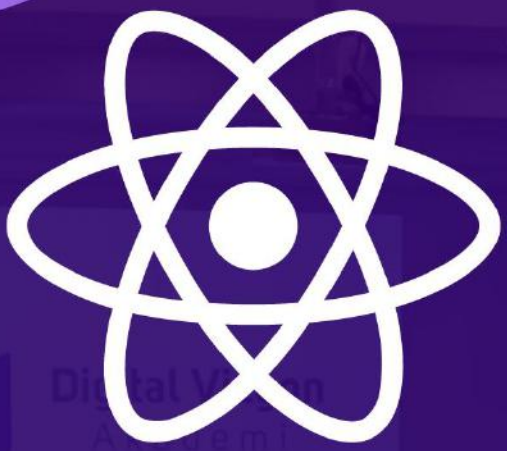

TYPESCRIPT KULLANARAK REACT'E GİRİŞ EĞİTİMİ

3 GÜN

Digital Vizyon Akademi

www.digitalvizyon.net

İçindekiler

Eğitim Hakkında

TypeScript Kullanarak React'e Giriş Eğitimi, React framework'üne TypeScript ile adım atmak isteyen herkes için mükemmel bir başlangıç noktasıdır. Bu eğitim, React temellerini TypeScript'in güçlü özellikleriyle birleştirerek, web uygulamaları geliştirmeye yönelik derinlemesine bir anlayış sağlar. Hem TypeScript'e yeni başlayanlar hem de deneyimli geliştiriciler için uygundur.

Eğitimin ilk bölümünde, TypeScript'in temellerini ve React ile nasıl entegre edileceğini ele alıyoruz. Katılımcılara, TypeScript'in statik tiplemesi, arayüzleri ve sınıfları gibi özelliklerinin, React bileşenlerinin daha sağlam ve ölçeklenebilir hale gelmesine nasıl yardımcı olduğu anlatılır. JSX ile TypeScript'in nasıl çalıştığını ve bileşenlerin nasıl oluşturulup birleştirildiğini gösteriyoruz.

Daha sonrasında, TypeScript ile React bileşenlerinin state ve props yönetimine odaklanırız. Katılımcılar, TypeScript'in tip güvenliği sayesinde, bileşenlere nasıl daha güvenilir şekilde state ve props verileri aktarabileceklerini ve bu verileri nasıl güncelleyebileceklerini öğrenirler.

TypeScript ile React uygulamalarını daha da güçlendirmek adına, TypeScript uyumlu araçlar ve kütüphaneler hakkında bilgi veriyoruz. Örneğin, React Router ve Redux gibi popüler kütüphaneleri TypeScript ile nasıl kullanacağınızı öğrenirsiniz. Bu, uygulamanın yönlendirme işlemlerini ve durum yönetimini kolaylaştırır.

Eğitim ayrıca pratik yapma fırsatı sunar; katılımcılar gerçek dünya projeleri üzerinde çalışırken TypeScript ve React ile ilgili öğrendikleri kavramları uygularlar. Bu, uygulama yapısı, bileşen entegrasyonu ve veri yönetimi gibi temel konuların daha iyi anlaşılmasına yardımcı olur.

Eğitim sonunda, geliştirilen projelerin nasıl test edileceği ve canlı ortama nasıl taşınacağı üzerinde duruyoruz. TypeScript ile geliştirilen React uygulamalarının performansını ve güvenliğini artırmak için en iyi uygulamaları sunuyoruz.

TypeScript ile React'e Giriş Eğitimi, katılımcılara TypeScript'in güçlü özelliklerini kullanarak React ile modern web uygulamaları geliştirmek için gereken bilgi ve becerileri kazandırır. Eğitim, katılımcılara hem teorik bilgi hem de pratik deneyim kazandırarak web geliştirme kariyerlerine sağlam bir temel atmalarını sağlar.

Neler Öğreneceksiniz?

TypeScript kullanarak React'e giriş eğitimi sırasında, aşağıdaki konuları öğrenebilirsiniz:

- TypeScript: Tip tanımlama sistemi, Interfaces, Generics, Namespaces ve diğer TypeScript özelliklerinin nasıl kullanılacağı.
- React: React component yapıları, state ve props, component lifecycle methods, event handling ve diğer React kavramları.
- TypeScript ile React: TypeScript kullanarak React componentlerinin nasıl yazılacağı, TypeScript ile React'in nasıl kullanılacağı ve bu ikilinin avantajları.

- Uygulama Geliştirme: TypeScript ile React kullanarak basit bir uygulamanın nasıl geliştirileceği, veri akışının nasıl yapılacağı ve componentlerin nasıl bir araya getirileceği.
- Debugging: TypeScript ile React kodlarında hata ayıklama teknikleri ve nasıl kullanılacağı.

Bu konuların öğrenilmesi, TypeScript kullanarak React geliştirme sürecinde başarılı bir şekilde ilerlemene yardımcı olacaktır.

Ön Koşullar

TypeScript kullanarak React'e giriş eğitimi için aşağıdaki ön koşulların yerine getirilmiş olması önerilir:

- JavaScript: TypeScript JavaScript dili üzerine kurulmuş bir dil olduğu için, JavaScript'in temel kavramlarının anlaşılmış olması gerekir.
- HTML ve CSS: React component yapılarının görsel olarak nasıl görüneceği konusunda fikir sahibi olmak için HTML ve CSS'in temel kavramlarının anlaşılmış olması gerekir.
- React: React kavramlarının anlaşılmış olması, component yapıları, state ve props gibi kavramlar hakkında bilgi sahibi olmak gerekir.

