**❓ Is the props keyword necessary in React components?**

**✅ Short Answer:**

**No, props is not a required keyword** — it’s just a **conventionally used variable name** to access properties passed to a component.

What *is* required is:

* You receive the **first argument** in a **functional component** to access the props.
* You can name it anything, but props is the most readable and widely used name.

**🔍 Example: Using Custom Name Instead of props**

const Welcome = (xyz) => {

return <h2>Hello, {xyz.name}!</h2>;

};

And call it like:

<Welcome name="Manav" />

✅ This works fine.

**✅ Best Practice: Destructure for Clarity**

Instead of using props.name, many developers use **destructuring**:

const Welcome = ({ name }) => {

return <h2>Hello, {name}!</h2>;

};

Still the same under the hood — the props object is just unpacked directly in the function parameter.

**🔁 In Class Components**

Yes, props is the default name used like:

class Welcome extends React.Component {

render() {

return <h2>Hello, {this.props.name}</h2>;

}

}

In this case, you can’t change the name props because it’s a **property of the Component superclass**.

**✅ Summary**

| **Case** | **Is props necessary?** | **Notes** |
| --- | --- | --- |
| Functional Component | ❌ No | You can rename it (e.g. data, info) |
| Destructuring | ❌ No | Use { name } directly |
| Class Component | ✅ Yes | Must use this.props |