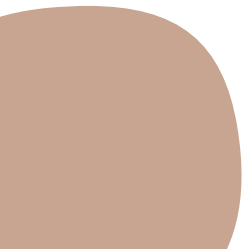
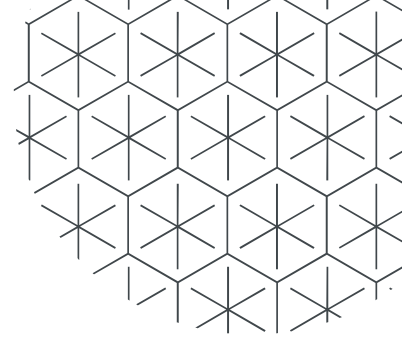


# React a komponenty

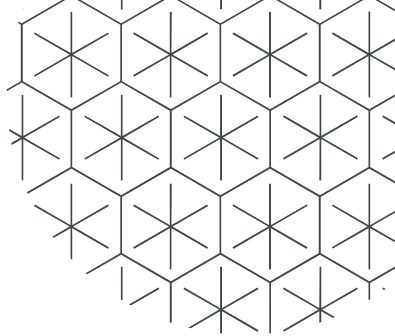
Jonáš Václavek, Tereza Vaňková, 22. 4. 2025

**Poděkování**

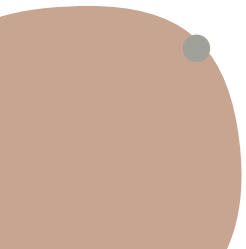
**MEWS**



# Obsah React akademie

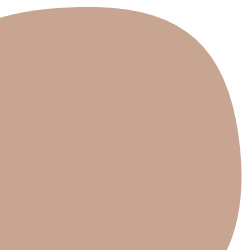
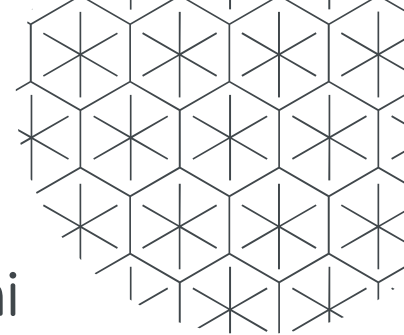


- Co je to React
  - Komponenty a jejich kompozice
  - Předávání dat mezi komponentami
  - Stav komponent
  - Kontext (useContext)
- Navigace mezi stránkami (React Router)
- Komunikace se serverem (useEffect, fetch)
- Tvorba vlastní CRUD aplikace



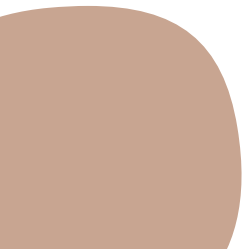
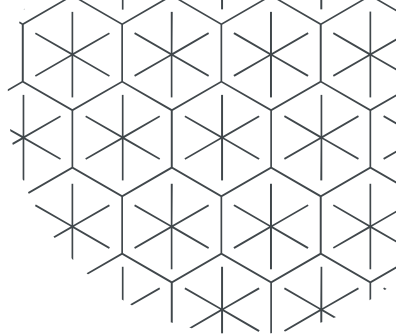
# Co je součástí akademie

- Teorie základů Reactu spojená s praktickými příklady
- Samostatná práce na hodinách a dobrovolné úkoly
- Pomocní koučové na hodinách i na Slacku
- Vzájemná motivace a podpora



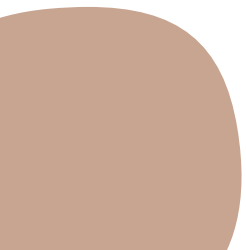
# Co **NENÍ** součástí akademie

- Kompletní React
- Podrobný popis všech React hooks
- Zkoušení z toho, co už umíte nebo co jste se naučily



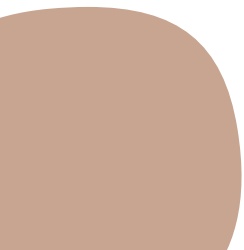
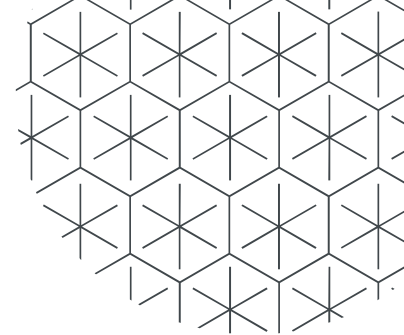
# Prerekvizity

