**Step 1:**

import logo from './logo.svg';

import './App.css';

const App = () => {

window.navigator.geolocation.getCurrentPosition((position) => console.log(position),

(error) => console.log(error));

return (

<div className="App">

</div>

);

}

export default App;

**Timeline of Application inside the browser**

**1. js file loaded up by the browser**

**2. App component gets created**

**3. We call the geolocation service**

**4. App gets rendered to page as HTML**

**5. We get result of geolocation**

**Follow 3 steps for class component**

**1. must be javascript class**

**2. must extends React.Component**

**3. must define render method**

**Step 2: transformation from function to class component**

**import React from 'react'**

**// transformation from functional component to class component**

class App extends React.Component{

render(){

return(

<div>

You are in the northeren hemisphere.

</div>

)

}

}

export default App;

**Step 3: Using State**

**1. State : its only usable with class component**

**2. State is a js object that contains data relevant to the component**

**3. Updating the state on a component cause the componnent re-render**

**4. State must be initialized when a component is created**

**5. using 'setState' method state can be updates**

**import React from 'react'**

**// transformation from functional component to class component**

class App extends React.Component{

constructor(props) {

super(props);

this.state = {latitude: 25}

}

render(){

window.navigator.geolocation.getCurrentPosition((position) => console.log(position),

(error) => console.log(error));

return(

<div>

{this.state.latitude}

</div>

)

}

}

**Step 4: adding setState**

import React from 'react'

// transformation from functional component to class component

class App extends React.Component{

constructor(props) {

super(props);

this.state = {latitude: 25, error: null}

window.navigator.geolocation.getCurrentPosition(

(position) => {

this.setState({latitude: position.coords.latitude}) },

(error) => {

console.log(error)

;

})

}

render(){

return(

<div>

{this.state.latitude}

</div>

)

}

}

**Step 4:**

**Called ComponentDidMount Method of component mount phase**

**1. Js file loaded up by browser**

**2. App component get created**

**3. Constructure function gets called**

**4. this.state property assigned the state object**

**5. We call geolocation service**

**6. React calls the component render method**

**7. App returns jsx , get rendered to page as HTML**

import React from 'react'

// transformation from functional component to class component

class App extends React.Component{

constructor(props) {

super(props);

this.state = {latitude: 25, error: null}

}

componentDidMount()

{

window.navigator.geolocation.getCurrentPosition(

(position) => {

this.setState({latitude: position.coords.latitude}) },

(error) => {

console.log(error)

;

})

}

render(){

return(

<div>

{this.state.latitude}

</div>

)

}

}

export default App;

**Step 5: Removed the Construtor Function**

import React from 'react'

// transformation from functional component to class component

class App extends React.Component{

constructor(props) {

super(props);

this.state = {latitude: 25, error: null}

}

componentDidMount()

{

window.navigator.geolocation.getCurrentPosition(

(position) => {

this.setState({latitude: position.coords.latitude}) },

(error) => {

console.log(error)

;

})

}

render(){

return(

<div>

{this.state.latitude}

</div>

)

}

}

export default App;

**Step 6:**

import logo from './logo.svg';

import './App.css';

import React from 'react'

// transformation from functional component to class component

class App extends React.Component{

state = {latitude: null, errorMessage: ''}

componentDidMount()

{

window.navigator.geolocation.getCurrentPosition(

(position) => {

this.setState({latitude: position.coords.latitude}) },

(error) => {

this.setState({errorMessage: error.message})

;

})

}

render(){

return(

<div>

{this.state.latitude}

{this.state.errorMessage}

</div>

)

}

}

export default App;

**Step 7: Conditional rendering**

render(){

if(this.state.errorMessage && !this.state.latitude){

return <div>{this.state.errorMessage}</div>

}

if(this.state.latitude && !this.state.errorMessage){

return <div>{this.state.latitude}</div>

}

else{

return <div>Loading....</div>

}

}

}

**Step 8:Passing state as props**

**App.js**

render(){

if(this.state.errorMessage && !this.state.latitude){

return <div>{this.state.errorMessage}</div>

}

if(this.state.latitude && !this.state.errorMessage){

return <div><HemisphereDisplay latitude={this.state.latitude}/></div>

}

else{

return <div>Loading....</div>

}

}

}

**HemisphereDisplay.js**

import React from 'react';

import south from './south.jpg';

import north from './north.jpg'

const HemisphereDisplay = ({latitude}) => {

console.log(latitude);

const display = latitude > 0 ? 'Northeren Hemisphere' : 'Southeren Hemisphere';

const picture = latitude > 0 ? north : south;

return(

<div>

{display}

<img src={picture} alt="location" />

</div>

)

}

export default HemisphereDisplay;