**Highlights:**

* How to render component on the server side
* How to handle the rendered component at the client side

**Demo steps:**

Let's create a React component, render the same component on both sides.

1. Create Component.jsx as shown below to create **Component**:

1. var React = require('react')
2. module.exports = class AppComp extends React.Component {
3. handleClick(){
4. alert("Clicked");
5. }
6. render(){
7. return (
8. <html>
9. <head>
10. <title>Server Side Rendering</title>
11. </head>
12. <body>
13. <div>
14. <h1>React server side rendering</h1>
15. <button onClick={this.handleClick}> Click </button>
16. </div>
17. <script src="/bundle.js" />
18. </body>
19. </html>
20. );
21. }
22. }

2. Create main.js as shown below for rendering component on the client side:

1. import React from 'react';
2. import ReactDOM from 'react-dom';
3. import App from './App';
4. ReactDOM.render(<App/>, document);

3. Create an express server as shown below which will accept the request on port 3000 and returns the response which is a React component in HTML string.

server.js

1. require('babel-register')({
2. presets: ['react']
3. });
4. var express = require ('express'); *//importing*
5. var app= express(); *//*
6. var React = require ('react');
7. var App = require('./App'); *//comp*
8. var ReactDOMServer = require('react-dom/server'); *//rendering comp on server*
9. app.use(express.static('public')); *// assets folder generaTING bundle.js*
10. app.get('/', function(request, response) { *//function will be executed when the path is /*
11. var html = ReactDOMServer.renderToString( *//render html string*
12. React.createElement(App) *// // this will be converted to html string*
13. );
14. response.send(html); *//html string will sent to client*
15. });
16. var PORT =3000; *//mentioning the port*
17. app.listen(PORT, function() { *//listen() method on application object to specify the port number on which the application listens for requests.*
18. console.log('http://localhost' +PORT);
19. });

Steps to run the demo:

a. Download this demo from the link - [Server-Side Rendering](https://academy.onwingspan.com/common-content-store/Shared/Shared/Public/lex_7787340008098592000_shared/web-hosted/assets/ServerSideRendering.zip)

b. Navigate to the ServerSideRendering folder and install the required packages by running the command **npm install**

c. Command to be used to run the demo: npm start

d. Output can be seen in port 3000

The npm start command will run the webpack command first, the webpack command will create the bundle file bundle.js within the public folder. The bundle.js file can also be sent from the server as the part of the HTML which server sends

The bundle.js file contains the JavaScript code (JSX and ES6 converted) which browser can understand and it is included to the HTML file using script tag.

1. <script src="/bundle.js" />

Observe the below points in the given demo:

a. When JavaScript is disabled at the browser side, we cannot handle click event, it just renders the content from the server

b. To handle the click event, JavaScript has to be enabled at the browser side