- Znalost HTML, CSS, JS (do té hloubky, jak tyto technologie učíme na předešlých akademiích)
- Instalace IDE (VS Code + rozšíření ESLint, Prettier), Node



# Obsah první lekce

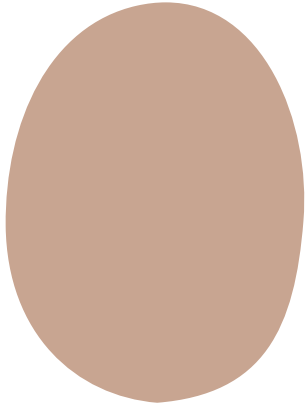
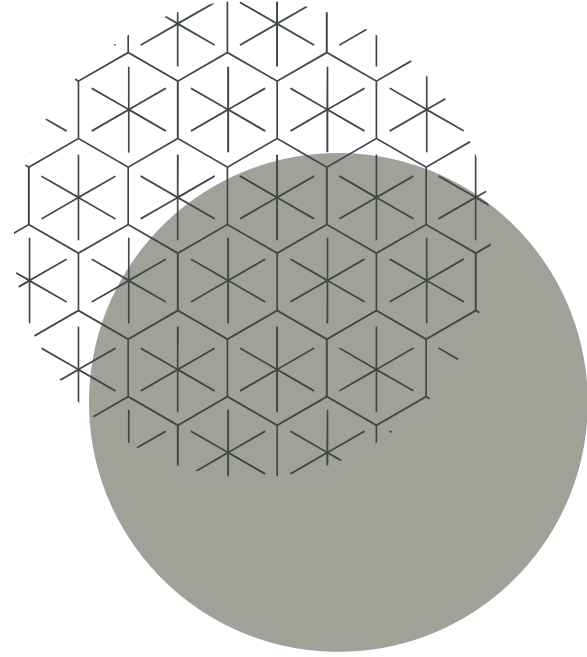
- Co je React?
- Vytvoření React aplikace pomocí VITE
- Úvod do tvorby komponent





Ol

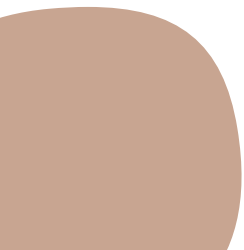
**Co je React?**





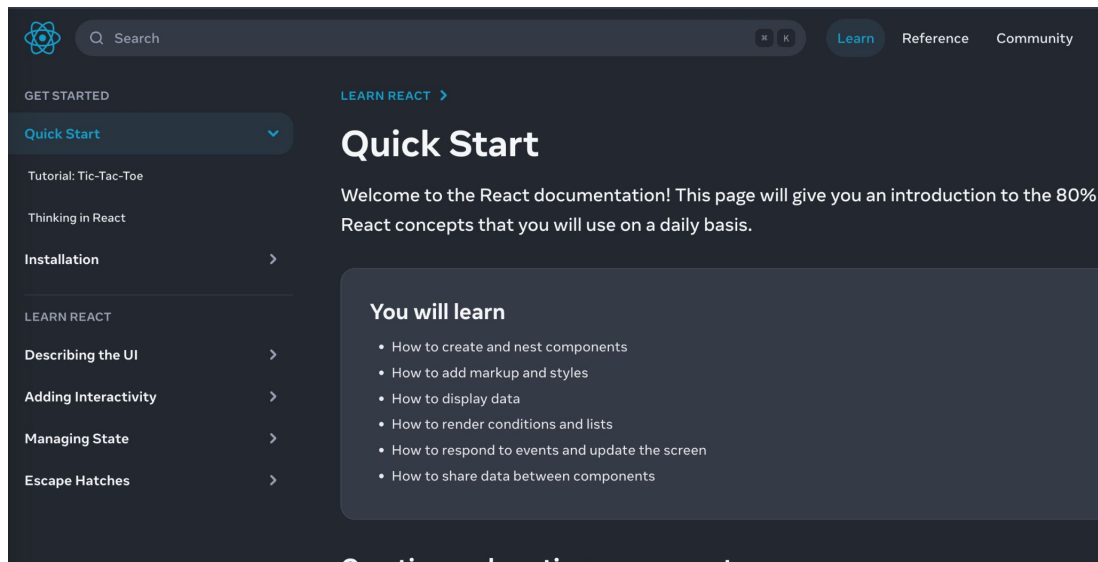
# Co je React?