Bu ön koşulların yerine getirilmesi, TypeScript kullanarak React'e giriş eğitimini daha verimli ve kolay hale getirecektir. Eğer bu konular hakkında bilginiz yoksa, öncelikle bu konuları öğrenmeniz önerilir.

Kimler Katılmalı

TypeScript kullanarak React'e giriş eğitimi, aşağıdaki kişiler için uygun olabilir:

- Web Geliştiricileri: Web geliştirme deneyimi olan ve React kullanmak isteyen kişiler, TypeScript kullanarak React'e giriş eğitimi için uygun adaylardır.
- JavaScript Kullanıcıları: JavaScript'i iyi bilen ve ileri seviyede kullanmak isteyen kişiler, TypeScript kullanarak React'e giriş eğitimi için uygun adaylardır.
- Uygulama Geliştiricileri: Uygulama geliştirme deneyimi olan ve React ve TypeScript kullanmak isteyen kişiler, TypeScript kullanarak React'e giriş eğitimi için uygun adaylardır.
- Front-end Geliştiricileri: Front-end geliştirme deneyimi olan ve React ve TypeScript kullanmak isteyen kişiler, TypeScript kullanarak React'e giriş eğitimi için uygun adaylardır.

TypeScript kullanarak React'e giriş eğitimi, React ve TypeScript kullanmayı öğrenmek isteyen herkes için faydalı olabilir.

Outline

React Component Lifecycle

- Lifecycle overview
- Startup and mounting
- Rendering

- Updating
- Unmounting
- Using useEffect() for lifecycle methods
- Run once
- Run every render
- Run on specific changes / updates
- Lifecycle methods in tests
- Error handling and error boundaries
- Intermediate component usage
- Asynchronous dat
    - When should asynchronous fetching be done?
    - What challenges does async offer?
    - Working with Promises and generic types
    - Asynchronous best practices
    - Testing against async fetches
- Lists of data
    - Iterating over a list
    - The key property
    - Sorting data
    - Testing component interactions

Forms

- Controlled vs uncontrolled componentsorm field types
- What does React know about your form field?
- Does React control your form field?
- When does React find out about changes to your form field?
- Form field types
- Controlling a text field
- ther form fields
- Getting data out of a form
- Working with form data in test

Introduction to Reux

- WhWhat problems does Redux solve?
- How does it solve them?
- Basic Redux pattern
- Store
- Reducers
- Actions
- Redux types

Modern Redux with the Redux Toolkit

- What is the Redux toolkit
- What does it provide?
- The ducks pattern
- Testing Redux
- React and Redux
- Plugging into React
- State as props
- Events as dispatch
- Introducing higher-order components
- Types with React-Redux
- Too many variations
- Using Generics
- Solving TypeScript issues with React-Redux
- Turning our standalone Redux program into a component
- Middleware

Provided by the toolkit

- ther middleware
- Building a real-world React-Redux component
- Testing React-Redux components
- Higher-order components in detail
- What do higher-order components do?
- Why would I use a higher-order component?

Asynchronous Redux

- The difficulties of asynchronous Redux
- Asynchronous middleware
- Depending on your needs, we can use either thunks, sagas, or survey both techniques for asynchronous interactions
- Types as appropriate
- Dispatching async actions
- Typing async results
- Catching results
- Handling errors
- Testing asynchronous Redux

React Component Lifecycle

- Lifecycle overview

- Startup and mounting
- Rendering
- Updating
- Unmounting
- Using useEffect() for lifecycle methods
- Run once
- Run every render
- Run on specific changes / updates
- Lifecycle methods in tests
- Error handling and error boundaries
- Intermediate component usage
- Asynchronous dat
    - When should asynchronous fetching be done?
    - What challenges does async offer?
    - Working with Promises and generic types
    - Asynchronous best practices
    - Testing against async fetches
- Lists of data
    - Iterating over a list
    - The key property
    - Sorting data
    - Testing component interactions

Forms

- Controlled vs uncontrolled componentsorm field types
- What does React know about your form field?
- Does React control your form field?
- When does React find out about changes to your form field?
- Form field types
- Controlling a text field
- ther form fields
- Getting data out of a form
- Working with form data in test

Introduction to Reux

- WhWhat problems does Redux solve?
- How does it solve them?
- Basic Redux pattern
- Store
- Reducers

- Actions
- Redux types

Modern Redux with the Redux Toolkit

- What is the Redux toolkit
- What does it provide?
- The ducks pattern
- Testing Redux
- React and Redux
- Plugging into React
- State as props
- Events as dispatch
- Introducing higher-order components
- Types with React-Redux
- Too many variations
- Using Generics
- Solving TypeScript issues with React-Redux
- Turning our standalone Redux program into a component
- Middleware

Provided by the toolkit

- ther middleware
- Building a real-world React-Redux component
- Testing React-Redux components
- Higher-order components in detail
- What do higher-order components do?
- Why would I use a higher-order component?

Asynchronous Redux

- The difficulties of asynchronous Redux
- Asynchronous middleware
- Depending on your needs, we can use either thunks, sagas, or survey both techniques for asynchronous interactions
- Types as appropriate
- Dispatching async actions
- Typing async results
- Catching results
- Handling errors
- Testing asynchronous Redux