strategy | consulting | digital | technology | operations

REACTJS – MODULE 3

COMPONENT

accenture

AGENDA

- Introduction to Component
- Props
- State
- Difference between props and state
- Component life cycle
- Types of component
- Demo
- Activity

accenture

Introduction to component

React allows us to create independent, reusable, maintainable and testable smaller unit known as “Component”

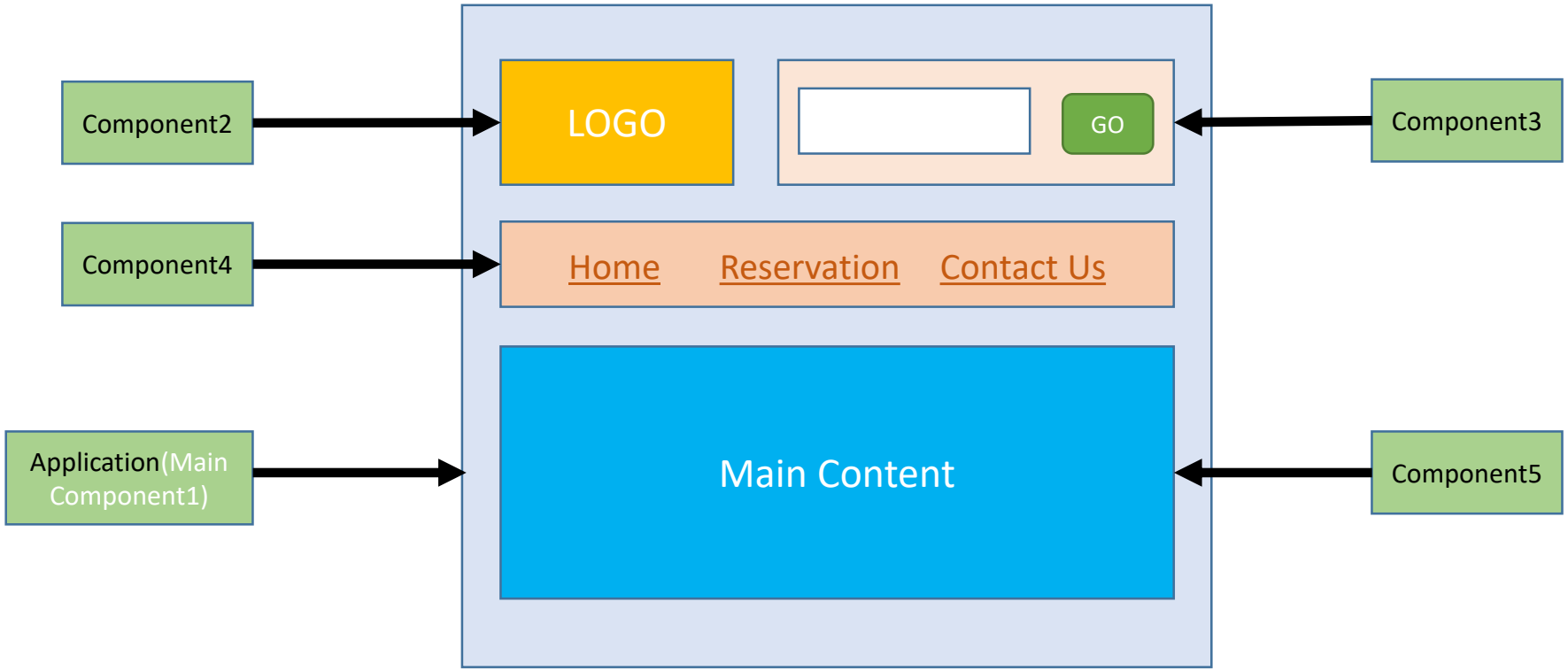
React applications are nothing but collection of components

