**How can we make the Component reusable?**

When we write functions in JavaScript, then we make the functions reusable by accepting parameters. And that allows us to call one of the same functions with different parameter values. And, therefore the function typically will produce different results for different input values, but it's still always the same function being called. And the React basically has the same concept built in. We can make our components reusable by using parameters and a concept called props in React.

**Understanding Props:**

Let's say in your App Component, you have a goalItem variable or constant defined which holds a string of Finish! And then let's say you have a custom Component, the CourseGoalItem Component, which has a list item inside of it where this goalItem should be output.

The problem is that the goalItem variable lives in the App Component, not in the CourseGoalItem Component. And to a certain extent, that is good because it makes the CourseGoalItem independent if it doesn't store the concrete value internally. But, we want to define what's being output by CourseGoalItem with help of variable managed in App.

And we don't have direct access to the HTML code output by some Component in other Components. We can't just send our stored data there, but instead we can utilize a concept called Props. We can pass data to the custom Component by adding an attribute. And inside of that Component, we can then get access to all these attributes which might have been set on our custom Component. And props simply stand for properties.

We can set properties of our own custom Components. Now, how does this props concept work though? In ExpenseItem, we have the date, the title and the amount and we'd like to output this here. But the data shouldn't be stored in ExpenseItem.js but instead in App.js.

In App.js Component, we could have another constant or variable, expenses, which is an array. So multiple ExpenseItems and let's say every expense in here is a simple JavaScript object, which now has a title like Car Insurance which also has an amount like the 294.67 and where we have a date.

After defining the array of object, we want to pass the data of these different objects to these different ExpenseItems. That means we wanna make these ExpenseItems configurable from outside. The data should not be stored inside of them but instead received from the outside. And that works with this props concept I just mentioned. In App.js, we can simply add attributes to these custom HTML elements.

For example, we can add a title attribute to ExpenseItems and set this to a value of Toilet Paper. But in the end, I wouldn't be just hard coding some data in the JSX code. Since, we already have the expenses array here, I also can dynamically retrieve the title stored in the first ExpenseItem.

There the first item with index zero, and it has a title property. So, we can access it like this:



Now this title is being passed to this ExpenseItem. We can do same for the amount attribute and date attribute. You have to make sure that the part after the dot (.) matches with the property names in the array object. Inside of ExpenseItem.js we also need to do something with these received attribute values. Now the question is how do we get access to the values defined in the place where we use our custom Components?

And the answer is parameters. I mentioned that in regular JavaScript, we use parameters for passing data into functions, and it's kind of similar for React. However, we're not getting a title, amount and date parameters but instead we get one parameter. React will ensure that we get one parameter in every Component which we use as a Component.

And that one parameter will be an object which holds all the received attributes as properties, hence the name props for the overall concept. Therefore, we get one parameter, and you can name this parameter whatever you want. It's your function you could name it data. But typically, it's named props to make it clear that this is the object which holds all the values we get for the attributes on our custom element.

And to be precise, we get key-value pairs in the props object, which is passed in by React automatically. The keys will be the attribute names defined, so title, amount and date. And the values will be the values set here. So, if I now wanna output the title received, in h2 tag I can now access **{props.title}**. If you choose a different name than the title here, for example "name" then you have to access props.name here. That's how you share data between React Components.