


4 Události, stav

Přidáme do našich komponent stav, abychom mohli měnit obsah stránky podle interakce s uživatelem.

[◀ Předchozí
Zobrazování seznamů](#)

[Následující >
Formulářové prvky, efekty](#)

Stav

Pokud chceme udělat naše React komponenty skutečně interaktivní, je potřeba naučit se pracovat s takzvaným **stavem** . Porozumět tomu, jak stav v Reactu funguje, je občas výzva i pro zkušenější programátory. Naštěstí význam slova *stav* v Reactu vychází z významu tohoto slova v realitě. Můžeme proto začít příkladem přímo ze života.

Stav kolem nás

Představte si, že někde přes inzerát prodáváte svoji starou pohovku. V takovém inzerátu se sluší zmínit její *stav*, tedy například, že je mírně vybledlá, na některých místech odřená apod. Zde se můžeme zamyslet, o čem to vlastně mluvíme, když říkáme *stav pohovky*. Mohli bychom říct, že stav je nějaká sada vlastností, které se mohou během života pohovky měnit. Stavem pohovky například není její váha nebo to, zda je rozkládací. To jsou vlastnosti, které se během života pohovky nemění. Pohovka však snadno může časem vyblednout nebo se na některých místech odřít.

Dalším příkladem může být stav vašeho auta. Během používání auta se průběžně mění stav nádrže, počet najetých kilometrů, počet pasažérů v autě a další vlastnosti. U některých aut je možné přidat nebo ubrat sedadla, takže se může například změnit i počet míst v autě. Rozhodně se však nemění například obsah nádrže, počet kol apod.

Uvnitř webové aplikace samozřejmě také nalezneme mnoho stavů. Zaškrtnávací políčko může být zaškrtnuté nebo nezaškrtnuté, vyjížděcí menu může být otevřené nebo zavřené. Stavem je však také například počet objednaných produktů v košíku, počet nepřečtených emailů ve schránce, jméno právě přihlášeného uživatele, text, který uživatel vyplnil do nějakého formuláře a tak dále.

Stav v JavaScriptu

V JavaScriptu každou vlastnost představující stav reprezentujeme jednou proměnnou. Můžeme mít například proměnnou udávající stav nádrže v autě.

```
let tankLevel = 'full';
```

Po dlouhé cestě autem se nejspíš stav nádrže změní takto:

```
tankLevel = 'almost empty';
```

a pokud dlouho ignorujeme blikající kontrolku, můžeme se dostat do velmi nepříjemného stavu

```
tankLevel = 'empty';
```

Takto bychom mohli se stavem pracovat v čistém JavaScriptu. React však pro práci se stavem nabízí speciální konstrukci. Pokud bychom chtěli například vytvořit komponentu `Auto`, která pracuje se stavem nádrže, napsali bychom toto:

```
import React, { useState } from 'react';

const Auto = () => {
  const [tankLevel, setTankLevel] = useState('full');

  return <div className="auto">Tank level: {tankLevel}</div>;
};
```

Pomocí speciální funkce `useState` říkáme Reactu, že chceme vytvořit proměnnou, která se během života komponenty bude měnit. Jedná se o speciální funkci, která vrací pole o dvou prvcích. První položka reprezentuje hodnotu našeho stavu. V našem případě budeme používat `'full'`, `'almost empty'` nebo `'empty'`. Druhá položka je funkce pro změnu hodnoty našeho stavu. Funkce `useState` navíc přijímá jeden vstup, který použije pro počáteční hodnotu stavu.

V našem případě tedy můžeme v komponentě pracovat s proměnnou `tankLevel`, která bude mít při prvním renderu hodnotu `'full'`. Hodnotu `tankLevel` však můžeme například v reakci na nějakou událost změnit. To provedeme voláním

```
setTankLevel('almost empty');
```

Tím spustíme přerenderování komponenty s novým stavem, kde v `tankLevel` bude uloženo `'almost empty'`.

[< Předchozí](#)
Cvičení: Události

[Následující >](#)
Pravidla pro práci se stavem

Kódím.cz

Verze 2.0.0-beta.7