Component represent the view/ UI of an application

Components are considered as main building block in ReactJS

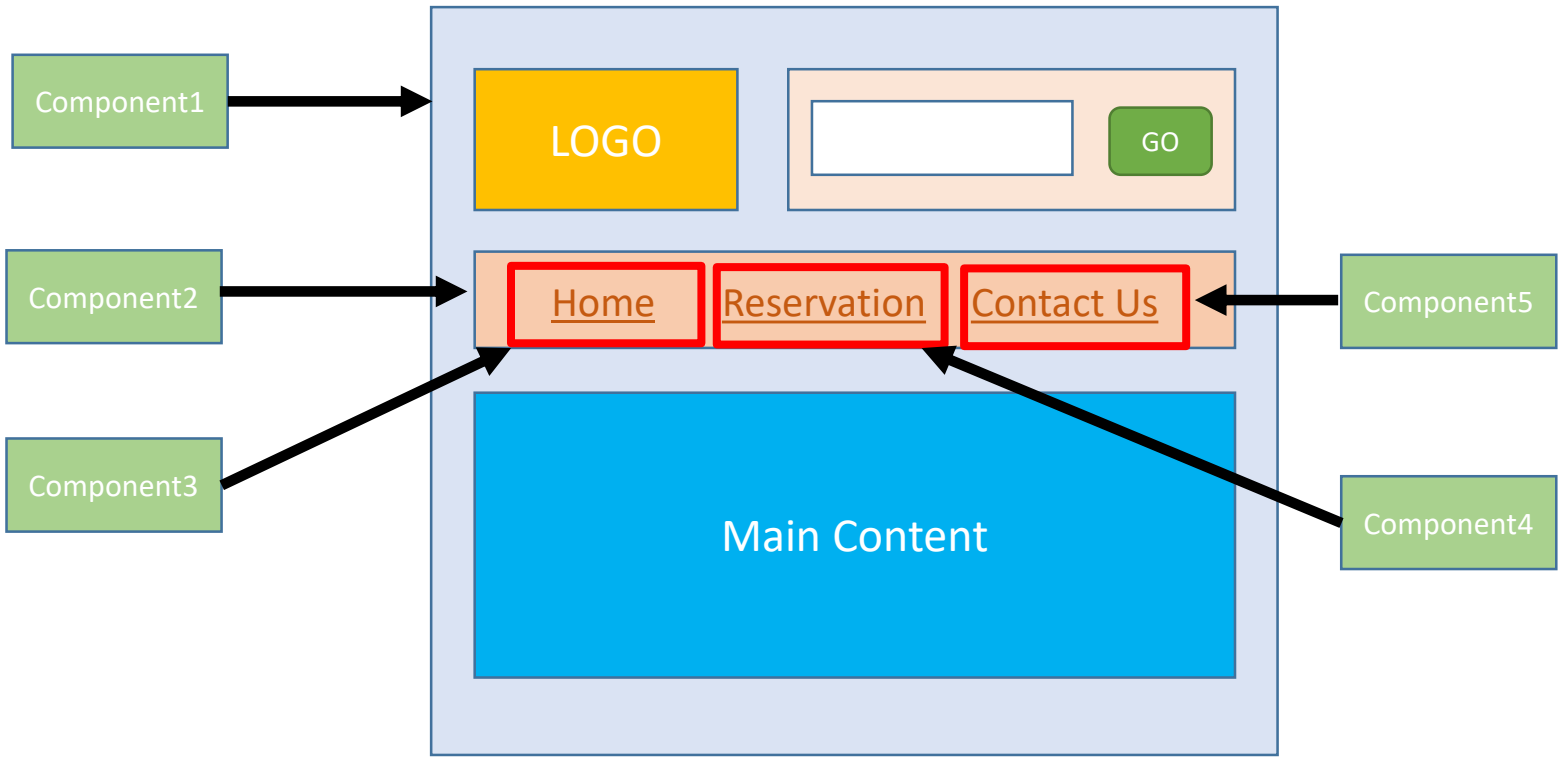
Introduction to component

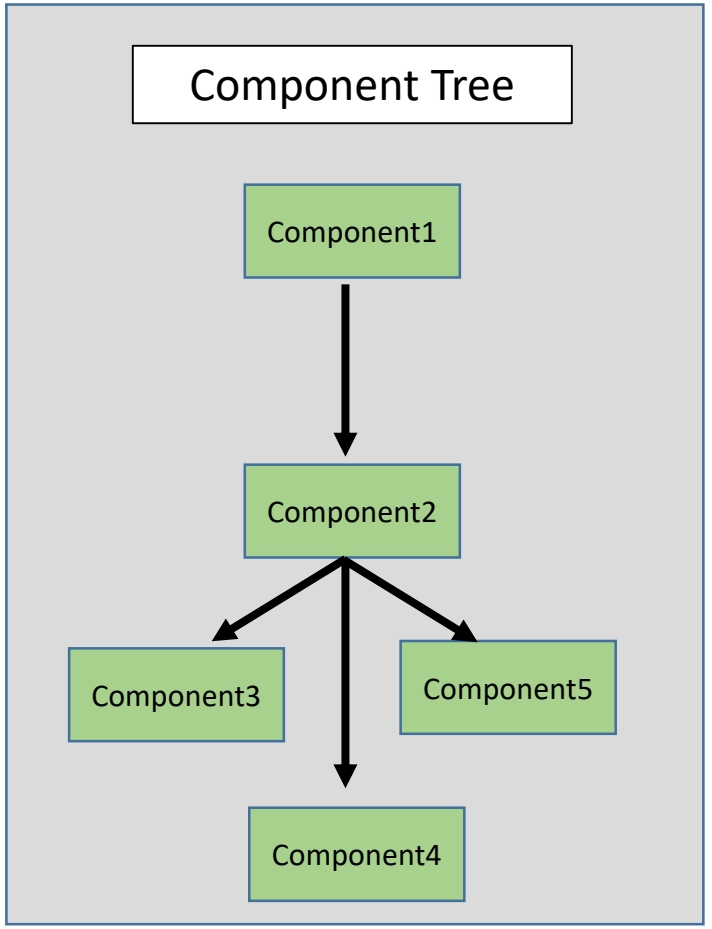
The complete application when broken down into smaller parts(component) it becomes easy to maintain. This way react application can be modularized



Component Tree

When the component is rendered(displayed) inside other component, this will form component tree





