

What is the State Initialization Pattern?

Here's a quick introduction to patterns in React with state initialization!

WE'LL COVER THE FOLLOWING ^

- Setting the Value of the State
- Quick Quiz!


Setting the Value of the State

To initialize means to set the value of something. Well, technically, initialization is setting the value of something right when it is created/defined and any subsequent value setting is referred to as ‘assignment’. However, we’re going with the first definition to make things simpler. The state initializer pattern exists to make it easy for the consumer of your custom hook to set the “value of state”.

Note that the state initializer pattern doesn’t give full control over setting the “value of the state” every single time. It mostly allows for setting initial state resetting it.

This is not the same as having full control over setting the state value, but it offers similar benefits as you’ll see soon.

In our implementation of the custom hook, we’ve done the following:

```
export default function useExpanded () {  
  // look here   
    const [expanded, setExpanded] = useState(false)  
    ...  
}
```



We’ve assumed that the initial state passed to the `useState` call will be `false` every single time.

That may not be the case.

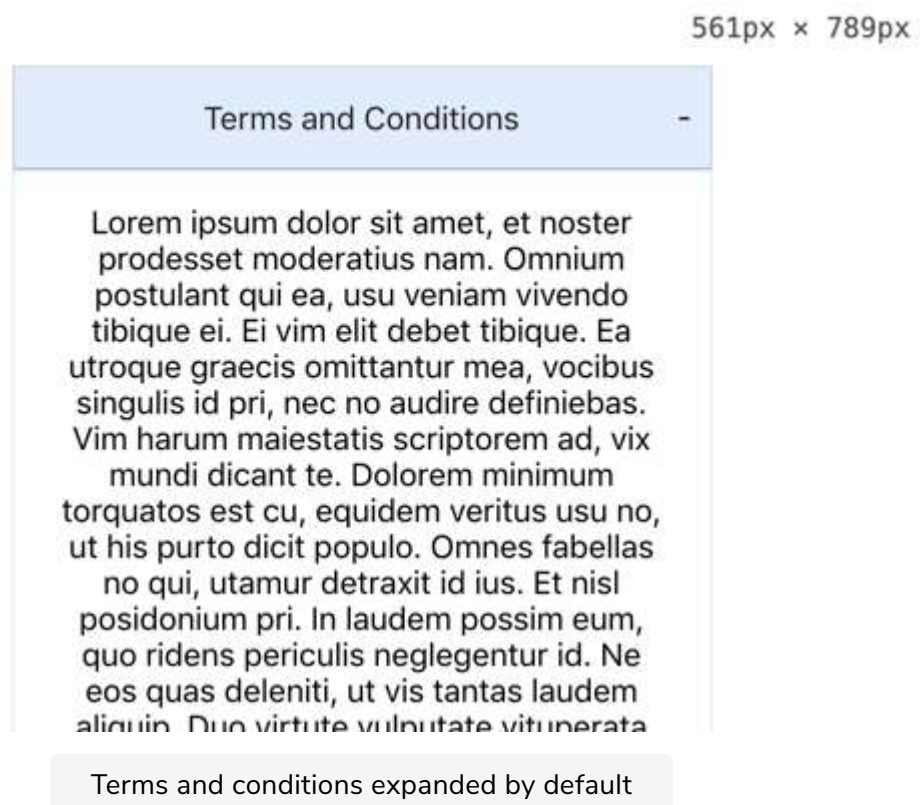
Let's have this value controllable from an argument the user can pass in. Let's call this parameter `initialExpanded` as seen below:

```
export default function useExpanded (initialExpanded = false) {  
  const [expanded, setExpanded] = useState(initialExpanded)  
  ...  
  return value  
}
```

Now the user can pass in `initialExpanded` into the `useExpanded` custom hook call to decide the initial state of the expanded component.

Here's an example where a user may want the terms and conditions content expanded by default:

```
// user's app  
function App () {  
  // look here - "true" passed into hook call  
  const { expanded, toggle } = useExpanded(true)  
  return (  
    ...  
  )  
}
```



With our previous setup, this was not an easy feat for the user because we had

internally hardcoded the initial state as `false`.

As you can see, this is a simple but helpful pattern.

Quick Quiz!

Time for a quiz!

Q

What key problem does this pattern address?

COMPLETED 0%

1 of 1

<

✓

In the next lesson, we'll have a look at how the state can be reset!