

------------------------------------------------------------------------------------------------

\*The whole idea with React is this simple idea that we are going to make use of Components / Functions. This is why when you have a look at how we write our React code we have functions that have a return statement.

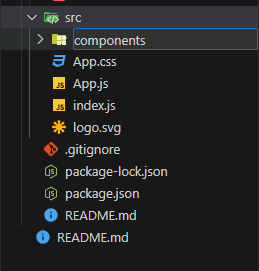
\*and the reason why we need to work with functions is very simple, functions are re-usable, and this means that we can create a function and then re-use it by calling it.

\*The basic idea with React is that we are writing Java script code, we are writing a Java script function that returns HTML.

------------------------------------------------------------------------------------------------

**How to Create our first Component/ function:**

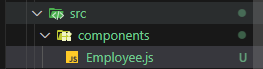
Step 1: Inside of our src folder, we are going to create a new folder called components.



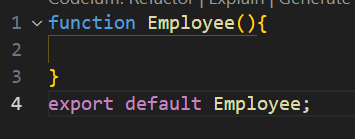
**Step 2**: Inside of this components folder, which is just a folder that will contain our components, we are going to create a component/java script file.

\*what we want to do is that we want to create a java script function that is called Employee, and this means that we need to name the file as “Employee.js”

\*this is very similar to working with classes in Java, because when we create a class in java, we have to name the file with the exact same name as the class name.



**Step 3**: we need to create this function called Employee

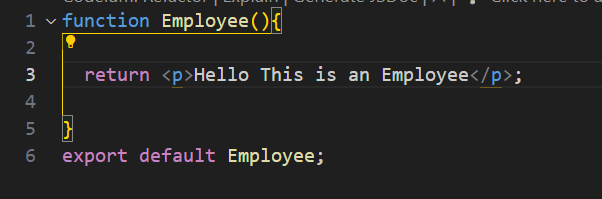


\*in order to create a function in java script we have to use the keyword “function” and then we have to give the name of the function, in this case the name of the function is “Employee” which matches with the name of the component file that we created.

\*it is a function so it must have the 2 brackets() that allow the function to take a parameter.

\*you also need that last piece of code at the bottom, the one that says export default Employee;

**Step 4**: we need to code the body of the function



\*I know that every Java script function must have a “return” keyword. Which is basically what I want the function to return / output. And in this case I want this function to simply output HTML paragraph text.

\*So all I have here is a Java Script function that returns HTML Paragraph text.

\*and I know that every Java script statement needs to end in a semi-colon.

**Step 5**: I need to use this component / Call the function:

1. The first thing that we need to take note of here is that a Java script function, is called inside of another Java script function.

\*The java script component that I want to call this Java script component, is called App.js

\*what you need to take note of here is simply the following, if we go back to the basic of Object Orientated programming, we have a main() function that becomes the entry point to our program. Meaning that this is the first function that is executed. This means that App.js is the main function. It is the function that we will use to call our component function.

\*the very first thing that we need to do inside of our Main function / App.js function is that we need to import our component file so that we can have access to the functions that are inside of the file.

\*So what we do is that we copy the relative path, and then we adjust it.

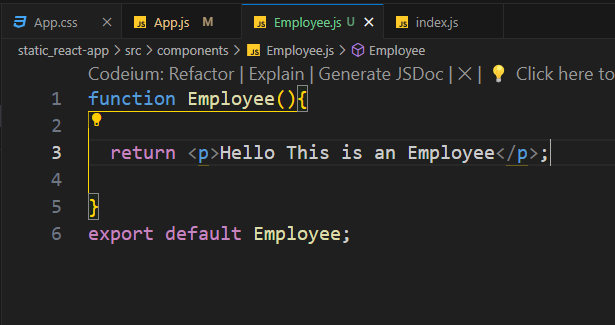


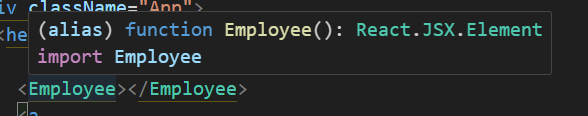
\*to adjust this all we want is the folder name and the file name.



\*we don’t use back-slash we use forward-slash. And then in front of the first forward-slash we have to put a dot.

**JSX**

****



\*If we hover above the function call for the function called Employee() this is what the intellisense is able to pick up.

\*what the intellisense picks up is that Employee() is a custom function, it is a component it is an alias. That’s what the words alias basically mean, they are there to tell us that we have a custom function, we have an alias.

\*Not only is the intellisense able to tell us that we have a custom function that we needed to import, but its able to tell us what the function returns. And in this case the function returns a React.JSX.Element, its not returning a html element, its returning a React.JSX.Element which in our case is equivalent to HTML.

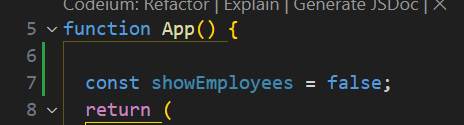
------------------------------------------------------------------------------------------------

Ternary Operators in Java Script:

\*what Java script allows us to do is to create a Boolean variable outside of our return statement, we are then able to code logic inside of the return statement that depends on the evaluation of the Boolean variable that we declared outside of the return statement.

How to create and use a Ternary operator in Java script:

1. The very first thing that we have to is that we need to go and create a constant variable outside of the return statement, and this constant variable has to be a Boolean.