How To Create component

Create Hello component(Hello.js)

```
import React,{Component} from 'react';

class Hello extends Component{

    render(){
        return(

            <div>
                <h3>Hello Component</h3>
            </div>
        )
    }
}

export default Hello;
```

How To Create component

Create Hello component(Hello.js)

```
import React,{Component} from 'react';

class Hello extends Component{

    render(){
        return(

            <div>
                <h3>Hello Component</h3>
            </div>
        )
    }
}

export default Hello;
```

Import the definition of component from react library

Hello is a custom component extended from Component

Render is a method which can return a single element(here it is element)

Export Hello component so that it can be imported when required

How To Create component

App component(App.js)

```
import React, { Component } from 'react';
import Hello from './Hello';

class App extends Component {
  render() {
    return (
      <div>
          <h2>App Component</h2>
        <Hello/>
      </div>
    );
  }
}

export default App;
```

How To Create component

App component(App.js)

```
import React, { Component } from 'react';
import Hello from './Hello';

class App extends Component {
  render() {
    return (
      <div>
          <h2>App Component</h2>
        <Hello/>
      </div>
    );
  }
}

export default App;
```

Import Hello component

The UI of Hello component is captured

How To Create component

Index component(index.js)

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import registerServiceWorker from './registerServiceWorker';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
```

How To Create component

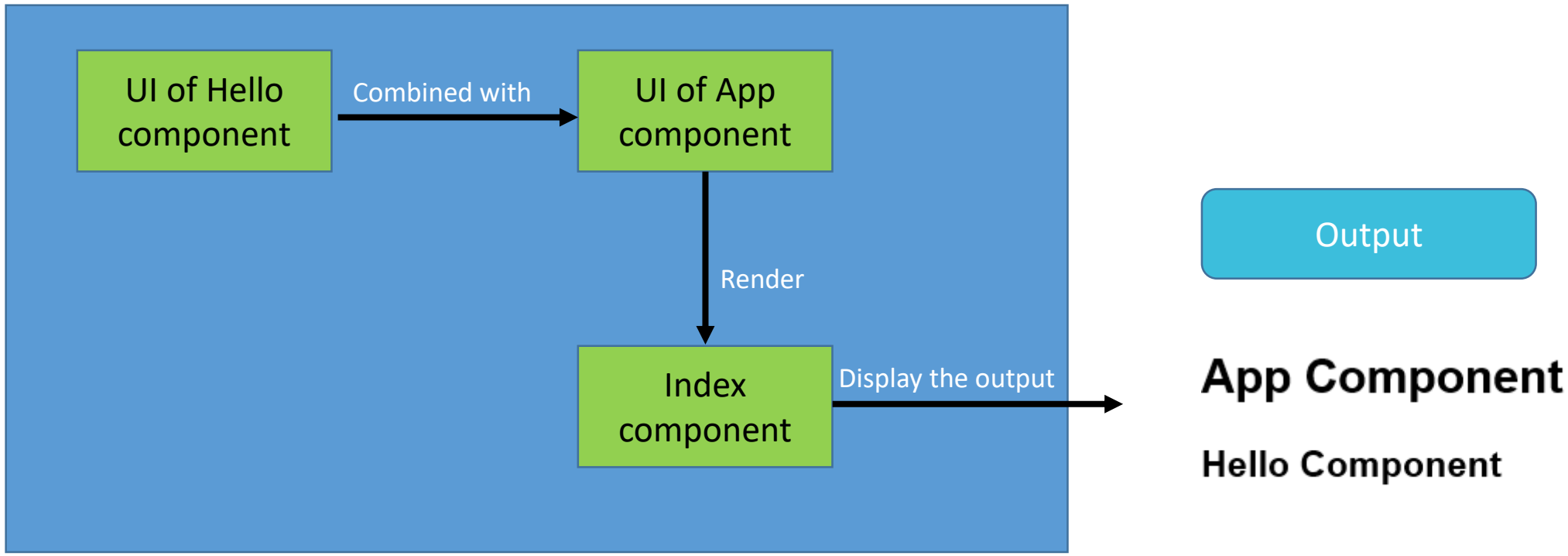
Index component(index.js)

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import registerServiceWorker from './registerServiceWorker';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
```

This is required to use DOM specific methods

App component is rendered inside root element

How To Create component



Props in Component

Props are considered as arguments/parameter to a component

Using props we can also customize the component

Components can be reused by passing different Props

Props are defined by parent component, once defined with the value it cannot be changed(Props are read-only).

accenture

Props in Component

App.js

```
import React, { Component } from 'react';
import Hello from './Hello';

class App extends Component {
  render() {
    return (
      <div>
          <h2>App Component</h2>
          <Hello name="Accenture"/>
      </div>
    );
  }
}

export default App;
```

Hello.js

```
import React,{Component} from 'react';

class Hello extends Component{

    render(){
        return(

            <div>
                <h3>Hello Component {this.props.name}</h3>
            </div>
        )
    }
}

export default Hello;
```

Props in Component

App.js

Parent component

```
import React, { Component } from 'react';
import Hello from './Hello';

class App extends Component {
  render() {
    return (
      <div>
          <h2>App Component</h2>
          <Hello name="Accenture"/>
      </div>
    );
  }
}

export default App;
```

