Gymnázium, Praha 6, Arabská 14

Předmět Programování



ROČNÍKOVÝ PROJEKT

Vliv implementace prvků přístupnosti do webu a knihovna s nadstandartními funkcemi

Autoři: Filip Beneš

IV.E 2023/24

Škola: Gymnázium, Praha 6, Arabská 14

Kraj: Hlavní město Praha

Konzultant: Mgr. Jan Lána

30. březen 2024 Praha

#### Prohlášení

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 30. března 2024 Filip Beneš ……………………………………………………

#### Anotace

Tato ročníková práce se zaměřuje na implementaci prvků přístupnosti do webových aplikací v jazyce Javascript a frameworku VueJS, s primárním cílem vytvořit knihovnu s nadstandardními funkcemi, včetně Text to Speech a Speech to Text. Cílem je usnadnit nevidomým uživatelům pohyb a interakci na webových stránkách, přičemž analýza technologických možností bude klíčovým prvkem.

Výsledná knihovna představuje inovativní nástroj pro vývojáře, kteří chtějí vylepšit přístupnost svých webových projektů. Zohledněním specifických potřeb nevidomých uživatelů a využitím moderních technologií přispěje k rozvoji inkluzivního designu na internetu.

#### Klíčová Slova

#### Annotation

#### Keywords

#### Obsah

[1. Zadání 6](#_Toc151975371)

[2. Úvod 7](#_Toc151975372)

[3. Filip 7](#_Toc151975373)

[3.1 Beneš 7](#_Toc151975374)

[3.2 Vbciejc 7](#_Toc151975375)

[4. Aplikace 8](#_Toc151975376)

[5. Jednotlivé funkce 8](#_Toc151975377)

[6. Použité technologie 8](#_Toc151975378)

[6.1 Vue.js 8](#_Toc151975379)

[6.2 Scss 10](#_Toc151975380)

[6.3 Externí knihovny 10](#_Toc151975381)

[7. Závěr 10](#_Toc151975382)

[8. Použité zdroje 11](#_Toc151975383)

[8.1 Internetové zdroje 11](#_Toc151975384)

[8.2 Knižní zdroje 11](#_Toc151975385)

[9. Seznam obrázků 11](#_Toc151975386)

[10. Seznam ukázek kódu 11](#_Toc151975387)

# Zadání

**Téma projektu:** Vliv implementace prvků přístupnosti do webu a knihovna s nadstandartními funkcemi

**Jméno a příjmení:** Filip Beneš

**Popis projektu:** Tato ročníková práce se zaměřuje na implementaci prvků přístupnosti do webu, správného plánování UI/UX a jejich následného vlivu na spektrum uživatelů webu – tedy uživatele, kteří prvky potřebují a uživatele, kteří je mohou využívat nepodmínečně. Skládá se tedy ze dvou komplementárních částí:

* **Implementace prvků přístupnosti**

tyto prvky jsou většinou standardem W3C WAI. Patří mezi ně například alternativní popisky obrázků (alt), ovládání pomocí klávesnice, přepis audia nebo barevné módy. Cílem je spojit se s nějakou organizací sdružující například zrakově postižené a využít jejich dalších podnětů na základě zpětné vazby.

* **Knihovna s nadstandartními funkcemi**

další funkce, které nepatří mezi ty standartní, mohou být obzvláště přínosné i pro běžné uživatele a zároveň je možné je implementovat obecně – nezávisle na webu, je cílem seskupit do knihovny. Tato knihovna by pak tyto funkce poskytovala s návodem na jejich použití. Mezi tyto funkce patří například TTS (Text To Speech) nebo ovládání pomocí hlasu.

**Platforma:** Javascript, Vue.js, HTML, SCSS

# Úvod

Vstup do světa moderního webového vývoje přináší s sebou nejen technologický pokrok, ale také odpovědnost za zajištění, aby digitální prostředí bylo dostupné pro všechny. Tato ročníková práce se zaměřuje na analýzu vlivu implementace prvků přístupnosti do webových aplikací, přičemž klade důraz na použití programovacího jazyka Javascript a frameworku VueJS. Současně se bude zkoumat význam a přínosy knihovny s nadstandardními funkcemi v kontextu celkové uživatelské zkušenosti.

