

2 Podmíněné zobrazení

Naučíme se, jak sestavovat CSS třídy a jak se rozhodnout, kterou část JSX chceme zobrazit.

[◀ Předchozí](#)
[Úvod do Reactu](#)

[Následující >](#)
[Zobrazování seznamů](#)

Ve vanilla JavaScriptu jsem se nejdříve učili vytvářet statický obsah stránky a později jsme postupně přidávali interaktivitu. V Reactu budeme postupovat stejně. V této lekci uděláme další krok k tomu, aby naše stránky mohly být dynamičtější. Bude se nám k tomu hodit jedna hezká pomůcka z čistého JavaScriptu, kterou jsme v předešlých lekcích vynechali.


Ternární operátor

Představte si situaci, kdy chceme uživateli zobrazit jednoduchou zprávu podle jeho věku.

```
let message = null;
if (age >= 18) {
  message = 'Smíš vstoupit';
} else {
  message = 'Utíkej za mamkou';
}
```

Tato podmínka vypadá velmi přímočaře. Má však určité nevýhody.

1. Jde o celkem dlouhý kód pro velmi jednoduchou věc.
2. Musíme požívat proměnnou `let`, čemuž se snažíme co nejvíce vyhnout.

JavaScript nám pro tuto situaci nabízí zkratku, které se říká **operátor pro podmíněný výraz** .

```
const message = age >= 18 ? 'Smíš vstoupit' : 'Utíkej za mamkou';
```

Tento operátor se dá použít vždy, když chceme do nějaké proměnné uložit různé hodnoty na základně nějaké podmínky.

V Reactu se nám tento operátor bude hodit ve více situacích. První z nich je situace, kdy chceme zkonstruovat název CSS třídy podle nějaké podmínky. Vezměme položku nákupního seznamu napsanou jako React komponentu.

```
const ShoppingItem = (props) => {  
  const { product, amount } = props;  
  
  return (  
    <li className="item">  
      <div className="item__product">{product}</div>  
      <div className="item__amount">{amount}</div>  
    </li>  
  );  
};
```

V této verzi komponenty nám zatím chybí možnost označit položku jako koupenou. Předáme tedy komponentě novou `prop` s názvem `done` a použijeme ji takto.

```
<ShoppingList product="jablka" amount="1 kg" done={true} />
```

Možná si ještě vzpomenete, že zaškrtnutí položky jsme dělali pomocí CSS třídy `item__done--tick`. Vybraná položka by tak měla mít atribut `className` nastaven takto.

```
<div className="item__done item__done--tick">
```

Obsah atributu `className` chceme zkonstruovat dle hodnoty `props.done`. To bychom mohli udělat pomocí podmínky.

```
const ShoppingItem = (props) => {  
  const { product, amount, done } = props;  
  
  let tickClass = 'item__done';  
  if (done) {  
    tickClass = 'item__done item__done--tick';  
  }  
  
  return (  
    <li className="item">  
      <div className="item__product">{product}</div>  
      <div className="item__amount">{amount}</div>  
      <div className={tickClass}></div>  
    </li>  
  );  
};
```

Díky podmíněnému operátoru si ovšem situaci můžeme zjednodušit takto.

```
const ShoppingItem = (props) => {  
  const { product, amount, done } = props;  
  
  const tickClass = done ? 'item__done item__done--tick' : 'item__done';  
  
  return (  
    <li className="item">  
      <div className="item__product">{product}</div>  
      <div className="item__amount">{amount}</div>  
      <div className={tickClass}></div>  
    </li>  
  );  
};
```

Dokonce bychom hodnotu ani nemuseli ukládat do proměnné a použít podmíněný operátor přímo na místě.

```
const ShoppingItem = (props) => {  
  const { product, amount, done } = props;  
  
  return (  
    <li className="item">  
      <div className="item__product">{product}</div>  
      <div className="item__amount">{amount}</div>  
      <div  
        className={done ? 'item__done item__done--tick' : 'item__done'}  
      ></div>  
    </li>  
  );  
};
```

Takovýto kód už však může být hůře čitelný, takže je dobré jej používat s mírou a uvážením.

Následující 
Předávání hodnot pomocí props

Kódím.cz

Verze 2.0.0-beta.7