Hello.js

Child component

```
import React,{Component} from 'react';

class Hello extends Component{

    render(){
        return(

            <div>
                <h3>Hello Component {this.props.name}</h3>
            </div>
        )
    }
}

export default Hello;
```

accenture

Props in Component

App.js

```
import React, { Component } from 'react';
import Hello from './Hello';

class App extends Component {
  render() {
    return (
      <div>
          <h2>App Component</h2>
          <Hello name="Accenture"/>
      </div>
    );
  }
}

export default App;
```

Hello.js

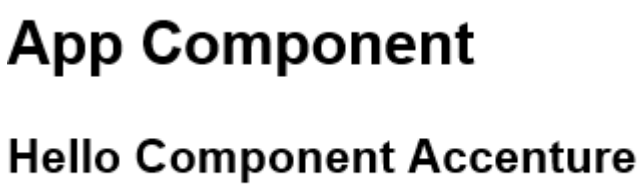
```
import React,{Component} from 'react';

class Hello extends Component{

                <h2>Hello Component</h2>
                {this.props.name}</h3>
            </div>
        )
    }
}

export default Hello;
```

Here name is a parameter it is passed to Hello component with the value "Accenture"

Props(properties) are accessed using this.props.propname

Props in Component

Output



State in Component

States are considered as changing data of a component

States can be initialized only inside the constructor of a component

Using setState one can modify the state of a component

State in Component

Hello.js

```
import React,{Component} from 'react';

class Hello extends Component{

    constructor(){
        super();
        this.state={
            noOfEmps:2000
        }
    }

   render(){
       return(

           <div>
               <h3>Hello Component {this.props.name}</h3>
               <h3>{this.props.name} has {this.state.noOfEmps} Employees</h3>
           </div>
       )
   }
}

export default Hello;
```

State in Component

Hello.js

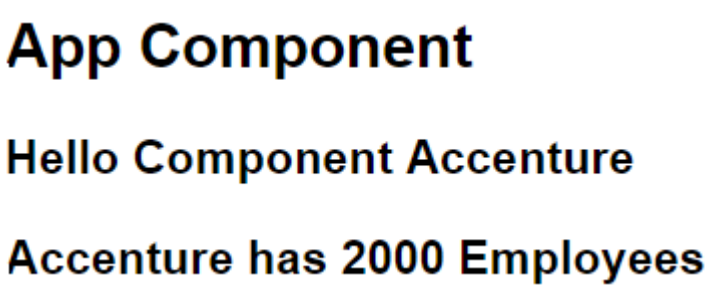
```
import React,{Component} from 'react';

class Hello extends Component{

    constructor(){
        super();
        this.state={
            noOfEmps:2000
        }
    }

    render(){
        return(

            <div>
                <h3>Hello Component {this.props.name}</h3>
                <h3>{this.props.name} has {this.state.noOfEmps} Employees</h3>
            </div>
        )
    }
}

export default Hello;
```

State of a component is initialized inside constructor. Here “noOfEmps” is the state of Hello component

This is the way to access state of a component

State in Component

Output



State in Component – Modify State

Hello.js

```
import React,{Component} from 'react';

class Hello extends Component{

    constructor(){
        super();
        this.state={
            noOfEmps:2000
        }
    }
    modifyState(){
        this.setState({
            noOfEmps:3000
        })
    }
   render(){
       return(

           <div>
               <h3>Hello Component {this.props.name}</h3>
               <h3>{this.props.name} has {this.state.noOfEmps} Employees</h3>

              <button onClick={()=>this.modifyState()}>Click to Modify State</button>

           </div>
       )
   }
}
export default Hello;
```

State in Component – Modify State

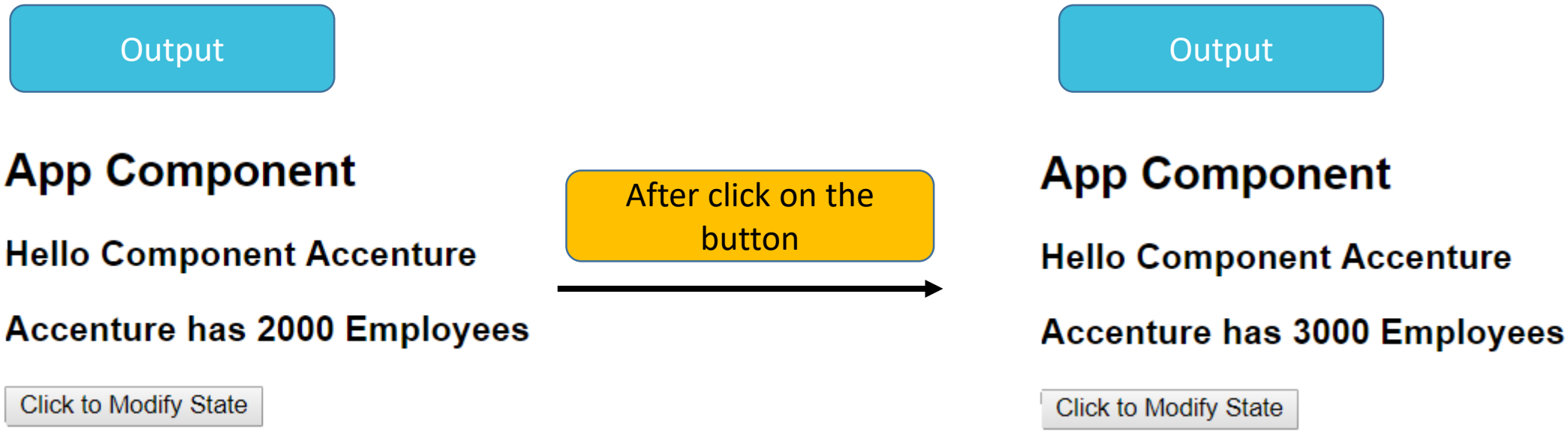
Hello.js

```
import React,{Component} from 'react';

class Hello extends Component{

    constructor(){
        super();
        this.state={
            noOfEmps:2000
        }
    }
    modifyState(){
        this.setState({
            noOfEmps:3000
        })
    }
    render(){
        return(

            <div>
                <h3>Hello Component {this.props.name}</h3>
                <h3>{this.props.name} has {this.state.noOfEmps} Employees</h3>

                <button onClick={()=>this.modifyState()}>Click to Modify State</button>

            </div>
        )
    }
}
export default Hello;
```