- JavaScriptová knihovna, která slouží pro tvorbu dynamických webových aplikací.
- V Reactu využíváme psaní komponent, které nám umožňují psaní znovupoužitelného, modulárního, organizovaného kódu.
- React aplikaci si můžeme představit jako strom komponent.



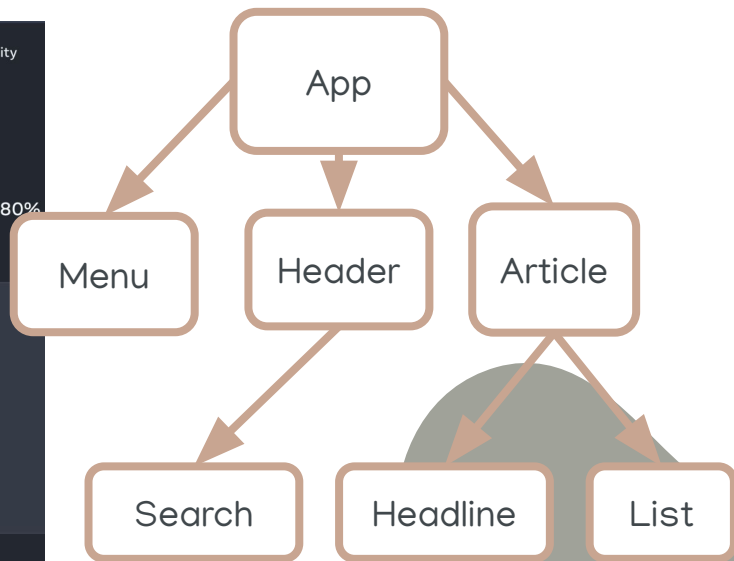
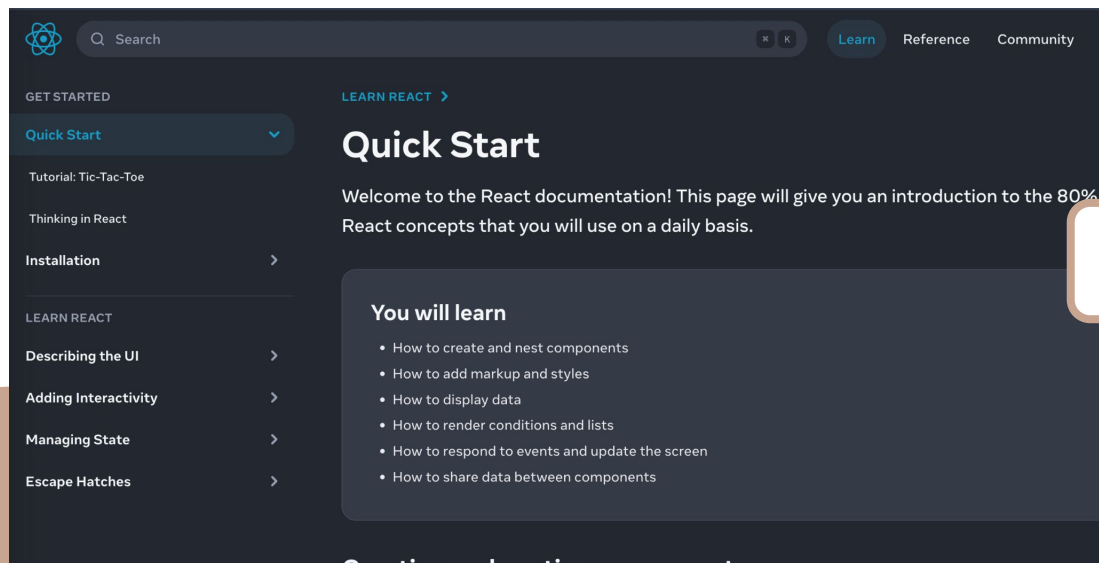
# Co je React?

