strategy | consulting | digital | technology | operations

REACTJS – MODULE 4
CONDITIONAL RENDERING
LISTS AND KEY

accenture

AGENDA

- Understand conditional render
- How to achieve conditional rendering
- Lists
- Keys
- Demo

accenture

Introduction to Conditional Rendering

React provides the facility of displaying different layout/templates based on some conditions

It is also possible to render complete component based on the conditions

Conditions can be created using if-else statement

Introduction to Conditional Rendering

In case if the small portion of a component is required to render based on the condition, then it is always better to create variable and store the small portion of UI inside the variable

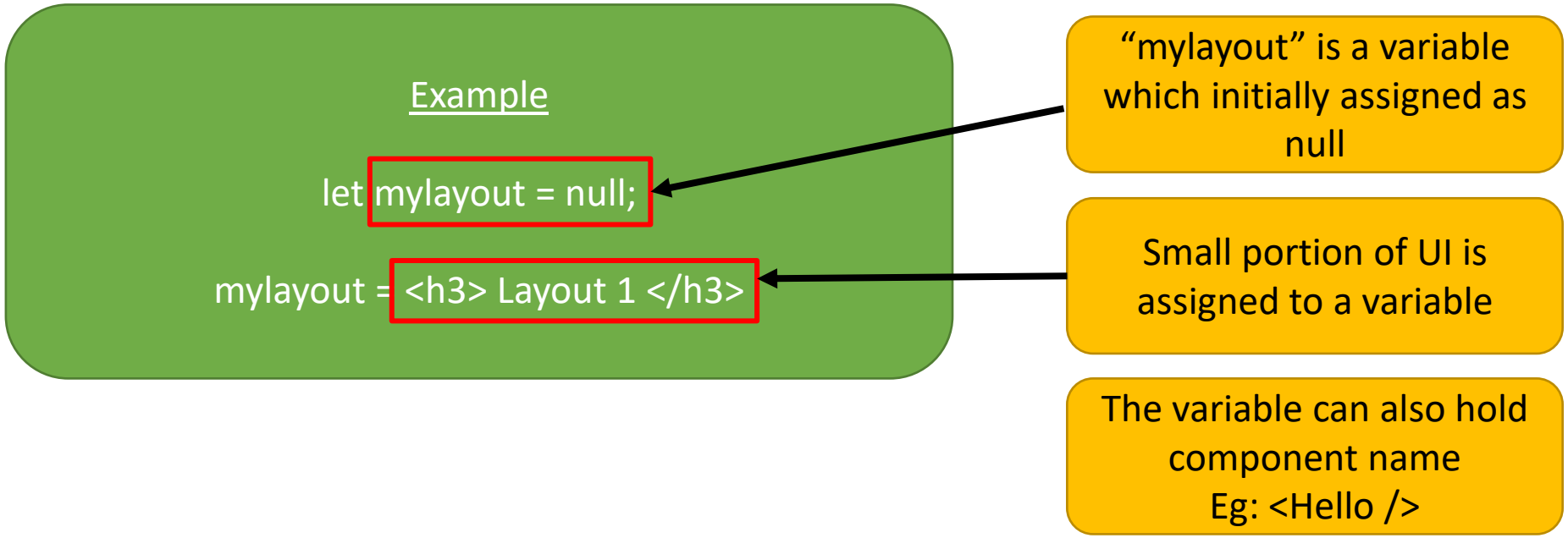
Example

let mylayout = null;

mylayout = <h3> Layout 1 </h3>

Introduction to Conditional Rendering

In case if the small portion of a component is required to render based on the condition, then it is always better to create variable and store the small portion of UI inside the variable



Demo – Conditional Render

Create a new component(Tryconditionalrender.js)

```
import React,{Component} from 'react';
class Tryconditionalrender extends Component{
    render(){
        let val=1;
        let mylayout=null;
        if(val===1){
            mylayout= <div style={{'color':'red','backgroundColor':'lightgreen','height':'50%',
                                   'width':'50%'}}>
                            <h3>Layout 1</h3>
                      </div>
        }
        else{
            mylayout=<div style={{'color':'red','backgroundColor':'yellow','height':'50%',
                                  'width':'50%'}}>
                            <h3>Layout 2</h3>

                      </div>
        }
        return(
            <div>
                <h3>Conditional Render</h3>
                {mylayout}
            </div>
        )
    }
}

export default Tryconditionalrender;
```