setState is used to modify the state of a component.

When the button is clicked modifyState method is called which will change the state

State in Component – Modify State



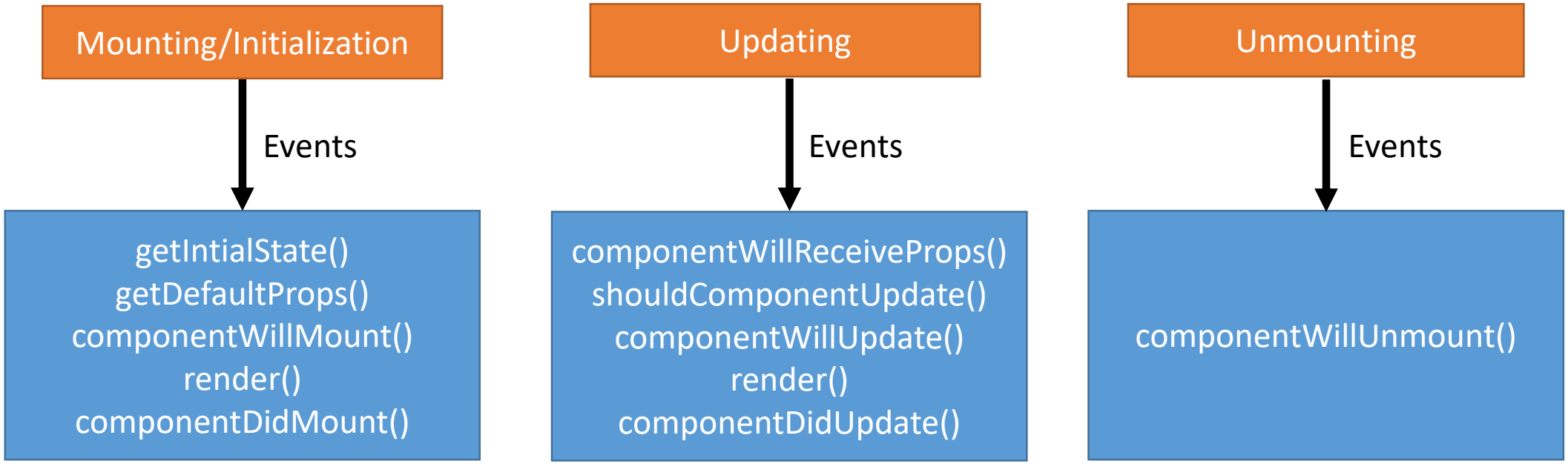
Difference between Props and State

Props	State
Passed as a parameter from parent component to child component	Defined inside a component. Defines the state of the component
Fixed value, once defined cannot be changed	Data(State) can be changed using setState
Accessed using {this.props.propname}	Accessed using {this.state.statename}