- React dokumentace <https://react.dev>
- Skvělý zdroj aktuálních informací (+ ukázka react komponent)



# Co je React?

- Zanořování (nesting) komponent, tvorba znovu použitelných (reusable) komponent



# Vytvoření React aplikace pomocí VITE

- V terminálu si otevřu složku, do které chci aplikaci vytvořit
  - `cd` pohyb mezi složkami
  - `mkdir` vytvoří novou složku,
  - `ls` vylistuje položky ve složce (mac / linux)
  - `dir` vylistuje položky ve složce (win)
- `npm create vite@latest`
- Projděte průvodce pomocí šipek a enteru

# Vytvoření React aplikace pomocí VITE

```
jv896821@K74W42DXD2 lecture_1 % npm create vite@latest
```

```
Project name:
```

```
 vite-project
```

# Vytvoření React aplikace pomocí VITE

```
jv896821@K74W42DXD2 lecture_1 % npm create vite@latest
```

◇ Project name:

my-app

◆ Select a framework:

○ Vanilla

○ Vue

● React

○ Preact

○ Lit

○ Svelte

○ Solid

○ Qwik

○ Angular

○ Others

# Vytvoření React aplikace pomocí VITE

```
[jv896821@K74W42DXD2 lecture_1 % npm create vite@latest
```

◆ Project name:

my-app

◆ Select a framework:

React

◆ Select a variant:

○ TypeScript

○ TypeScript + SWC

○ JavaScript

● JavaScript + SWC

○ React Router v7 ↗

○ TanStack Router ↗

# Vytvoření React aplikace pomocí VITE

```
jv896821@K74W42DXD2 lecture_1 % npm create vite@latest  
◇ Project name:  
  my-app  
◇ Select a framework:  
  React  
◇ Select a variant:  
  JavaScript + SWC  
◇ Scaffolding project in /Users/jv896821/Documents/repos/react-akademie/pro_lektora/lecture_1/my-app...  
- Done. Now run:  
  cd my-app  
  npm install  
  npm run dev  
jv896821@K74W42DXD2 lecture_1 %
```



# Vytvoření React aplikace



## Vite + React

count is 4

Edit `src/App.jsx` and save to test HMR

Click on the Vite and React logos to learn more

# Struktura React aplikace

- node\_modules – to nás nemusí trápit, tuhle složku nebudeme vůbec upravovat
- public – public assets naší webové aplikace jako videa, obrázky, soubory.
- src – zdrojový kód naší aplikace. Zde budeme přepisovat a upravovat vše podle toho, jak chceme, aby se naše aplikace chovala a vypadala.

# Struktura React aplikace

- index.html basic html šablona a v body je `<div>` obsahující `id="root"`, což je container pro naši aplikaci
- package.json – balíčky, knihovny a jejich verze, skripty
- README.md
- Ostatní složky a soubory můžeme zatím ignorovat



02

# **Tvorba komponent**



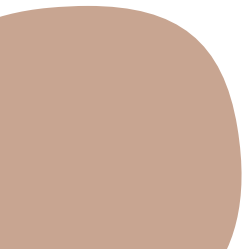
# Tvorba komponenty

- Ve složce src si vytvořím nový soubor a pojmenuji ho například FirstComponent.jsx
  - používám tzv. Pascal Case
  - koncovka `.jsx` znamená, že komponenta je psaná v JavaScript XML

# Tvorba komponenty



1. Vytvořím funkci (kterou za použití PascalCase pojmenuju)
2. Funkce musí něco vracet – return
3. Funkce mi bude vrací JSX (velmi podobné html, jen malé rozdíly)
4. Vše co vracím musí být obalené v jednom elementu
5. Funkci musím exportovat, abych ji mohl použít jinde v aplikaci



