ANY JS value can be a state variable

- Strings
- Numbers
- Booleans
- ARRAYS!
- OBJECTS!

What do you save as state?

- Anything that can change during run of app
  - Unless based solely on other state
    - Known as **derived state**
    - Ex: A class name based on a state value
    - Ex: Number of items in a state array
    - Don't store derived state
    - Calculate it from the state when needed

Immutable State

- Objects and Arrays have special fact
    - They can **mutate**
    - Change contents without changing identity
- State is used to make rendering decisions
    - Compares "previous" state to "current" state
    - Should elements recreate from components?
- State MUST be **immutable**
    - State Objects/Arrays should never mutate
    - Replace with copies that have changes

Changing Arrays in State

YES:

```
const [names, setNames] = useState( ['Jorts', 'Jean'] );
//...
function addName(newName) {
  setNames( [...names, newName ] ); // set to a new array!
}
```

NO:

```
const [names, setNames] = useState( ['Jorts', 'Jean'] );
//...
function addName(newName) {
  names.push(newName); // BAD! mututes existing array!
  setNames(names); // new array and old array are the same!
}
```

Changing Objects in State

YES:

```
const [cat, setCat] = useState( { name: 'Jorts' });

function updateCat(age) {
  setCat( { ...cat, age } ); // set to new object!
}
```

NO:

```
const [cat, setCat] = useState( { name: 'Jorts' });

function updateCat(age) {
  cat.age = age; // BAD! Mutates existing object!
  setCat( cat ); // new object and old object are the same!
}
```

Remember the timing of setting State

- Incredibly common mistake!

```
const [count, setCount] = useState(1);

return (
  <button
    onClick={ () => {
      setCount( count + 1 );
      console.log(count); // What does this output?
    }}
  > {count} </button>
);
```

- Calling `setCount()` does NOT change `count`!
- Changes `count` from `useState()` next render
    - Which will happen because you called setter
    - But hasn't happened yet

How to get new value of state before next render?

- You literally just set it, you have the value!

Original:

```
onClick={ () => {
  setCount( count + 1 );
  console.log(count); // What does this output?
}}
```

Knowing the next value:

```
onClick={ () => {
  const nextCount = count + 1;
  setCount( nextCount );
  console.log( nextCount ); // What does this output?
}}
```