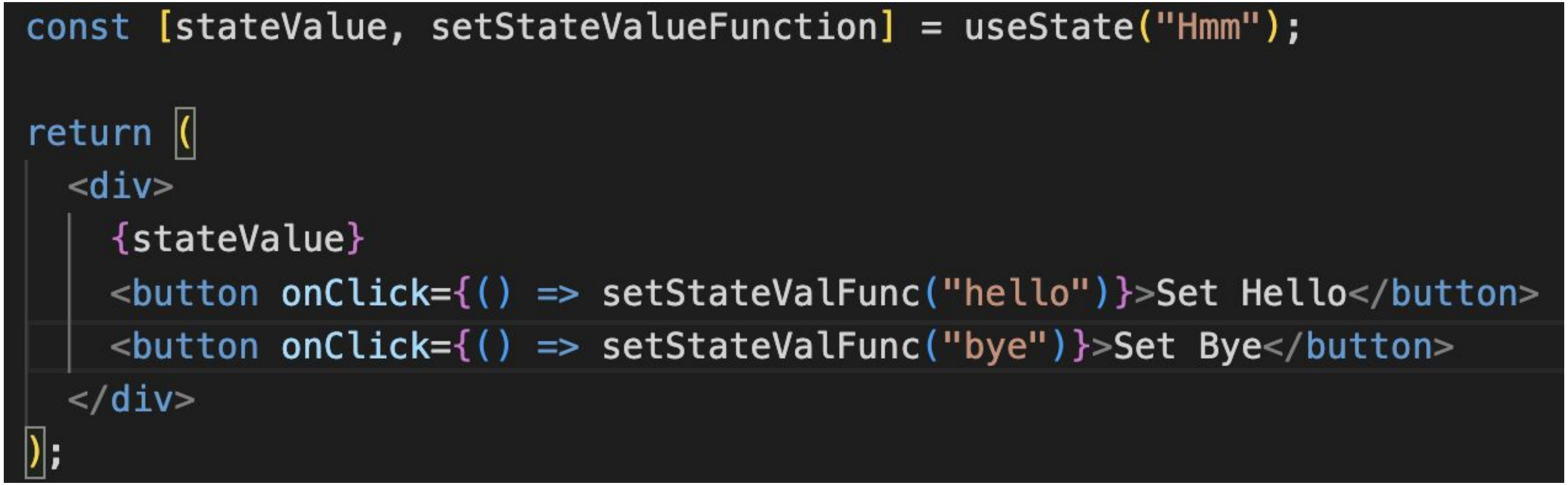
useEffect



```
const [dataValues, setDataValues] = useState()
const needToUpdate = 'this is a value that when it will update the
useEffect will run again'

useEffect(
  () => {
    const response = await fetch('this')
    setDataValues(response.tojson())
  },
  [needToUpdate]
)
```

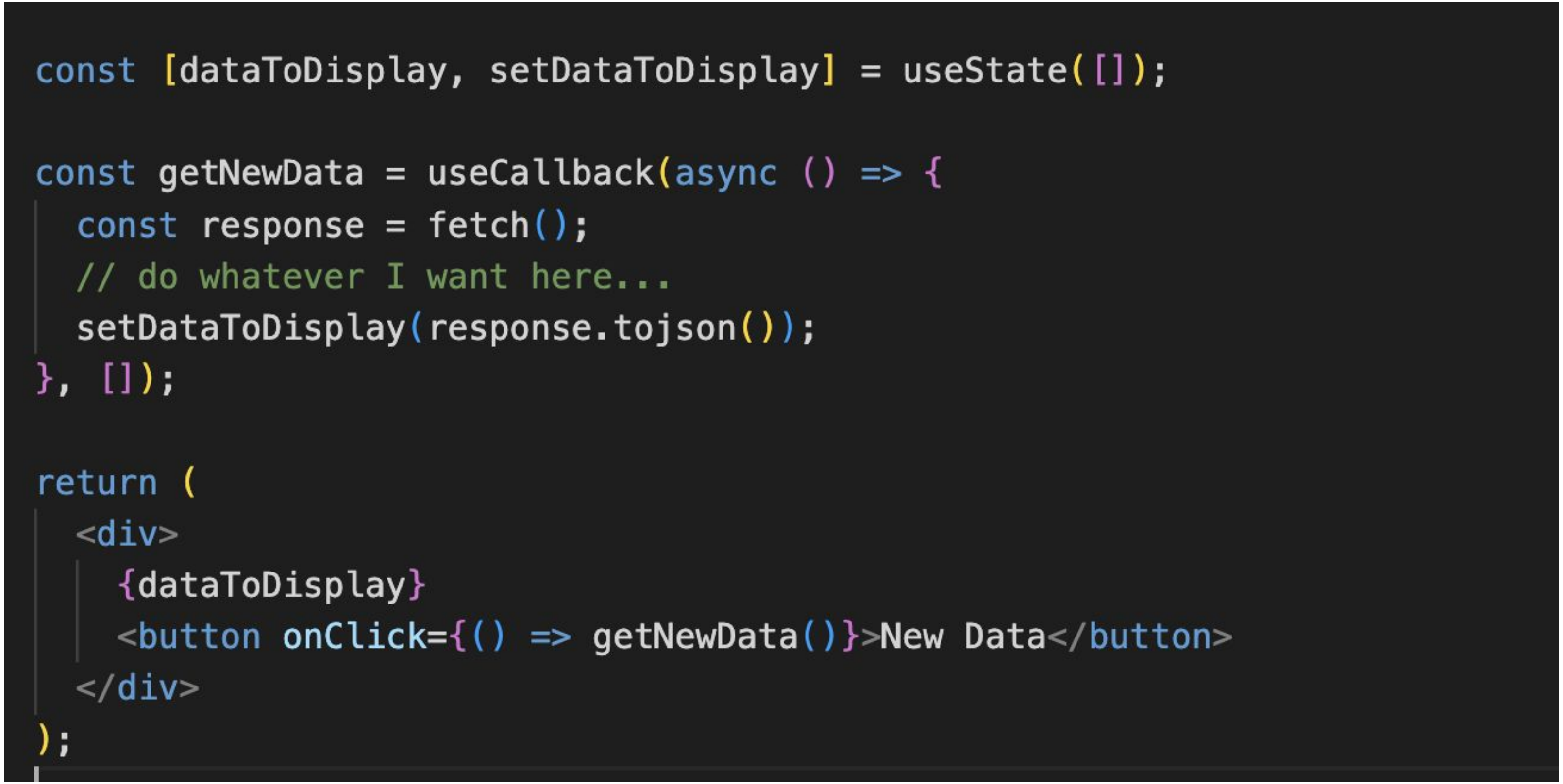
useState



```
const [stateValue, setStateValueFunction] = useState("Hmm");

return (
  <div>
    {stateValue}
    <button onClick={() => setStateValFunc("hello")}>Set Hello</button>
    <button onClick={() => setStateValFunc("bye")}>Set Bye</button>
  </div>
);
```

useCallback



```
const [dataToDisplay, setDataToDisplay] = useState([]);

const getNewData = useCallback(async () => {
  const response = fetch();
  // do whatever I want here...
  setDataToDisplay(response.tojson());
}, []);

return (
  <div>
    {dataToDisplay}
    <button onClick={() => getNewData()}>New Data</button>
  </div>
);
```