Rakamin Academy

React Native

Redux Middleware

Raih Mimpi #TanpaBatas

Outline Pembelajaran

Rakamin Academy

React Middleware

- [ ] Redux Thunk Middleware

Objektif sesi

- Memahami dan mengimplementasikan Redux Thunk Middleware

Redux Thunk

Redux Thunk adalah middleware untuk Redux yang memungkinkan penanganan aksi (actions) di Redux untuk menjadi fungsi asinkron.

Ini sangat berguna ketika Anda perlu melakukan operasi asinkron, seperti pengambilan data dari server, sebelum mengirimkan aksi ke Redux store.

Redux Thunk: Preparation

Langkah-langkah Persiapan dan Contoh Penggunaan Redux Thunk:

1. Instalasi Paket: Pastikan Anda telah menginstal Redux dan Redux Thunk di proyek Anda.

```
npm install redux react-redux
```

```
npm install redux-thunk
```

```
npm install axios
```

Redux Thunk: Preparation

Langkah-langkah Persiapan dan Contoh Penggunaan Redux Thunk:

2. Konfigurasi Redux Store: Buat store Redux dan terapkan middleware Redux Thunk.

```
// store.js
import { createStore, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import rootReducer from './reducers';

const store = createStore(rootReducer, applyMiddleware(thunk));

export default store;
```

Redux Thunk: Preparation

Langkah-langkah Persiapan dan Contoh Penggunaan Redux Thunk:

3. Pembuatan Action Creator dengan Redux Thunk: Buat aksi (action) yang melibatkan operasi asinkron menggunakan Redux Thunk.

```
// Action Creator dengan Redux Thunk
export const fetchDataAsync = () => {
  return (dispatch) => {
    dispatch(fetchData());

    axios.get('https://api.example.com/data')
      .then(response => {
        // Dispatch aksi sukses setelah mendapatkan data
        dispatch({ type: 'FETCH_DATA_SUCCESS', payload: response.data });
      })
      .catch(error => {
        // Dispatch aksi gagal jika terjadi kesalahan
        dispatch({ type: 'FETCH_DATA_FAILURE', payload: error.message });
      });
  };
};
```

Redux Thunk: Preparation

Langkah-langkah Persiapan dan Contoh Penggunaan Redux Thunk:

4. Penggunaan pada Component:

```
const MyComponent = () => {
  const dispatch = useDispatch();

  useEffect(() => {
    // Mengirim aksi asinkron saat komponen dipasang
    dispatch(fetchDataAsync());
  }, [dispatch]);

  return (
    <View>
      <Text>My Component</Text>
      <Button
        title="Fetch Data"
        onPress={() => dispatch(fetchDataAsync())}
      />
    </View>
  );
};

export default MyComponent;
```

Redux Thunk: Preparation

Langkah-langkah Persiapan dan Contoh Penggunaan Redux Thunk:

5. Penggunaan pada Component:
Dalam contoh ini, kita menggunakan **useSelector** untuk mengambil data dari Redux store, seperti data, **isLoading**, dan **error**.

Komponen ini akan menampilkan pesan yang sesuai berdasarkan status pengambilan data.

```
const AnotherComponent = () => {
  // Menggunakan useSelector untuk mendapatkan data dari Redux store
  const data = useSelector(state => state.data);
  const isLoading = useSelector(state => state.isLoading);
  const error = useSelector(state => state.error);

  return (
    <View>
      {isLoading ? (
        <Text>Loading...</Text>
      ) : error ? (
        <Text>Error: {error}</Text>
      ) : (
        <Text>Data: {data}</Text>
      )}
    </View>
  );
};

export default AnotherComponent;
```

Rakamin Academy

Referensi

Redux Thunk


Rakamin
Academy
Thank you
Raih Mimpi #TanpaBatas