REACT JS

Form Controls

Sreekanth M. E.

Freelance Trainer & Consultant

http://www.SreekanthME.com

UNCONTROLLED COMPONENTS

There can be an attribute named ref on a component instance or DOM node which takes a callback as its value.

The callback gets reference to the component instance or DOM node.

This reference can be used to modify that component instance or DOM node.

React will call the ref callback with the DOM node / Component instance during mounting, and call it with a null during unmounting.

Using the ref callback just to set a property on the class is a common pattern for accessing nodes.

Here is an example:

```
<input type="text“ ref={
    (node) => {
    this.textInput = node;
    }
} />
```

Using this.refs.<key> instead of ref callback pattern is an obsolete way.

Likely to be removed in one of the future releases.

Functional components cannot have ref attribute.

Only components created with class can have ref attribute.

A functional component must be converted to a class if ref is needed.

A ref must not be accessed inside any component's render method

Form controls naturally keep some internal state.

They remember what was typed and their value can be accessed using refs.

Such components are called “Uncontrolled” components.

Uncontrolled component keeps the source of truth in the DOM

Specifying initial value to an uncontrolled component in JSX prevents any future updating of the component value.

If initial value has to be specified then use:

- defaultValue: in case of text, select & textarea.
- defaultChecked: in case of checkbox and radio support

CONTROLLED COMPONENTS

The other alternative is to make the form controls controlled (by React).

Then the React component that renders a form also controls what happens in that form on subsequent user input.

React controls the form control's state either in component's state, or in parent component's state or in a state management system like Redux.

With a controlled component, every change in its value is updated to the state through a handler function.

And any relevant change to the state is reflected back to the component’s value.

This means data (state) and UI (inputs) are always in sync.

DETAILS NEEDED TO MAKE FORM CONTROLS CONTROLLED

Control	Default Value	Change Callback	New Value in Callback
Text	value	onChange	event.target.value
Checkbox	checked	onChange	event.target.checked
Radio	checked	onChange	event.target.checked
Textarea	value	onChange	event.target.value
Select	value	onChange	event.target.value

END OF CHAPTER

APPENDIX