\*that’s exactly what we have done here, we have created this constant variable called “showEmployees”, we have used the keyword “const” in order to show that it’s a constant variable, it’s a variable whose value cannot be changed later on. And we have assigned it the value false.

return(

<div>

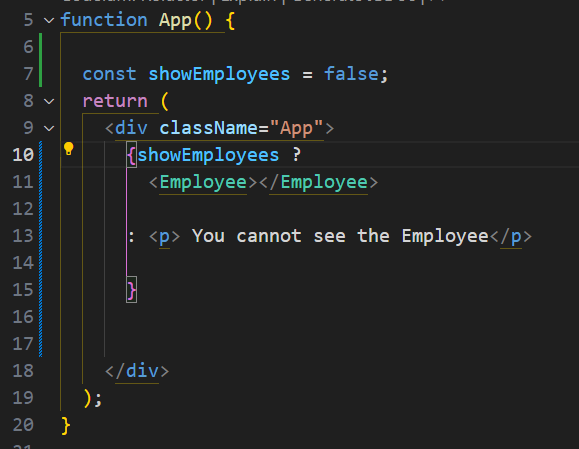
{

//java script code that uses a variable declared outside the return

}

</div>

);



------------------------------------------------------------------------------------------------

PROPS in React:

Props allow us to pass data from a parent component to a child component.

\*Props are very important, because they allow us to do 2 important things that we know from programming

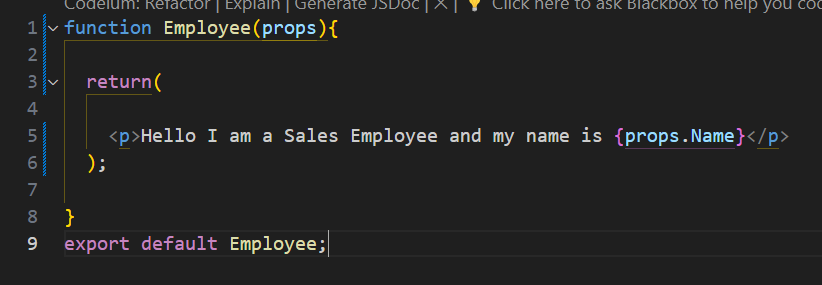
1) Props allow us to create a component / function that takes an arbitrary parameter.

\*this means that when we create the Employee() function, we can now create this Employee function with an arbitrary/prop parameter.

\*what we need to keep in mind here is that java script has what we call automatic type inference. This means that when we create a data type in java script, we don’t have to explicitly state the data type of the variable that we create, java script will look at what value we have assigned to the given variable and then based on the value that we have assigned to the given variable, java script will determine the data type of that variable.

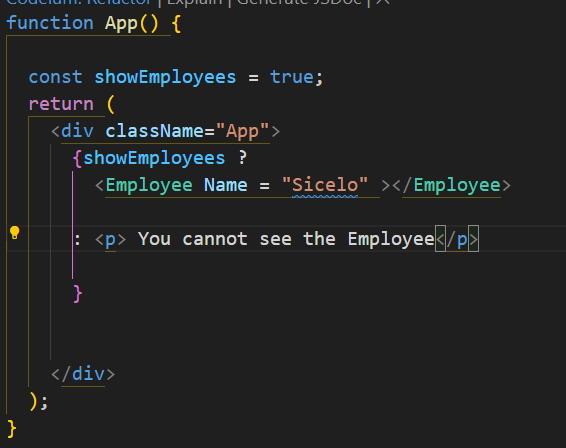
\*Once I have created my props, all I now have to do is to use the props inside of my JSX. And I use it as an Object, where I am able to give it a unique name, that identifies what the prop actually is.

\*the word props just means generic.



\*because I have a function that takes a props/generic parameter. What I need to do is that when I call this function, I need to provide an argument for the parameter.

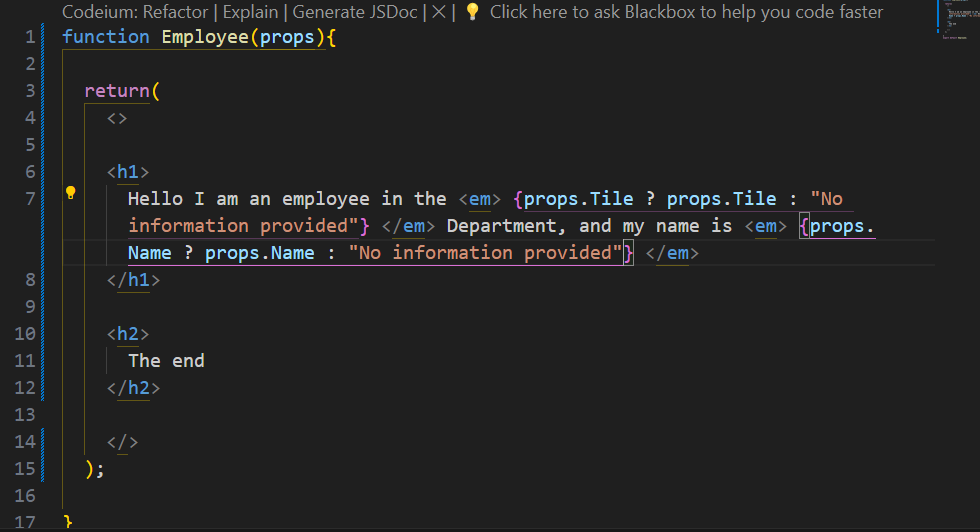
\*In this case I know I know that the function <Employee/> has a props parameter called props.Name, so when I call this <Employee> function inside of my main function, I need to provide an argument for this props parameter.



\*you can create as many props parameters as you need to, you just need to ensure that you when you call the function, you are able to provide arguments for all of the props parameters that you created In the order you create them on.

\*So you have basically created a props function, and this means that you have created a component/function that is a template, and its only supplied with information/data when it is called.

\*The word props simply means that we are creating a template that is only passed information when it is called, and again here we see the overall theme when it comes to React which is re-usability.



-----------------------------------------------------------------------------------------------