Controlled Components

Controlled component nedir, value ve onChange ile bir input'u state'e bağlamak, checkbox/select'lerde de aynı deseni kullanmak -- basit örneklerle.

Orta 11 dk

Son güncelleme: 18 Ağustos 2026
Çevrimiçi oku: https://www.learnforgex.com/tr/topics/controlled-components

---

Controlled Components

Events dersinde `onChange`'i, State dersinde `useState`'i ayrı ayrı gördük. Bu ders, ikisini bir araya getiren ve React'te form yazmanın temel deseni olan **controlled component**'i anlatıyor.

Controlled Component Nedir?

Normalde bir `<input>`, kendi değerini kendi içinde (tarayıcının DOM'unda) tutar -- React bunun farkında bile olmayabilir. Bir **controlled component**'te ise input'un değeri React'in state'i tarafından BELİRLENİR: input, kendi başına bir değer tutmaz, değerini her zaman state'ten alır.

value ile Input'u State'e Bağlamak

Bir input'u controlled yapmanın ilk adımı, `value` attribute'unu bir state değişkenine bağlamak:

```
import { useState } from "react";

function ControlledInputExample() {
  const [text, setText] = useState("");

  function handleChange(event) {
    setText(event.target.value);
  }

  return (
    <div>
      {/* value={text} says that React's state should DECIDE the value the
          input displays on screen -- the input no longer holds its own
          value, it gets its value entirely from the `text` state. */}
      <input type="text" value={text} onChange={handleChange} />
      <p>You typed: {text}</p>
    </div>
  );
}
```

`value={text}` yazdığımız için, input'ta görünen değer HER ZAMAN `text` state'inin o anki değeri. Ama tek başına bu yeterli değil -- kullanıcı yazı yazdığında da bir şey olması lazım, bunu bir sonraki bölümde görüyoruz.

onChange ile State'i Güncellemek

`value` tek başına input'u salt-okunur yapar -- kullanıcı hiçbir şey yazamaz. State'i güncel tutmak için `onChange`'i de eklemen gerekir:

```
import { useState } from "react";

function WhyControlledMattersExample() {
  const [text, setText] = useState("");

  // Because the value lives in state, we can instantly use it in ANOTHER
  // part of the screen as the user types each letter -- for example to
  // show a character count or an uppercase preview.
  return (
    <div>
      <input
        type="text"
        value={text}
        onChange={(event) => setText(event.target.value)}
        placeholder="Type something..."
      />
      <p>Character count: {text.length}</p>
      <p>Uppercase preview: {text.toUpperCase()}</p>
    </div>
  );
}
```

Her tuş vuruşunda `onChange` çalışır, `setText` state'i günceller, React yeniden render eder, ve input'un `value`'su bu yeni state'i yansıtır. Bu döngü (yaz → state güncelle → yeniden render et → input'ta göster) bir controlled component'in kalbi. Değer state'te olduğu için, aynı anda ekranın başka bir yerinde de (karakter sayısı, önizleme gibi) anında kullanabiliyoruz.

Checkbox ve Select ile Controlled Component'ler

Aynı desen, metin input'larıyla sınırlı değil -- checkbox'lar ve select'ler de controlled yapılabilir:

```
import { useState } from "react";

function ControlledCheckboxExample() {
  const [isSubscribed, setIsSubscribed] = useState(false);

  // Checkboxes use checked instead of value -- the logic is the same:
  // React's state decides whether the checkbox is checked or not.
  function handleChange(event) {
    setIsSubscribed(event.target.checked);
  }

  return (
    <div>
      <label>
        <input type="checkbox" checked={isSubscribed} onChange={handleChange} />
        Subscribe to newsletter
      </label>
      <p>{isSubscribed ? "You are subscribed." : "You are not subscribed."}</p>
    </div>
  );
}
```

Checkbox'larda `value` yerine `checked` kullanılır, ama mantık aynı: işaretli olup olmadığını React'in state'i belirliyor.

```
import { useState } from "react";

function ControlledSelectExample() {
  const [language, setLanguage] = useState("tr");

  return (
    <div>
      {/* Selects are controlled with the same pattern: value + onChange. */}
      <select value={language} onChange={(event) => setLanguage(event.target.value)}>
        <option value="tr">Turkish</option>
        <option value="en">English</option>
      </select>
      <p>Selected language: {language}</p>
    </div>
  );
}
```

`<select>` da tıpkı bir metin input'u gibi `value` ve `onChange` ile kontrol edilir.

Neden Controlled Component?

Değeri React'in state'inde tutmanın en somut faydalarından biri, o değer üzerinde kolayca işlem yapabilmen -- örneğin programatik olarak sıfırlamak:

```
import { useState } from "react";

function ResettingControlledInputExample() {
  const [text, setText] = useState("");

  function handleClear() {
    // Because the value lives in state, clearing the input is as simple
    // as resetting the state -- there's no need to touch the actual DOM
    // element (e.g. via useRef or event.target.reset()).
    setText("");
  }

  return (
    <div>
      <input
        type="text"
        value={text}
        onChange={(event) => setText(event.target.value)}
      />
      <button onClick={handleClear}>Clear</button>
    </div>
  );
}
```

Değer state'te olduğu için, input'u temizlemek yalnızca `setText("")` kadar basit. Eğer değeri React'in dışında (tarayıcının kendi DOM'unda) tutsaydık, aynı işi yapmak için bir DOM referansına (`useRef`) ya da formun kendi `reset()` metoduna ihtiyacımız olurdu.

Özet ve Terimler Sözlüğü

Bir controlled component, `value` (ya da checkbox'larda `checked`) ile state'e bağlanan ve `onChange` ile bu state'i güncel tutan bir form elemanıdır. Değer her zaman React'in state'inde yaşadığı için, o değeri okumak, doğrulamak ya da sıfırlamak kolaylaşır -- input'un kendi başına tuttuğu bir değere güvenmene gerek kalmaz.

Terimler Sözlüğü

Controlled Component — Değeri React'in state'i tarafından belirlenen ve `onChange` ile güncellenen bir form elemanı.

Uncontrolled Component — Değerini kendi başına (tarayıcının DOM'unda) tutan, React state'ine bağlı olmayan bir form elemanı.