React Props Cheat Sheet

by Ndeye Fatou Diop

1. What are Props?

- Props (short for "properties") are used to pass data from parent components to child components.
- They are read-only and cannot be modified inside the child component.

```
<MyComponent propName="value" />
```

2. Passing Props to Components

You pass props from a parent to a child component via attributes in JSX.

```
function ParentComponent() {
  return <ChildComponent name="John" age={25} />;
}

function ChildComponent(props) {
  return <div>Hello, {props.name}. You are {props.age} years old.</div>;
}
```

3. Accessing Props in Functional Components

Props can be accessed in a functional component using the props object.

```
function Greeting(props) {
  return <h1>Hello, {props.name}!</h1>;
}
```

4. Destructuring Props

Props can be destructured for easier access.

```
function Greeting({ name }) {
  return <h1>Hello, {name}!</h1>;
}
```

5. Default Props

You can set default values for props.

```
function Greeting({ name = "Anonymous" } = {}) {
  return <h1>Hello, {name}!</h1>;
}

// Renders "Hello, Anonymous!"
<Greeting />
```

6. Prop Drilling

Passing props down multiple levels of components can become cumbersome. This is called `prop drilling`.

```
function Parent() {
  const name = "John";
  return <Child name={name} />;
}

function Child({ name }) {
  return <Grandchild name={name} />;
}

function Grandchild({ name }) {
  return <p>{name}</p>;
}
```

7. Conditional Rendering with Props

Props can be used in conditional rendering inside a component.

```
function Greeting({ name, isLoggedIn }) {
  return isLoggedIn ? <h1>Welcome back, {name}</h1> : <h1>Please log
in</h1>;
}
```

8. Spread Operator for Props

The spread operator can be used to pass all props at once.

```
function Profile(props) {
    return <p>{props.name}, {props.age}</p>;
}
```

```
const user = { name: "John", age: 25 };
<Profile {...user} />
```

9. Handling Events with Props

You can pass event handlers as props to handle user interactions.

```
function Button({ onClick }) {
  return <button onClick={onClick}>Click me</button>;
}

function App() {
  const handleClick = () => alert("Button clicked!");
  return <Button onClick={handleClick} />;
}
```

10. Props vs State

- Props are passed to a component and are immutable.
- State is local to the component and can be changed.

```
import { useState } from 'react';

// Child Component: Receives props from the parent
function ChildComponent({ name }) {
  return <h2>Hi, my name is {name}.</h2>;
}

// Parent Component: Holds state and passes it as props to the child
function ParentComponent() {
  // State to manage the name value
  const [name, setName] = useState('John');

  const changeName = () => {
    setName('Jane');
  };

  return (
    <div>
      <ChildComponent name={name} />
      {/* Change state when button is clicked */}
      <button onClick={changeName}>Change Name</button>
    </div>
  );
}
```