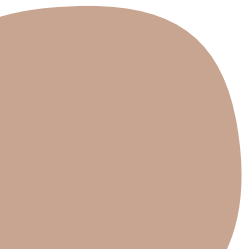
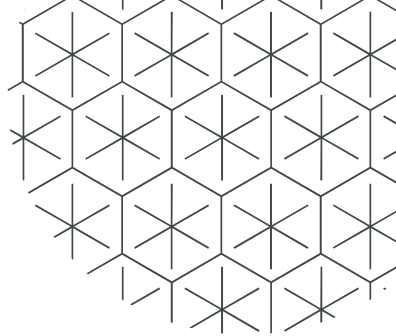
# Naming (vsuvka)

camelCase

snake\_case

kebab-case

PascalCase



# Komponenta je funkce

```
function FirstComponent() {
```

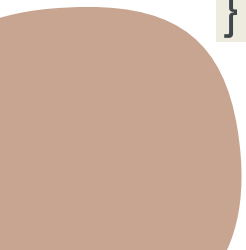
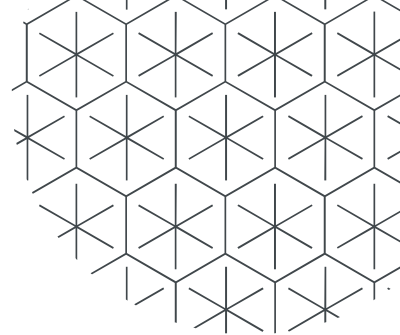
```
  return
```

```
}
```

```
const FirstComponent = () => {
```

```
  return
```

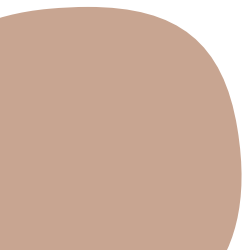
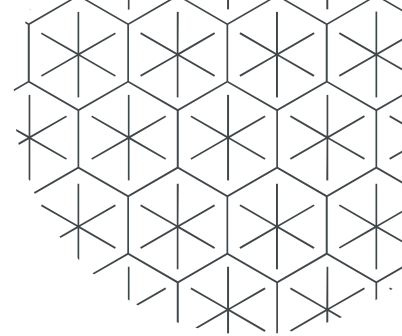
```
}
```





# JSX

- Komponenty vrací JSX
- Jedná se o syntaktické rozšíření pro JS
- Vypadá jako HTML
- Mocné jako JS
- Používá se zejména v React aplikacích



# JSX vs HTML

- Místo class (jako u HTML) používám className

```
<h1 className="FirstComponent"> Hello </h1>
```

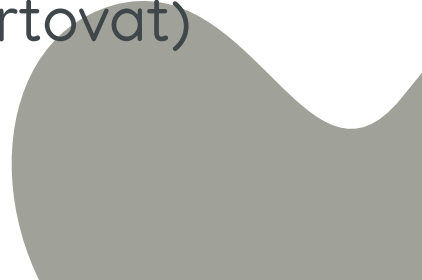
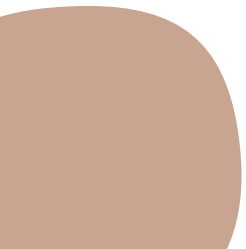
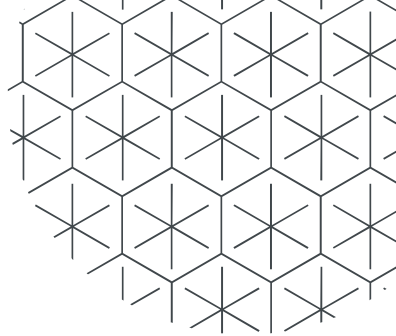
- V JSX můžeme používat JS výrazy v kudrnatých závorkách

```
const name = 'ReactGirls';
```

```
const element = <h1>Hello, {name}</h1>;
```

# JSX

- Všechny tagy musí mít zavírací tag
  - `<div>...</div>` / `<span>...</span>`
  - nebo být tzv. self-closing `<div />`
- Standardně vše, co komponenta vrací, musí být v jednom elementu
- Víc elementů můžu zabalit do fragmentu
  - `<></>`
  - `<Fragment></Fragment>` (potřeba importovat)