Demo – Conditional Render

Create a new component(Tryconditionalrender.js)

```
import React,{Component} from 'react';
class Tryconditionalrender extends Component
    render(){
        let val=1;
        let mylayout=null;
        if(val===1){
            mylayout= <div style={{'color':'
                                    'width':'
                            <h3>Layout 1</h3>
                        </div>
        }
        else{
            mylayout=<div style={{'color':'red','backgroundColor':'yellow','height':'50%',
                                  'width':'50%'}}>
                            <h3>Layout 2</h3>

                        </div>
        }
```

,'height':'50%',

"val" is a variable assigned with the value=1

"mylayout" is a variable assigned with the null value intially

```
        return(
            <div>
                <h3>Conditional Render</h3>
                {mylayout}
            </div>
        )
    }
}

export default Tryconditionalrender;
```

accenture

Demo – Conditional Render

Create a new component(Tryconditionalrender.js)

```
import React,{Component} from 'react';
class Tryconditionalrender extends Component{
    render(){
        let val=1;
        let mylayout=null;
        if(val===1){
            mylayout= <div style={{'color':
                                   'width':
                            <h3>Layout 1</h3>
                        </div>
        }
        else{
            mylayout=<div style={{'color':'red','backgroundColor':'yellow','height':'50%',
                                  'width':'50%'}}>
                            <h3>Layout 2</h3>

                        </div>
        }
```

Compare the variable value with equal to 1. if this is true then mylayout variable will have some view to display

```
        return(
            <div>
                <h3>Conditional Render</h3>
                {mylayout}
            </div>
        )
    }
}

export default Tryconditionalrender;
```

accenture

Demo – Conditional Render

Create a new component(Tryconditionalrender.js)

```
import React,{Component} from 'react';
class Tryconditionalrender extends Component{
    render(){
        let val=1;
        let mylayout=null;
        if(val===1){
            mylayout= <div style={{'color':'red','backgroundColor':'lightgreen','height':'50%',
                                   'width':'50%'}}>
                          <h3>Layout 1</h3>
                      </div>
        }
        else{
            mylayout=<div style={{'color':'red','backgroundColor':'yellow','height':'50%',
                                  'width':'50%'}}>
                         <h3>Layout 2</h3>

                     </div>
        }
```

```
            retu
                <
                <
            )
        }
}

export default Tryconditionalrender;
```

This is the view/UI which is assigned to the variable “mylayout” and displayed when the if block is executed

This is the view/UI which is assigned to the variable “mylayout” and displayed when the else block is executed

accenture

Demo – Conditional Render

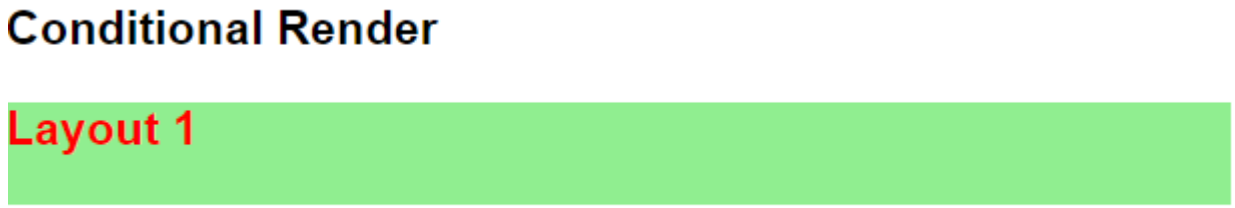
Create a new component(Tryconditionalrender.js)

```
import React,{Component} from 'react';
class Tryconditionalrender extends Component{
    render(){
        let val=1;
        let mylayout=null;
        if(val===1){
            mylayout= <div style={{'color':'red','backgroundColor':'lightgreen','height':'50%',
                                   'width':'50%'}}>
                            <h3>Layout 1</h3>
                      </div>
        }
        else{
            mylayout=<div style={{'color':'red','backgroundColor':'yellow','height':'50%',
                                  'width':'50%'}}>
                            <h3>Layout 2</h3>

                      </div>
        }
        return(
            <div>
                <h3>Conditional Render</h3>
                {mylayout}
            </div>
        )
    }
}

export default Tryconditionalrender;
```