# Filip

## Beneš

## Vbciejc

# Aplikace

# Jednotlivé funkce

# Použité technologie

Před zahájením práce na projektu jsme museli důkladně promyslet jednotlivé technické možnosti a nástroje, ve kterých budeme aplikaci vytvářet. V úvahu jsme brali naše dosavadní znalosti, které ovlivnily výběr programovacího jazyka (JavaScript – JS), ale také výhody jednotlivých nástrojů, abychom dosáhli co možná nejvýkonnějšího řešení, které bude fungovat svižně a zároveň bezpečně a bez chyb. Pro vývoj frontendu jsme se rozhodli použít JS framework, abychom dosáhli výsledku v kratším čase a s vyšší spolehlivostí. Dalším důvodem, proč jsme se nespokojili pouze se základní funkcionalitou JS, byla vůle pracovat s technologiemi hojně používanými velkými softwarovými giganty jako jsou Facebook, Netflix, Nintendo, či samotný Google, kteří využívají Vue.js stejně jako naše aplikace. Node.js využívají další velké instituce jako například NASA, Uber, PayPal či už zmiňovaný Netflix. Vzhledem k níže popsaným výhodám Node.js jsme si toto prostředí zvolili pro vytvoření serverové části aplikace (backend).

## Vue.js

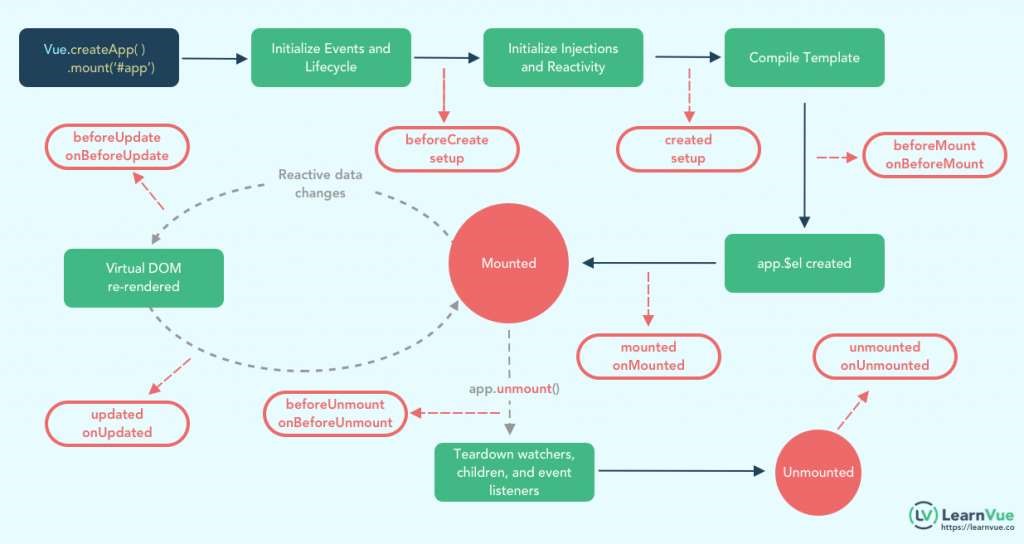
Při volbě frontend frameworku, který je postaven na HTML, CSS a JS jsme se rozhodovali ve finále mezi Vue.js a React.js. Vue.js je JS framework sloužící k vytváření uživatelských rozhraní. Pokrývá většinu běžných funkcí potřebných k vývoji frontendu a je navržen tak, aby byl flexibilní, rychlý a přizpůsobitelný. Je postaven na standardním HTML, CSS a JS a poskytuje deklarativní programovací model založený na komponentách, které pomáhají efektivně vyvíjet uživatelská rozhraní. Definitivní volbou se pak stalo Vue.js díky své menší velikosti výsledné aplikace a vyššímu výkonu. Rozšiřuje standardní HTML o šablonu syntaxe, která umožňuje popsat výstup HTML na základě stavu JS. Vue.js také automaticky sleduje změny stavu JS a při změnách reaktivně aktualizuje DOM[[1]](#footnote-1). V našem projektu používáme jeho nejnovější verzi.