# Tvorba komponenty

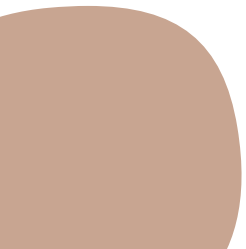
- Komponentu musím exportovat a poté importovat v tzv. parent komponentě

```
import { FirstComponent } from "../FirstComponent";
```

# Tvorba komponenty

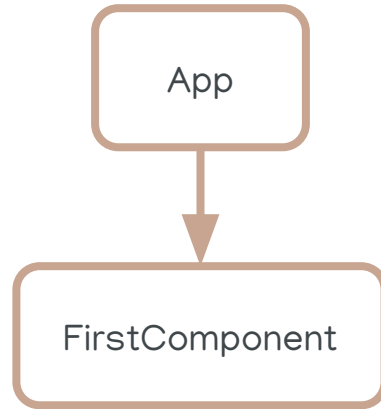


1. Vytvořím funkci (kterou za použití PascalCase pojmenuju)
2. Funkce musí něco vracet – return
3. Funkce mi bude vrací JSX (velmi podobné html, jen malé rozdíly)
4. Vše co vracím musí být obalené v jednom elementu
5. Funkci musím exportovat, abych ji mohl použít (importovat) jinde v aplikaci



# Tvorba komponenty

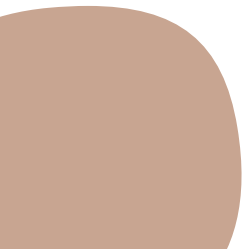
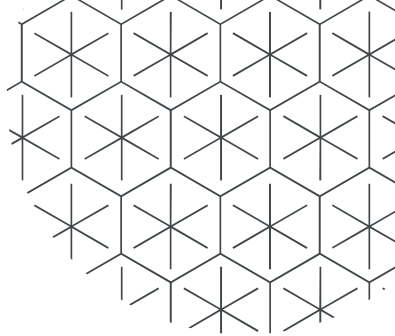
- Současný stav aplikace



# Tvorba komponenty

- Vložení proměnné do JSX

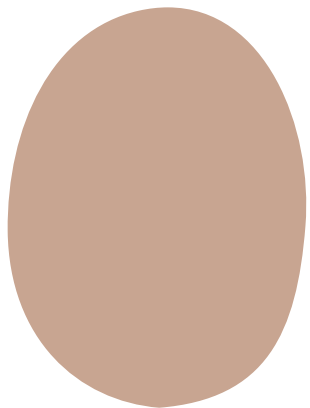
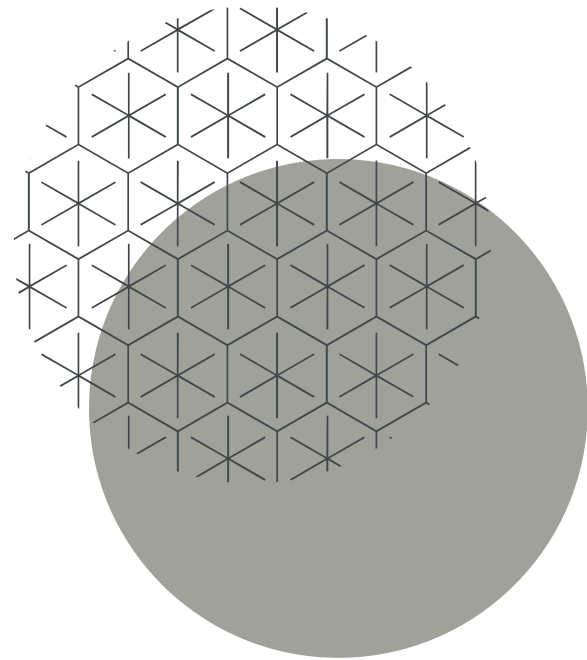
```
export const FirstComponent = () => {  
  const componentName = "FirstComponent";  
  return (  
    <h1>My component {componentName}</h1>  
  )  
}
```





