Events

React'te olay (event) sistemi -- onClick, event handler tanımlamak, onChange, onSubmit ve event object -- basit örneklerle.

Başlangıç 9 dk

Son güncelleme: 6 Ekim 2026
Çevrimiçi oku: https://www.learnforgex.com/tr/topics/events

Events

Component'ler ve Props kategorisinde component'leri nasıl birleştireceğimizi gördük. Şimdi arayüzü GERÇEKTEN etkileşimli hale getirmeye başlıyoruz -- kullanıcının tıklamalarına, yazdıklarına, form göndermelerine nasıl tepki vereceğimizi. Bu ders, React'in olay (event) sistemini anlatıyor.

Olay (Event) Nedir?

Bir olay (event), kullanıcının sayfada yaptığı bir eylemdir: bir butona tıklamak, bir input'a yazmak, bir formu göndermek gibi. React, bu eylemleri yakalayıp onlara tepki vermeni sağlayan hazır attribute'lar sunar -- `onClick`, `onChange`, `onSubmit` gibi.

onClick ile Tıklama Olaylarını Yakalamak

`onClick` attribute'una bir fonksiyon verirsen, o fonksiyon kullanıcı butona her tıkladığında çalışır:

```
function OnClickExample() {
  function handleClick() {
    console.log("Button clicked!");
  }

  return <button onClick={handleClick}>Click</button>;
}
```

`onClick={handleClick}` yazdık -- `handleClick()` değil. Bu fark önemli, bir sonraki bölümde bakıyoruz.

Event Handler Fonksiyonu Tanımlamak

Bir olaya tepki veren fonksiyona **event handler** denir. İki şekilde yazabilirsin: isimli bir fonksiyon olarak, ya da doğrudan satır içinde (inline):

```
function EventHandlerFunctionExample() {
  function sayHello() {
    console.log("Hello!");
  }

  return (
    <div>
      {/* CORRECT: we pass the function itself, we don't call it. */}
      <button onClick={sayHello}>Named Function</button>

      {/* You can also write an inline function. */}
      <button onClick={() => console.log("Hi!")}>Inline Function</button>

      {/* WRONG: if you write sayHello(), the function runs immediately
          as soon as the component renders -- it won't wait for a click. */}
      {/* <button onClick={sayHello()}>Wrong Usage</button> */}
    </div>
  );
}
```

En önemli kural: `onClick={sayHello}` yaz, `onClick={sayHello()}` **yazma**. İkincisini yazarsan, fonksiyon tıklamayı beklemeden, component render olur olmaz hemen çalışır -- çünkü `sayHello()` fonksiyonu ÇAĞIRIR ve sonucunu (`undefined`) `onClick`'e verir. `sayHello` (parantezsiz) ise fonksiyonun kendisini verir, React da onu senin yerine, doğru zamanda çağırır.

onChange ile Input Değişikliklerini Yakalamak

`onChange`, bir input'un değeri her değiştiğinde (kullanıcı her harf yazdığında) çalışır:

```
function OnChangeExample() {
  function handleChange(event) {
    console.log("Typed value:", event.target.value);
  }

  return <input type="text" onChange={handleChange} placeholder="Type something..." />;
}
```

Event handler'a otomatik olarak bir **event object** verilir -- `event.target.value` ile input'a o an ne yazıldığını okuyabilirsin.

onSubmit ile Form Gönderimini Yakalamak

`onSubmit`, bir `<form>` gönderildiğinde (kullanıcı Enter'a bastığında ya da submit butonuna tıkladığında) çalışır:

```
function OnSubmitExample() {
  function handleSubmit(event) {
    // Without preventDefault, the browser tries to submit the form and
    // the page reloads -- we don't want that in React apps.
    event.preventDefault();
    console.log("Form submitted!");
  }

  return (
    <form onSubmit={handleSubmit}>
      <button type="submit">Submit</button>
    </form>
  );
}
```

`event.preventDefault()` önemli bir satır: bu olmadan tarayıcı, formun "normal" davranışını sergiler ve sayfayı yeniden yükler. React uygulamalarında bunu neredeyse hiç istemeyiz -- sayfayı yeniden yüklemeden kendi kodumuzla ne olacağına karar vermek isteriz.

Event Object

Her event handler fonksiyonuna React tarafından otomatik olarak bir **event object** verilir. Bu nesne, olayla ilgili bilgiler taşır -- hangi tür olay olduğu, hangi elemanda gerçekleştiği gibi:

```
function EventObjectExample() {
  function handleClick(event) {
    console.log("Event type:", event.type);
    console.log("Target element:", event.target.tagName);
  }

  return <button onClick={handleClick}>See the Event Object (Check the Console)</button>;
}
```

`event.type` olayın türünü ("click", "change", "submit" gibi), `event.target` ise olayın gerçekleştiği DOM elemanını verir.

Özet ve Terimler Sözlüğü

React'te olaylara `onClick`, `onChange`, `onSubmit` gibi attribute'larla tepki verirsin. Event handler'a fonksiyonun kendisini ver, çağırma (`onClick={f}`, `onClick={f()}` değil). Her event handler'a otomatik olarak bir event object gelir; formlarda `event.preventDefault()` ile tarayıcının varsayılan davranışını (sayfa yenilemesini) engellersin.

Terimler Sözlüğü

Event (Olay) — Kullanıcının sayfada yaptığı bir eylem (tıklama, yazma, form gönderme gibi).

Event Handler — Bir olay gerçekleştiğinde çalışan fonksiyon.

Event Object — Bir olay gerçekleştiğinde event handler'a otomatik olarak verilen, olayla ilgili bilgiler taşıyan nesne.

`preventDefault()` — Tarayıcının bir olaya karşı normalde yapacağı şeyi (form gönderiminde sayfayı yenilemek gibi) engelleyen event object metodu.