Komponenty, označené příponou .vue, mohou být vytvořeny ve dvou různých stylech API[[2]](#footnote-2): Options API a Composition API. V našem projektu používáme novější Composition API (která řeší určité nedostatky o něco starší Options API), kde definujeme logiku komponentu pomocí importovaných funkcí API. V SFC[[3]](#footnote-3) se Composition API obvykle používá s <script setup>, nebo ekvivalentním <script> s metodou setup(). Setup je funkce, díky které Vue.js provádí změny v době kompilace, které nám umožňují používat Composition API s nižší zátěží při běhu aplikace. Například proměnné/funkce nejvyšší úrovně deklarované a vracené v setup) jsou přímo použitelné v šabloně DOMu.

Další hlavní výhoda Vue.js je v reaktivitě, díky které je možné okamžitě reagovat na změnu dat jak v DOMu, tak ve změně jiných závislých dat. To zjednodušuje práci s těmito daty a optimalizuje tyto úkony.

Requesty na naší API, které provádí získávání okamžitě zobrazovaných dat, provádíme co nejdříve v cyklu Vue.js – beforeCreate, který se provádí hned jako první, abychom vytvořili pro uživatele co nejrychlejší a nejplynulejší prostředí.

Obrázek 10: Vue.js lifecycle hooks



K přechodům mezi jednotlivými obrazovkami aplikace (tzv. views) používáme Vue Router. Ten hlídá veškeré změny v URL a zajišťuje plynulé přechody bez opětovného načítání celé stránky včetně předávání dat přes URL. Aby se zvýšila rychlost načítání, využíváme zde tzv. metody „lazy loading“ tak, že jednotlivé views se načítají až při jejich prvotním použití. Pokud se uživatel pokusí zadat URL, která není v aplikaci platná, je automaticky přesměrován na tzv. error page s chybou 404 - Not found.

Data jako například data uživatele, hlavní a načtené menu, načtené aktivity a učebnice jsou uložena sdíleně pro všechny komponenty ve Vuex Store. Tato data lze měnit pouze přes speciální metody zvané mutace, což zajišťuje standardizovanou a uvědomělou změnu dat. Také se zde nachází sdílené funkce, tzv. akce, jako jsou například funkce pro requesty na naší API nebo funkce pro smazání článku.

## Scss

Vzhledem k celkovému rozsahu projektu bylo nutné zvolit prostředek, v tomto případě preprocesor, pro jednoduší práci s CSS. SCSS je zkratka pro Sassy Cascading Style Sheets a jde de facto o pokročilejší variantu CSS. Kromě přípony souborů .scss se od standartního CSS liší i větším množstvím funkcí, které přispívají k více spolehlivému a DRY[[4]](#footnote-4) CSS.

Rozšiřuje CSS o řadu funkcí, jako jsou například mixiny, funkce, vnoření tříd apod. Mixiny jsou vhodné pro vytknutí stejných částí stylů do jednoho „bloku“, který pak stačí referovat na potřebných místech pomocí „@include“. Lze jim také vkládat argumenty, které pak mohou být použity ve stylech mixinu. Podobné mixinům jsou funkce, ale ty nevrací blok stylů, nýbrž mohou vracet hodnotu použitelnou ve stylu. Funkcím lze také vkládat argumenty. Výhodou také je, že všechny CSS verze jsou kompatibilní s SCSS. Pro optimalizaci kompilace je možné v názvu souboru použít prefix „\_“ a tím z něj vytvořit tzv. modul. Ten se nejdříve vloží do souborů, které ho importují, a až tyto soubory se zkompilují. Tuto funkci využíváme například pro soubor s mixiny.

## Externí knihovny

# Závěr

# Použité zdroje

## Internetové zdroje

Brewster, C. (2023). 15 Companies That Use Node.js in 2023 Successfully. Získáno 2023, z

Trio: https://www.netguru.com/blog/vue-js-companies

## Knižní zdroje

Bednařík, M. (1981). Problematika informační struktury učebnice fyziky. Univerzita

Palackého v Olomouci, Acta UPOL Fac RN, Olomouc. Získáno 2023

# Seznam obrázků

Obrázek 1: Komponenty učebnice (Průcha, 1998, s. 21) ........................................................... 8

# Seznam ukázek kódu

Kód 1: Funkce pro změnu a „lazy loading“ jazyků aplikace .................................................... 35

1. DOM – Document Object Model [↑](#footnote-ref-1)
2. API – Application Programming Interface [↑](#footnote-ref-2)
3. SFC – Single-File Component [↑](#footnote-ref-3)
4. DRY – Don’t Repeat Yourself [↑](#footnote-ref-4)