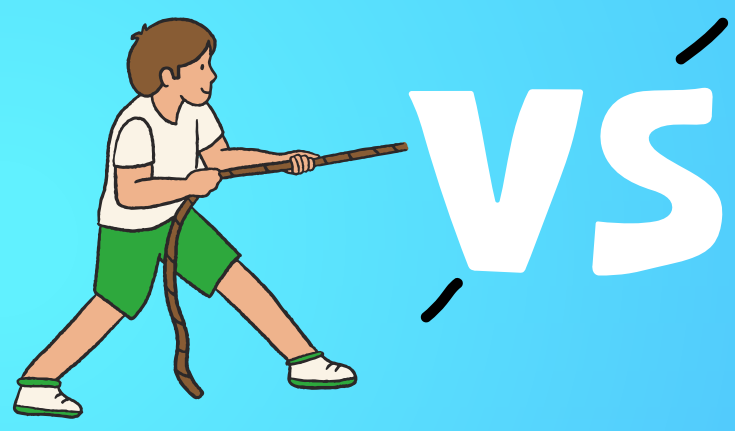
aayush-jain-code

Controlled



VS

Uncontrolled

Components in

ReactJS

aayush-jain-code

Controlled Components

Controlled components are forms that are handled entirely by a React component and a hook called useState.

useState is a hook that automatically refreshes the application whenever a change is made to its state variable (prop).

Form elements become controlled elements once the value is set to a prop.

It will track every change done to the form (every input down to the letter)

```
import React, { useState } from "react";

function Form() {
    const [name, setName] = useState("");

    console.log(name);

    return (
        <form>
            <input type="text" value={name} onChange=
                {(e) => setName(e.target.value)} />
            <button type="submit">Submit</button>
        </form>
    );
}

export default Form;
```

aayush-jain-code

Uncontrolled Components

Uncontrolled components are controlled by the DOM instead of an individual component.

Uncontrolled components use the useRef attribute. You can use useRef as a way to access values from the DOM.

We first bring in the useRef hook then we use a ref attribute in our JSX to give the current value to our ref variable.

This gives us access to the submitted form information.

```
import React, { useRef } from "react";

function Form() {

    const ref = useRef();

    const handleSubmit = (e) => {
        e.preventDefault();
        console.log(ref.current.value)
    };

    return (
        <form>
            <input type="text" ref={ref} />
            <button type="submit" onClick=
               {handleSubmit}>Submit</button>
        </form>
    );
}

export default Form;
```

in aayush-jain-code

Controlled Components Use Case

This is useful in cases where you need features such as:

- Search Functionality.
- Implement Throttling.
- Forcing specific characters (numbers only, no special characters, etc).
- Validations

Uncontrolled Components Use Case

This is useful in cases where you need features such as:

- Media playback
- Maintaining focus
- Selecting text
- Integrating 3rd party DOM libraries
- Imperative animations

Differences

Controlled Components	Uncontrolled Components
The parent component holds control over the data.	DOM itself holds control over the data.
There is validation control.	There is no validation control.
The internal state is not maintained.	The internal state is maintained.
Store the current value in the form of the **prop**.	Use **ref** for the current values.
Predictable because the component handles the form elements state.	Not predictable because the form elements can lose their reference and may be changed / affected by other sources during the lifecycle of a component.
Allow substantial control over the elements and data of the form.	Allow limited control over the elements and data of the form.