ReactJS - Nested Components

As we learned earlier, React component is the building block of a React application. A React component is made up of the multiple individual components. React allows multiple components to be combined to create larger components. Also, React components can be nested to any arbitrary level. Let us see how React components can be composed in this chapter.

FormattedMoney component

Let us create a component, *FormattedMoney* to format the amount to two decimal places before rendering.

Open our *expense-manager* application in your favorite editor.

Next, Create a *FormattedMoney.js* file in the *src/components* folder and, Import React library.

```
import React from 'react';
```

Next, create a class, *FormattedMoney* by extending *React.Component*.

```
class FormattedMoney extends React.Component
{  }
```

Next, introduce construction function with argument props as shown below −

```
constructor(props)
{  super(props);
}
```

Next, create a method *format* to format the amount.

```
format(amount) {  return
parseFloat(amount).toFixed(2)  }
```

Next, create a method *render* to emit the formatted amount.

```
render()
{ return (
<span>{this.format(this.props.value)}</span>
);
}
```

Here, we have used the *format* method by passing *value* attribute through *this.props*.

Next, specify the component as default export class.

```
export default FormattedMoney;
```

Now, we have successfully created our *FormattedMoney* React component.

```
import React from 'react';

class FormattedMoney extends
React.Component {    constructor(props) {
super(props)
   }    format(amount) {       return
parseFloat(amount).toFixed(2)
   }
render() {
return (
         <span>{this.format(this.props.value)}</span>
      );
   }
}
export default FormattedMoney;
```

FormattedDate Component

Let us create another component, *FormattedDate* to format and show the date and time of the expense.

Open our *expense-manager* application in your favorite editor.

Next, create a file, *FormattedDate.js* in the *src/components* folder.

Next, import *React* library.

```
import React from 'react';
```

Next, create a class by extending *React.Component*.

```
class FormattedDate extends React.Component
{  }
```

Next, introduce construction function with argument props as shown below −

```
constructor(props)
{  super(props);
}
```

Next, create a method format to format the date.

```
format(val) {    const months = ["JAN", "FEB", "MAR","APR", "MAY", "JUN",
"JUL", "AUG", "SEP", "OCT    let parsed_date = new Date(Date.parse(val));
let formatted_date = parsed_date.getDate() +
      "-" +
months[parsed_date.getMonth()] +
"-" + parsed_date.getFullYear()    return
formatted_date;
}
```

Create a method *render* to emit the formatted date.

```
render() { return ( <span>{this.format(this.props.value)}</span> ); }
```

Here, we have used the format method by passing value attribute through *this.props*.

Next, specify the component as default export class.

```
export default FormattedDate;
```

Now, we have successfully created our FormattedDate React component. The complete code is as follows −

```
import React from 'react'; class
FormattedDate extends React.Component {
constructor(props) {       super(props)
   }    format(val) {       const months = ["JAN", "FEB", "MAR","APR", "MAY",
"JUN", "JUL", "AUG", "SEP", "       let parsed_date = new
Date(Date.parse(val));       let formatted_date = parsed_date.getDate() +
         "-" +
months[parsed_date.getMonth()] +
"-" + parsed_date.getFullYear()
return formatted_date;
   }
render() {
return (

         <span>{this.format(this.props.value)}</span>
      );
   } } export default
FormattedDate;
```