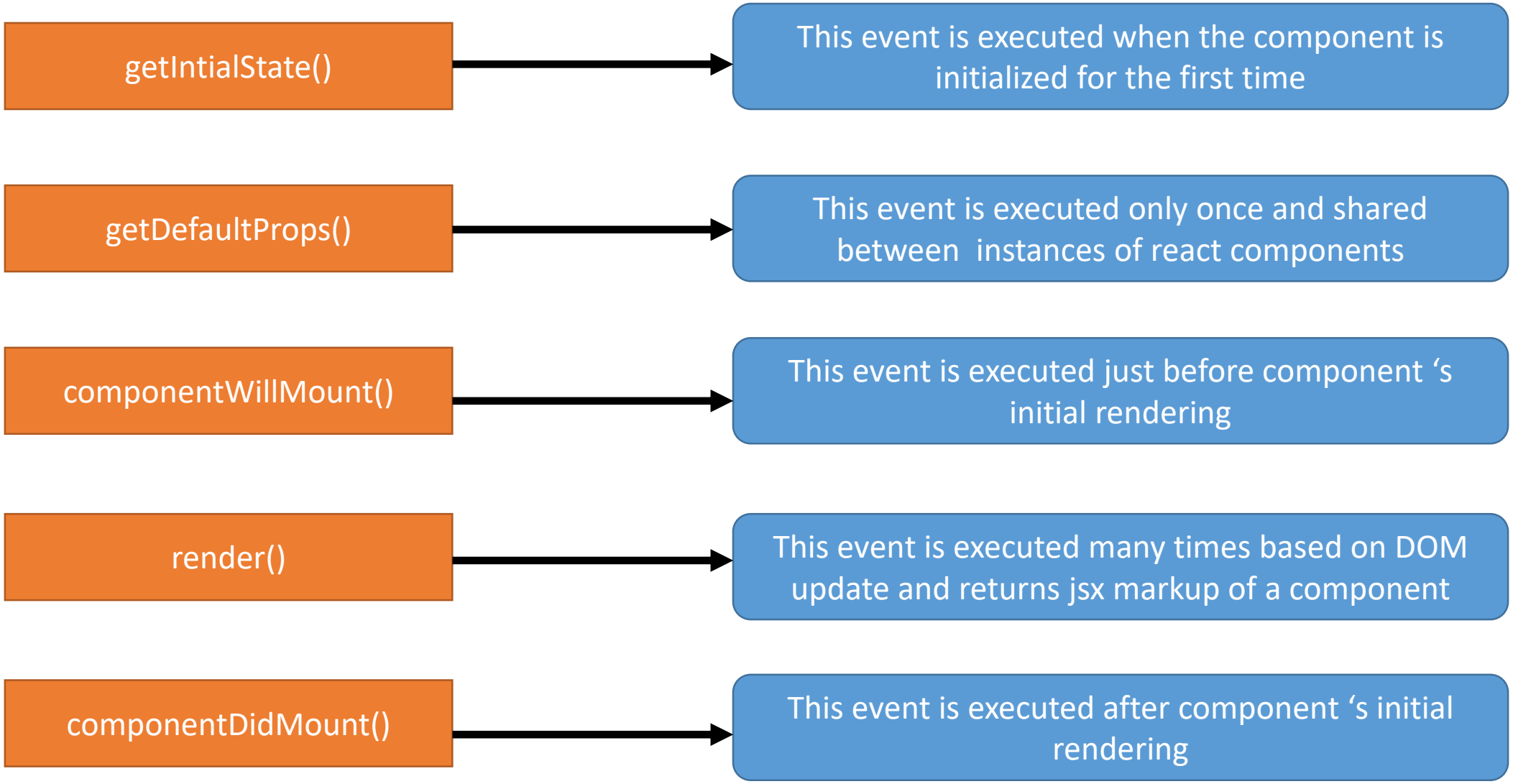
Component Life Cycle

Every component in react will have life cycle events which are executed based on the requirement

These events are basically categorized into three categories and each category will have related events

