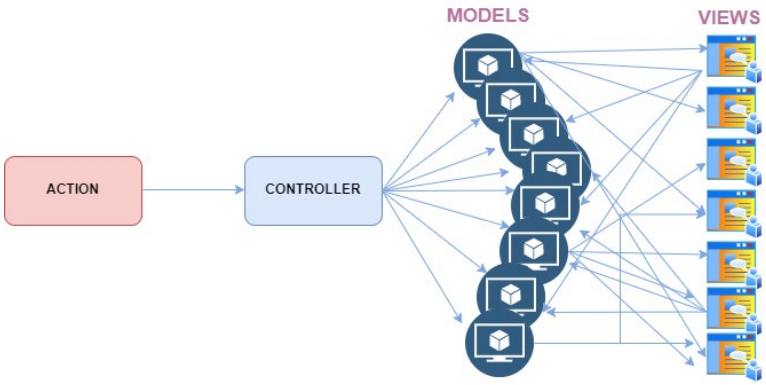
REDUX KULLANIMI

- ## Redux Ortaya Çıkartan Sorun

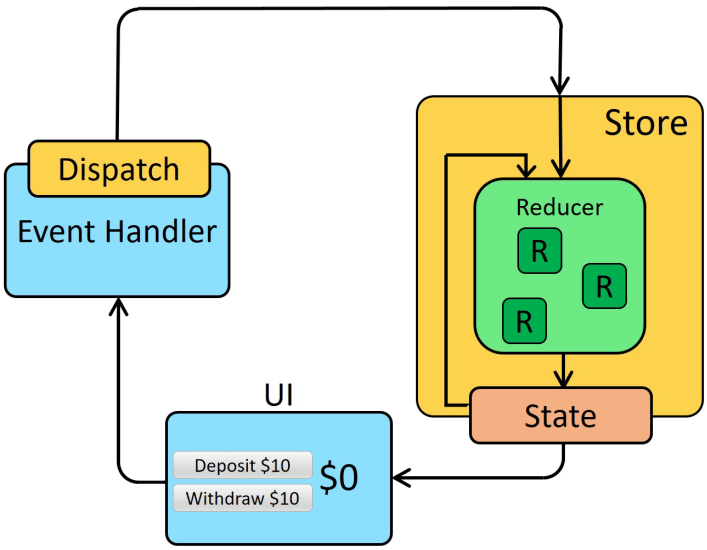
React gibi bolca component kullanan yapılarda ve aynı zamanda birden çok modelin etkilediği ve view lar tarafından etkilenen modellerin oluşturduğu yapılarda güncellemeler belli noktadan sonra büyük sorunlar çıkartmaya başlamakta idi. Facebook bu sorunsalla karşılaştığında bir çok güncelleme ile bunu gidermeye çalıştı ancak başarılı olmadı. Bu nedenle Redux dediğimiz javascript kütüphanesini oluşturdu.



Redux Sorunsalına giden yol 😊

- ## Redux Arhitecture

Redux, state i güncelleyen bir reducer ya da reducer ler bütünü ile çalışır. Bu reducer lar bir action tarafından tetiklenir. Değişiklik ilgili view ya da view lara iletilir.



- Redux' ı Kodlamak

1- Herşeyi derli tutmak adına "redux" adında bir klasör ve içinde "action" ve "reducer" adında iki klasör daha açılır.

2- "action" klasörü içinde "index.js" adında bir file oluşturulur. Bu file içinde reducer bileşenlerinde değişikliğe gitmek üzere function lar oluşturulur. Bir action method u şuna benzer.

```
export const login= ()=>{
    return{
      type: "SIGN_IN"
    }
  }
```

3- "reducer" klasörü içinde amacına uygun olarak file oluşturulur(Örn: "login.js"). Bu dosya içindeki method(reducer), action üzerinden gelebilecek istekleri yakalayacak bir action ve değişikliğe uğradığında kullanılacak bir state bulundurur. Reducer action dan aldığı parametreye göre güncelleme yapar.

```
const login= (state,action)=>{
    switch(action.type){
      case 'SIGN_IN': return true
      default: return false
    }
}
export default login
```

4- Eğer birden çok reducer var ise bunları bir dosya üzerinde tekilleştirmek kullanım kolaylığı açısından iyi olacaktır. Bunu yapmak için reducer kalsörü içinde "index.js" adında bir file açın ve içine reducer larınızı import edin. Ardından bu reducer ları combine edecek redux method unu ekleyin. Ardından, bu methodu kullanarak tüm reducer larınızı bu method içine ekleyin ve bir isim verin. Son olarak bu method u bir const a eşitleyerek export edin. Bir combin şuna benzer:

```
import { combineReducers } from 'redux'
import Islem from './islem'
import Login from './login'
const AllReducers= combineReducers({
        islem: Islem,
        login: Login
    })
export default AllReducers
```

5- Artık tüm action ve reducer yapımız oluşturuldu. Buradan sonra bu bileşenlerimizin store a çevirmeli ve uygulamamızın içine enjekte etmeliyiz. Bunun için bir takım adımları izleyelim.

