ReactJS - State Management API

As we learned earlier, React component maintains and expose it's state through *this.state* of the component. React provides a single API to maintain state in the component. The API is *this.setState()*. It accepts either a JavaScript object or a function that returns a JavaScript object.

The signature of the *setState* API is as follows −

```
this.setState( { ... object ...} );
```

A simple example to set / update name is as follows −

```
this.setState( { name: 'John' } )
```

The signature of the *setState* with function is as follows −

```
this.setState( (state, props) =>
... function returning JavaScript object ... );
```

Here, *state* refers the current state of the React component

- *props* refers the current properties of the React component.
-

React recommends to use setState API with function as it works correctly in async environment. Instead of lambda function, normal JavaScript function can be used as well.

```
this.setState( function(state, props) {
return ... JavaScript object ...  }
```

A simple example to update the amount using function is as follows −

```
this.setState( (state, props) => ({
amount: this.state.amount + this.props.additionaAmount  })
```

React state should not be modified directly through this.state member variable and updating the state through member variable does not re-render the component.

A special feature of React state API is that it will be merged with the existing state instead of replacing the state. For example, we can update any one of the state fields at a time instead of updating the whole object. This feature gives the developer the flexibility to easily handle the state data.

A special feature of React state API is that it will be merged with the existing state instead of replacing the state. For example, we can update any one of the state fields at a time instead of updating the whole object. This feature gives the developer the flexibility to easily handle the state data.

For example, let us consider that the internal state contains a student record.

```
{  name: 'John', age:
16
}
```

We can update only the age using setState API, which will automatically merge the new object with the existing student record object.

```
this.setState( (state, props) => ({  age:
18
});
```