**CONTEXT API**

Ilk once Context API imizi olusturuyoruz

import React from "react";

const authContext = React.createContext({

  authenticated: false,

  login: () => {}

});

export default authContext;

Sonra AuthContext ‘imizi asagida ki gibi import ediyoruz.

Daha sonra authenticated: false olusturuyoruz state miz icinde

Daha sonra loginHandler olusturuyoruz bu state imizi degistiren

Ve daha sonra import ettigimiz AuthContext imizi diger hangi Componentlardan ulasilmasini istiyorsak onlarin uzerinde yani onlari icine alarak tanimliyoruz asagidaki gibi.

Ve bu Provider imiz icine value olarak bir obje icinde,

authenticated icin state imizde tuttugumuz degeri,

login olarakda state mizi tetikleyen fonksiyonumuzu atiyoruz.

Authenticated ve login olusturdugumuz Context .js icinde belirledigimiz degerler.

Ve artik Persons ve Cockpit componentlarimiz icinden Context imizi manipule edebiliriz.

import React, { Component } from "react";

…

import AuthContext from "../context/auth-context";

class App extends Component {

  state = {

    …

    authenticated: false

  };

  …

  loginHandler = () => {

    this.setState({ authenticated: true });

  };

  …

  render() {

    return (

      <WithClass classes={classes.App}>

        …

        <AuthContext.Provider

          value={{

            authenticated: this.state.authenticated,

            login: this.loginHandler

        }}>

          <Persons …/>  
 <Cockpit …/>  
        </AuthContext.Provider>  
      </WithClass> );}}  
export default App;

Persons Context Manipule

Context’ten veri okumak icinde olusturdugumuz Context’i once import ederiz.

Sonra okumak istedigimiz yerde asagidaki gibi cagiririz.

Ve icinde curly bracer lar icinde fonksiyon kurariz. Ben context dedim siz istediginizi diyebilirsiniz bu sekilde argumenti context olup return alan bir fonksiyon olusturuyoruz. Bu context argument i context verilerimize erisiyor.

Ve asagida context e erisip authenticated eger dogruysa Authenticated yazdir degilse Please login yazdir demisim.

import React, { Component } from "react";

import AuthContext from "../../context/auth-context";

…

  render() {

    return (

      <Auxilliary>

        <AuthContext.Consumer>

          {context =>

            context.authenticated ? <p>Authenticated!</p> : <p>Please Login</p>

          }

        </AuthContext.Consumer>

        …

      </Auxilliary> );}}

…

export default Person;

Cockpit Context Manipule

Context’imizi yine import ediyoruz.

Sonra yine bu contextimize erisip butonun icinde contextimizdeki login fonksiyonunu onClick yapinca erisimini sagliyoruz.

import React, { useEffect, useRef } from "react";

import AuthContext from "../../context/auth-context";

const cockpit = props => {

  …

  return (

    <div className={classes.Cockpit}>

      …

      <AuthContext.Consumer>

        {context => <button onClick={context.login}>Log in</button>}

      </AuthContext.Consumer>

    </div> )}

export default React.memo(cockpit);

**static contentType**

static contextType = AuthContext; yaptiktan sonra artik this.context ile asagidaki gibi contextimize erisebiliriz.

Eristikten sonra eskisi gibi AutContext.Consumer yazip fonksiyonla contextimize erismek zorunda kalmiyoruz.

import React, { Component } from "react";

…

import AuthContext from "../../../context/auth-context";

class Person extends Component {

  constructor(props) {…}

  static contextType = AuthContext;

  componentDidMount() {

    this.inputElementRef.current.focus();

    console.log(this.context.authenticated);

  }

  render() {

    return (

      <Auxilliary>

//ESKISI

        //<AuthContext.Consumer>

        //  {context =>

        //    context.authenticated ? <p>Authenticated!</p> : <p>Please Login</p>

        //  }

        //</AuthContext.Consumer>

//YENISI

        {

          this.context.authenticated ? <p>Authenticated!</p> : <p>Please Login</p>

        }

        …

      </Auxilliary>

    );

  }

}

export default Person;

**useContext()**

Fuctional componentlarda ise static contextType kullanamayacagimiz icin Hooks yontemi useContext ile isimizi goruyoruz.

Once React icinde {} lar icinde import ediyoruz.

Daha sonra fonksiyon icinde const authContext degiskeni olusturup useContext() icinde import ettigimiz context dosyamizi yaziyoruz. Asagidaki gibi

Ve artik olusturdugumuz authContext degiskeni ile kisa yoldan contexte erisebiliriz. Ve butonda kullasnabiliriz asagidaki gibi.

Eskisi ve yenisi arasindaki farki gorebilirsiniz. useContext kullanmayip eskisi gibide erisebiliriz.

import React, { useEffect, useRef, useContext } from "react";

import AuthContext from "../../context/auth-context";

const cockpit = props => {

  const authContext = useContext(AuthContext);

  …

  return (

    <div className={classes.Cockpit}>

      …

//ESKISI

<AuthContext.Consumer>

        {context => <button onClick={context.login}>Log in</button>}

      </AuthContext.Consumer>

//YENISI

      <button onClick={authContext.login}>Log in</button>

    </div>

  );

};

export default React.memo(cockpit);