React Lecture - 4

Props

Props

What are Props ?

React components use props to communicate with each other. Every parent component can pass some information to its child components by giving them props.

Props might remind you of HTML attributes, but you can pass any JavaScript value through them, including objects, arrays, and functions.

Props are the information that you pass to a JSX tag.

Passing Props to a Component

We can pass props to a component similar to how we add attributes on HTML elements.

Think of props like HTML attributes, but with more flexibility.

The attribute name corresponds to the prop name, and the value assigned becomes the prop value.

Passing Props to a Component

You can access the props value in the Component by adding a **props** parameter to the function.

The props parameter will be an object containing all the props that you have passed to the component

In this example, the App component is the parent, and it passes name props with different values to the Greeting child component.

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

const App = () => {
  return (
    <div>
      <Greeting name="Alice" />
      <Greeting name="Bob" />
    </div>
  );
};
```

Destructuring Props

For better readability and managing complex props, you can use destructuring syntax within the child component function. This allows you to extract individual props from the props object passed as an argument.

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

Default value for a prop

If you want to give a prop a default value to fall back on when no value is specified, you can do it with the destructuring by putting = and the default value right after the parameter:

```
function Avatar({ person, size = 100 }) {
  // ...
}
```