This will have final view to display based on the condition

Demo – Conditional Render

Output



Lists and Keys

Lists are the way to display items one by one

In react list plays very important role as it provides the facility of displaying items very easily

Demo – Display Array

Create a new component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>


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

Demo – Display Array

Create a new component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>


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

Created array of numbers and stored in a variable "array1"

Using unordered list the complete array items are displayed

Demo – Display Array

Create a new component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>


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

Output

List Demo

12345

Demo – Display Array using List

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    arrayitem=this.array1.map((a)=>
        <li>{a}</li>
    );


    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

accenture

Demo – Display Array using List

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    arrayitem=this.array1.map((a)=>
        <li>{a}</li>
    );


    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

Created array of numbers and stored in a variable “array1”

“arrayitem” is a variable which will hold each element from array1 and rendered in the format of <li> , using map method

Each item from “arrayitem” is displayed using unordered list

accenture

Demo – Display Array using List

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    arrayitem=this.array1.map((a)=>
        <li>{a}</li>
    );


    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

Output

List Demo

12345

Display array using list format

- 1
- 2
- 3
- 4
- 5

accenture

Demo – Display Array using List

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,5];

    arrayitem=this.array1.map((a)=>
        <li>{a}</li>
    );


    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

When this code is executed this will display output, but on browser's console it will display error

```
▶Warning: Each child in an array or index.js:2177
iterator should have a unique "key" prop.

Check the render method of `Listdemo`. See http
s://fb.me/react-warning-keys for more information.
    in li (at Listdemo.js:6)
    in Listdemo (at App.js:17)
    in div (at App.js:11)
    in App (at index.js:7)
```

accenture

Lists and Keys

Keys plays an important role in react with respect to list element

This gives the facility to identify or track each item in the list

This is also useful to find which item in the list is added, modified or deleted

accenture

Demo – Display Array using List with Key

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,4,5];

    arrayitem=this.array1.map((a,id)=>
        <li key={id}>{id} - {a}</li>
    );
    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

accenture

Demo – Display Array using List with Key

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,4,5];

    arrayitem=this.array1.map((a,id)=>
        <li key={id}>{id} - {a}</li>
    );
    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

Created array of numbers and stored in a variable “array1”

id is added here to identify each element uniquely

Id is assigned to key, even though there are repeated items are present in the array, the key will be different for all the items. This way key will help to identify each item uniquely

accenture

Demo – Display Array using List with Key

Modify a existing component(Listdemo.js)

```
import React,{Component} from 'react';
class Listdemo extends Component{
    array1=[1,2,3,4,4,5];

    arrayitem=this.array1.map((a,id)=>
        <li key={id}>{id} - {a}</li>
    );
    render(){
        return(
            <div>
                <h3>List Demo</h3>
                <ul>{this.array1}</ul>
                <h4>Display array using list format</h4>
                <ul>{this.arrayitem}</ul>
            </div>
        )
    }
}
export default Listdemo;
```

Output

List Demo

123445

Display array using list format

- 0 - 1
- 1 - 2
- 2 - 3
- 3 - 4
- 4 - 4
- 5 - 5

accenture

Activity

1. Create Employee component

2. Declare employee array in the parent component

```
[ { id : 101, name : 'Jaya', skills : ['Angular', 'Node']},
  { id : 102, name : 'Shyam', skills : ['React', 'Node']},
  { id : 103, name : 'Jyothi', skills : ['MongoDB', 'Node']}, …
]
```

1. Pass this names array from parent component to Employee component

2. Capture the names array in Employee component and display each item as employee names

MODULE SUMMARY

- Understand how conditional rendering is implemented
- Setting styles in react
- Working with lists
- Using map
- Working with key

accenture

THANK YOU

accenture