**The code is same as the previous code:**

In react we use camel case for **onClick,** and use {} for writing function declaration. and write function definition inside class component

 <button onClick={this.handleClockButtonClick}>click me EN Timer</button>

And event handler inside class

import React from "react";

class Clock extends React.Component {

  // others code

  // event handler for clock button

  handleClockButtonClick() {

    this.setState({ locale: "en-US" });

  }

  render() {

    return (

      <>

        <h1 className="heading">

          <span className="text">

            {this.state.date.toLocaleTimeString(this.props.locale)}

          </span>

        </h1>

        <button onClick={this.handleClockButtonClick}>click me EN Timer</button>

      </>

    );

  }

}

export default Clock;

Note: to prevent default behavior of the form submit button we need to take **event** parameter into event handler method. And use **event.preventDefault();**

**We have a problem with the handleClockClicked method.** Below is the reason and solution

  // event handler for clock button

  handleClockButtonClick() {

    this.setState({ locale: "en-US" });

  }

This handler is another method which is not part of the class mainly. We write it into class but it comes from the button event handler. So button event handler have access class but inside event handler we have a method handleClockButtonClick() and has not any access of the class. So we have got undefined error. Bellow the two way to solve this problem.

**Arrow function:**

 // event handler for clock button

  handleClockButtonClick = () => {

    this.setState({ locale: "en-US" });

  };

**Bind() method with constructor**

Constructor

  // constructor

  constructor(props) {

    super(props);

    this.state = {

      date: new Date(),

    };

    this.handleClockButtonClick = this.handleClockButtonClick.bind(this);

  }

And function

  // event handler for clock button

  handleClockButtonClick() {

    this.setState({ locale: "en-US" });

  }

Another way: not need to use constructor, direct bind() inside event handler

  <button onClick={this.handleClockButtonClick.bind(this, parameter)}>click me EN Timer</button>

Another way to do this without constructor: use arrow function as return

 <button onClick={() => this.handleClockButtonClick("en-US")}>

          click me EN Timer

        </button>

  // event handler for clock button

  handleClockButtonClick(parameter) {

    this.setState({ locale: "en-US" });

  }

**This keywords problem and solution:** two ways

**Solution one:** **use bind()** method and pass **this.** inside forEach() method we have a function and inside this function we use this. but this function has not any context of this. because this is another function. This is callback and it has not any name and other properties. This is single and separate callback function. So we can pass **this inside bind(this)** method.

**Solution two: use arrow function inside forEach() method**

Arrow function have not any headache with this keywords.

**Recommended:** use arrow function and inside event handler use this.handlerClick

**Control re-render:**

When a parent component is re-render, then corresponding child component will be re-rendered. This is simple rule of react. Jodi amon hoi j parent state change hole child hudai re-render nah hok. Amra ata off korte pari.

shouldComponentUpdate() takes parameter **nextProps.** and compare with previous and current and return appropriate value. If there has no changed then return false.