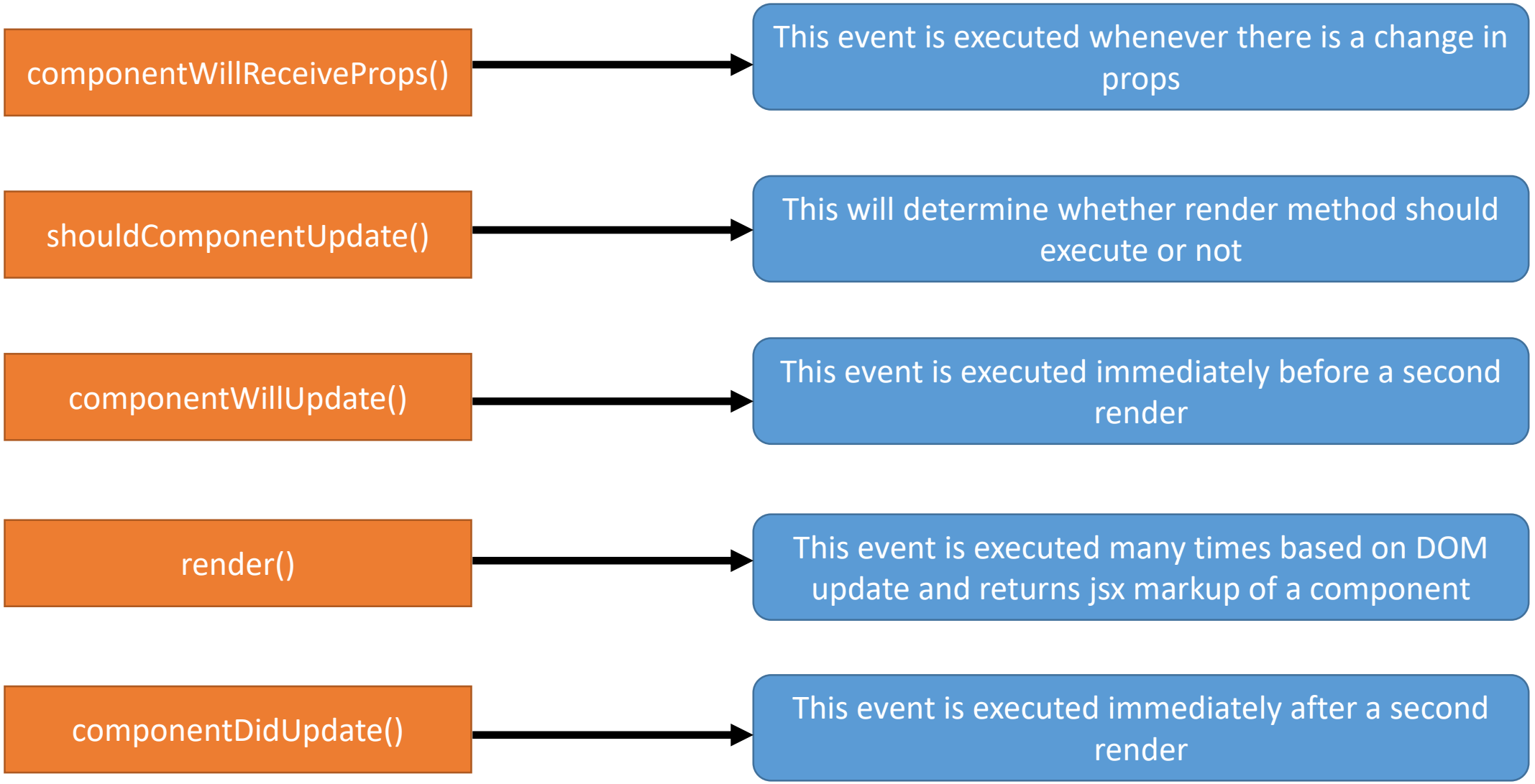


Component Life Cycle – Mounting/Initialization



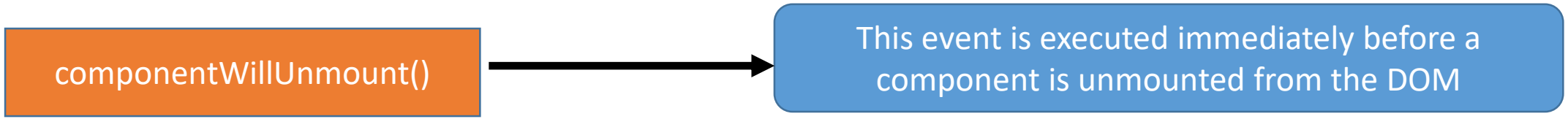
accenture

Component Life Cycle – Updating

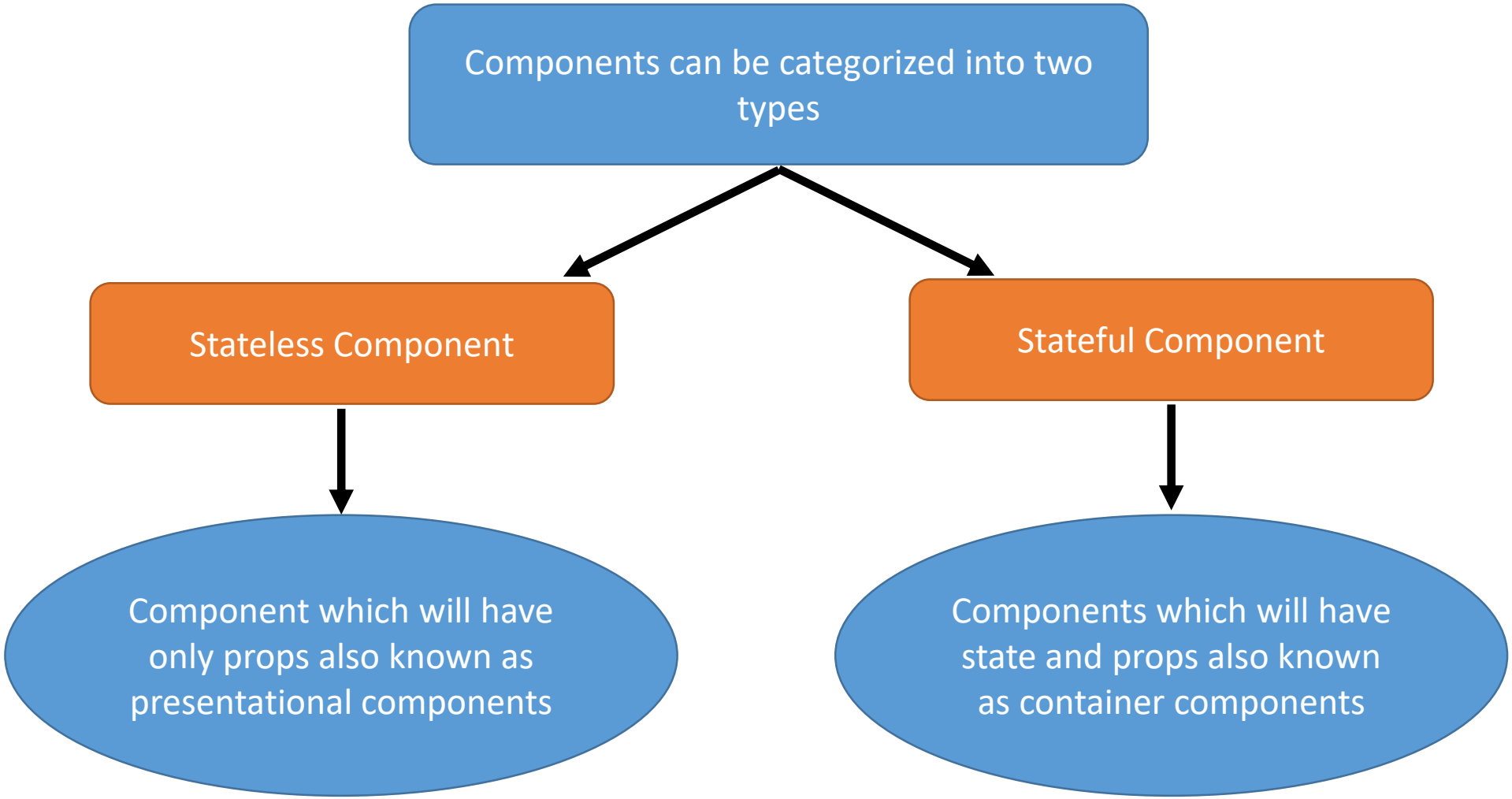


accenture

Component Life Cycle – Unmounting



Component Types

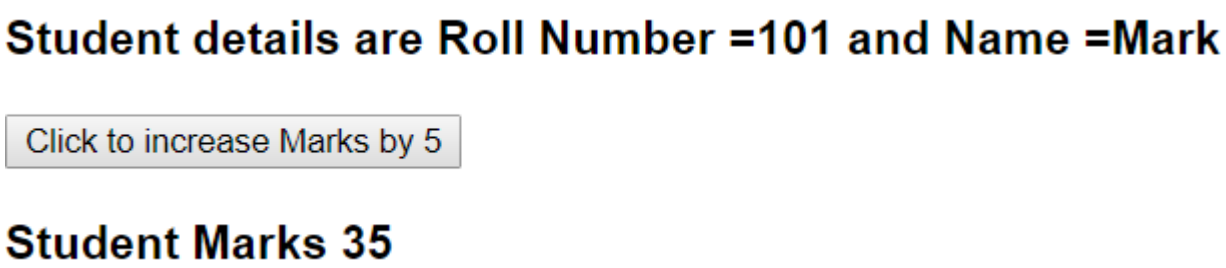


Activity

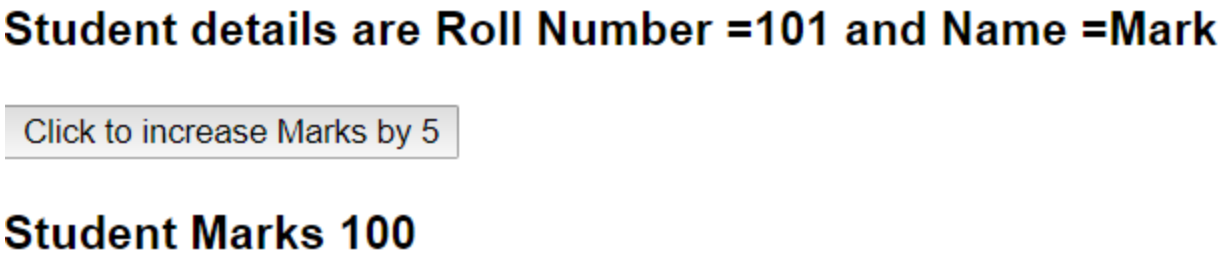
1. Create Student component
2. Display roll number and first name of a student using props of a component
3. Display marks of a student using state of a component