- *Tüm reducer ları içine eklediğimiz file ı import ediyoruz.*

```
import AllReducers from './redux/reducers'
```

- *Reducer ları store a çevirmek için redux methodu eklenir.*

```
import { createStore } from 'redux';
```

- *Store bileşenini tüm uygulamaya yaymak üzere Provider bileşeni eklenir.*

```
import {Provider} from 'react-redux'
```

- *Artık store oluşturmak üzere ilgili kodumuz yazılır. Tüm reducer lardan bir store oluştur.*

```
const store = createStore(
 AllReducers, window.__REDUX_DEVTOOLS_EXTENSION__ &&
 window.__REDUX_DEVTOOLS_EXTENSION__());
```

- *Sonrada tüm sisteme enjekte edilmek üzere store Provider ile eklenir.*

```
  <Provider store={store}>
    <App />
  </Provider>
```

6- Şimdi de sistemimize aktardığımız state bilgilerini istediğimiz yerden okumayı deneyelim.

- *NOT: bir sınıf içinden verileri okumak ile bir function içinden bilgileri okumak aynı değildir. Buna göre işlem yapılır.*
- *Function içinden okumak,*

İlk olarak redux içinde bulunan useSelector import edilir.

```
   import {useSelector} from 'react-redux'
```

sonra değerleri okumak için, selector ile BURAYA DİKKAT 4. Maddede combine ile oluşturduğunuz ve isim verdiğiniz state değişkeni ile çağırılır.

```
 let islogin  = useSelector(x=> x.login)
```

- *Class içinden okumak,*

İlk olarak redux bileşenlerine bağlanmak ve state içindeki bilgileri aktarmak için connect nesnesi import edilir.

```
import {connect} from 'react-redux'
```

redux içinden gelecek olan state bilgilerini class içine aktarmak için, connect nesnesi ile sınıf sarılır ve bir method yazılarak bu aktarım bilgisi connect e verilir.

```
const mapStateToProps = (state) =>
({
    loginstate: state.login
})
export default connect(mapStateToProps)(Login)
```

burada dikkat etmeniz gereken şey, "state.login" store içinde barındırılan değeri temsil eder. 4. Maddede eklediğimiz login değişkenidir.

"loginstate" ise, store içinden alınan değerin connect edildiği sınıfın props' u içinde aktarılan değişken adıdır, ve şu şekilde erişilir;

```
const {loginstate} = this.props
```

eğer bu şekilde erişecekseniz. Bu satırı "render(){" alt satırına geçerek yazın.

Diğer türlü ise "this.props.loginstate" şeklinde direkt çekebilirsiniz.

7- Bir Store içindeki değeri değiştirme üzere action göndermek

Verileri okumak için kullandığımız yöntemi bu seferde action göndermek ve değerleri güncellemek üzere yapacağız. İlk olarak connect bileşenini eklemedi iseniz import ile ekleyin.

```
import {connect} from 'react-redux'
```

redux içindeki reducer leri tetiklemek üzere "dispatch" yapmalıyız. Bunu yapabilmek için yine bir method tanımlıyoruz. Bu method bir action 'ı class içindeki props içine aktarıyor. Böylece biz props içinden bu method u tetikleyebiliyoruz. Tetiklediğimiz bu method da reducer lar tarafından yakalanmak üzere bir action gönderiyor ve işlemimiz tamamlanıyor.

```
const mapDispatchToProps = (dispatch) => {
    return {
      // dispatching plain actions
      dologin: () => dispatch({ type: 'SIGN_IN' }),

    }
  }
// Eğer aynı anda veriye erişecek ve action göndericekseniz bunu kullanın
export default connect(mapStateToProps,mapDispatchToProps)(Login)
// Eğer sadece action göndereceksenız bunu kullanını
export default connect(null,mapDispatchToProps)(Login)
```

yukarıda gördüğünüz, kodlamadan sonra artık "dologin" method u Login sınıfının içine props a gönderildi. Böylece istediğimiz gibi tetikleyebiliriz.

"this.props.dologin()" şeklinde çağırılır.