03

**Příklady**







# Příklad



Tvorba nové  
komponenty  
SecondComponent

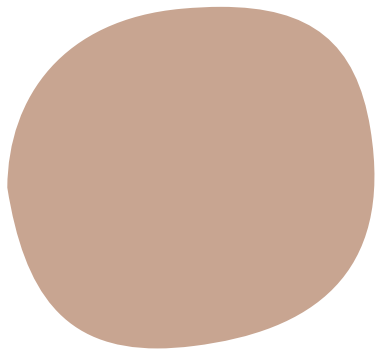




## Příklad

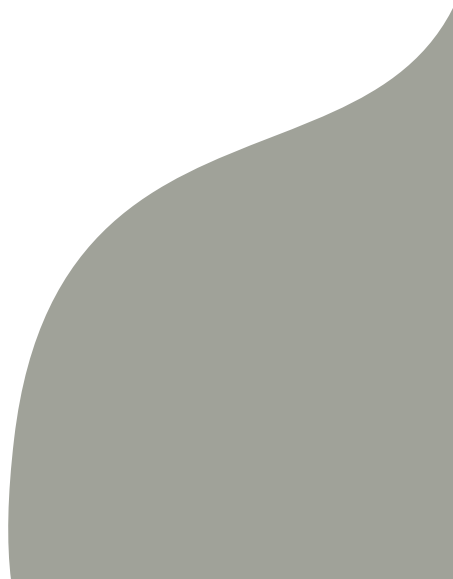
Tvorba proměnné, do  
které si uložím text, který  
použiju v JSX





# Příklad


Tvorba Button  
komponenty



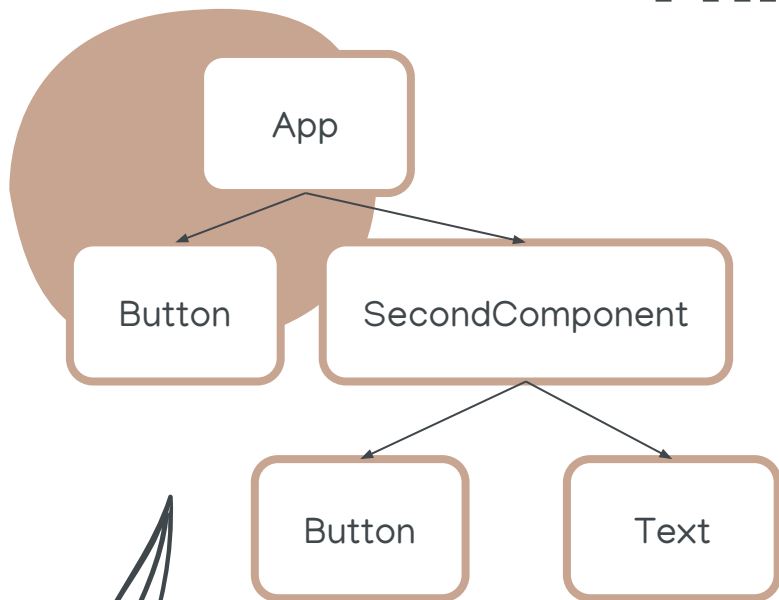


# Příklad

Tvorba funkce, která  
bude volat `console.log()`  
na kliknutí button



# Příklad



- App komponenta bude parent dvou komponent Button a SecondComponent
- SecondComponent bude obsahovat `<h1></h1>` s nadpisem “Header” a dvě komponenty Button a Text
- Button komponenta bude jedna a ta stejná použitá na dvou místech v aplikaci
- Button komponenta na klik zavolá funkci `console.log()` s textem “Button was clicked”
- Text komponenta obsahuje proměnnou `text`, kde bude řetězec “Hello world”, který se vypíše v paragrafu na stránce

# Příklad

Click me

## Header

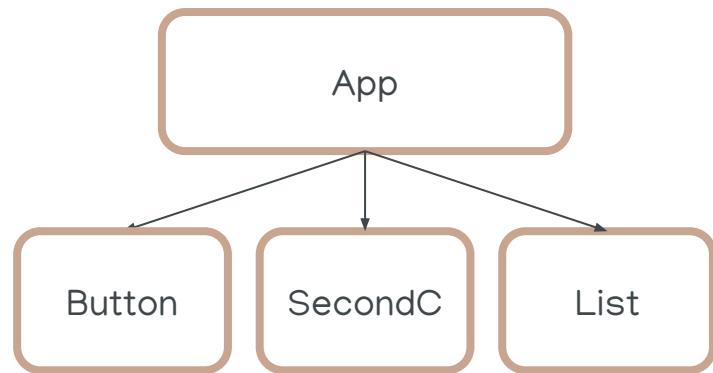
Click me

Hello world!