4. Create a button which should increase the marks of a student by 5
5. Stop the increament of a marks, once the marks reach to 100 and also show the alert with the message “student marks cannot be more than 100”

Expected Activity output

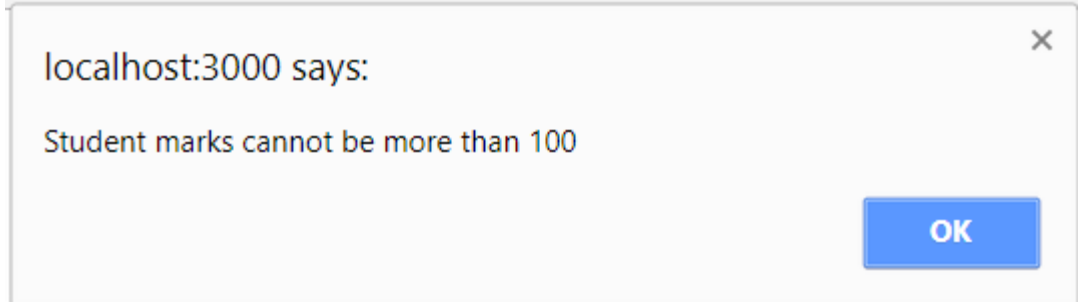
Initial view



After multiple clicks on increase marks button



Display error



MODULE SUMMARY

- What is component
- Role of component in react application
- How components created
- Props in component
- States in component

accenture

THANK YOU

accenture