# Příklad

- Použití `.map()` metody
- Vytvoř a vykresli komponentu `List`, ve které bude proměnná `'menu'` a v ní hodnoty `["Úvod", "O nás", "Článek", "Kontakt"]`
- Pomocí metody `.map()` zobraz jednotlivé prvky pole



# Příklad

- Ulož data zaměstnanců do proměnné (data na Slacku)
- Vytvoř a vykresli komponentu `EmployeeTable`, která renderuje seznam zaměstnanců jako tabulku
- Pomocí metody `.map()` zobraz jednotlivé řádky tabulky;





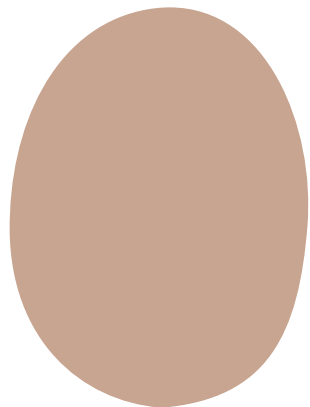
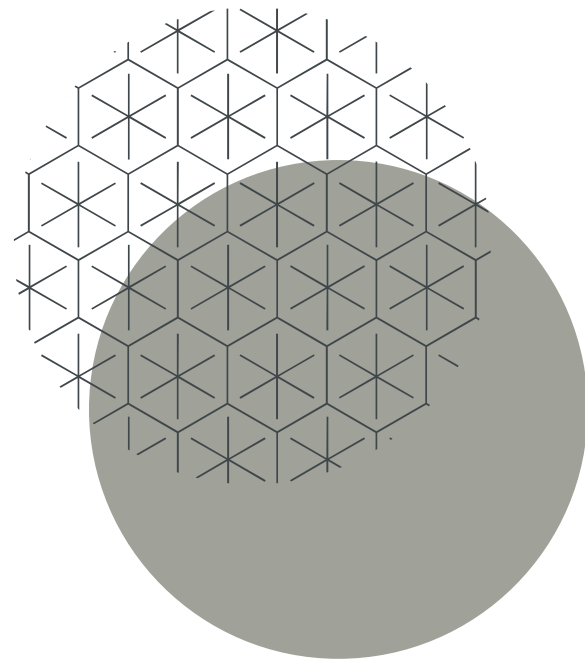
# Stylování

- plain CSS (známe z kurzů HTML, CSS), nicméně to se v reálných aplikacích až tak často nepoužívá
- CSS in JS
- CSS modules – oproti plain CSS nemám třídy definované globálně, tak si je nepřepisuju
- Knihovny – Styled Components, MaterialUI, Bootstrap, ChakraUI, TailwindCSS apod
- CSS Preprocessor



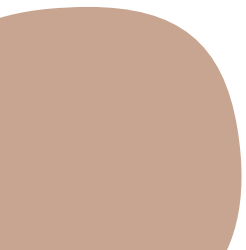
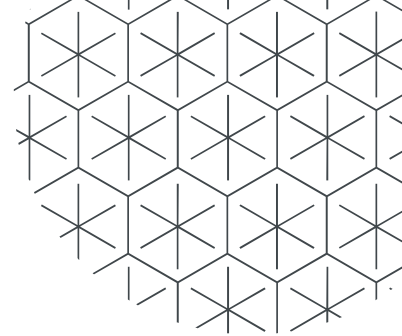
04

**Shrnutí**



# Shrnutí

- Co je React
- Vytvoření react aplikace (pomocí VITE)
- Úvod do tvorby komponent





# Díky za pozornost!



CREDITS: This presentation template was created  
by **Slidesgo**, including icons by **Flaticon**,  
infographics & images by **Freepik**

Please keep this slide for the attribution