Gymnázium, Praha 6, Arabská 14

Předmět Programování



ROČNÍKOVÝ PROJEKT

Vliv implementace prvků přístupnosti do webu a knihovna s nadstandartními funkcemi

Autor: Filip Beneš

IV.E 2023/24

Škola: Gymnázium, Praha 6, Arabská 14

Konzultant: Mgr. Jan Lána

27. února 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 27. února 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

Nevidomí lidé, knihovna, TTS (text to speech), STT (speech to text)

#### Annotation

#### Keywords

#### Obsah

[1. Zadání 6](#_Toc154841909)

[2. Úvod 7](#_Toc154841910)

[3. Uživatelská část 8](#_Toc154841911)

[3.1 Řečník 8](#_Toc154841912)

[3.2 Hlasový navigátor 8](#_Toc154841913)

[3.3 Speech fucus 8](#_Toc154841914)

[4. Administrátorská část 10](#_Toc154841915)

[4.1 Instalace knihovny 10](#_Toc154841916)

[4.2 Nastavení a spuštění knihovny 10](#_Toc154841917)

[5. Knihovna 10](#_Toc154841918)

[5.1 Založení knihovny 10](#_Toc154841919)

[6. Použité technologie 11](#_Toc154841920)

[6.1 Vue.js 11](#_Toc154841921)

[6.2 Scss 13](#_Toc154841922)

[6.3 Externí knihovny 14](#_Toc154841923)

[7. Závěr 14](#_Toc154841924)

[8. Použité zdroje 15](#_Toc154841925)

[8.1 Internetové zdroje 15](#_Toc154841926)

[8.2 Knižní zdroje 15](#_Toc154841927)

[9. Seznam obrázků 15](#_Toc154841928)

[10. Seznam ukázek kódu 15](#_Toc154841929)

# 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.

# Uživatelská část

Projekt klade důraz především na uživatele a jsou také alfou a omegou celé práce, proto uživatelská část je udělaná do posledního detailu.

## Řečník

Text to Speech (TTS) představuje klíčový prvek vylepšující uživatelskou stránku webových aplikací, a to zvláště pro uživatele s omezením vizuálního vnímání. Tato funkcionalita transformuje textový obsah na zvukový výstup, což má významné důsledky pro dostupnost informací. Z hlediska uživatelské stránky se jedná o revoluční nástroj, který překračuje bariéry pro lidi se zrakovým postižením, ale může být prospěšný i pro ty, kteří preferují poslech před čtením.

Pro nevidomé a slabozraké uživatele má Text to Speech klíčový význam při přístupu k obsahu webových stránek. Bez ohledu na to, zda jde o běžný text na blogu, komentáře pod příspěvky, nebo komplexní informace na odborném webu, TTS umožňuje těmto uživatelům snadný a plnohodnotný přístup k obsahu bez nutnosti spoléhat se výhradně na vizuální vnímání. Tímto způsobem se zvyšuje dostupnost informací a vytváří se inkluzivnější online prostředí.

Po uživatelské stránce se Řečník spouští pomocí klávesnice, konkrétněji je to defaultně nastavené na „2“. Po stisknutí se spustí oznamovací hlas, který uživateli řekne, že je Řečník spuštěný. Poté řekne první text k předříkání. Řečník lze ovládat pomocí šipek na klávesnici. Šipka doprava přeskočí na další element a šipka zpět se vrátí na předchozí element. Funkcionalitu lze uzavřít dvěma způsoby. Buď stisknutím klávesy „espace“, nebo klávesovou zkratkou, kterou se funkcionalita spouští.

## Hlasový navigátor

Další klíčovou součástí uživatelské části naší aplikace je Hlasový Navigátor. Tato funkce je navržena s důrazem na interaktivní a efektivní průvodce webovými stránkami pomocí hlasových pokynů, což zvyšuje přístupnost pro uživatele s omezením motoriky nebo vizuálního vnímání.

Hlasový navigátor využívá funkci Speech to Text (SpeechRecognition) v JavaScriptu umožňuje převod mluvené řeči na text. To poskytuje možnost interakce s webovou aplikací nebo stránkou pomocí hlasových příkazů, což může být užitečné pro uživatele s omezením nebo pro vytvoření hlasově aktivovaných ovládacích prvků. SpeechRecognition je standardní součástí Web Speech API, které je podporováno ve většině moderních prohlížečů, včetně Chrome, Firefoxu a Edge.

Hlasový Navigátor lze snadno aktivovat pomocí klávesové zkratky, což umožňuje uživatelům přizpůsobit si své preference. Defaultně je hlasový navigátor spustitelný pomocí klávesy „3“.

## Speech fucus

Speech Focus představuje klíčovou funkcionalitu aplikace, která zajišťuje orientaci na webových stránkách pomocí focusAble elementů. Tyto elementy jsou definovány jako takové, které mohou být aktivovány klávesou „TAB“. V HTML jsou některé elementy přednastaveny jako focusAble, ale pro ty, které tuto možnost nemají implicitně, nebo pro ty, které by neměly být zobrazeny, lze využít atributu „tabindex“. Tímto atributem lze nastavit prioritu zobrazení pro daný element. Pokud je hodnota atributu nastavena na „0“, znamená to, že element nebude aktivován pomocí klávesy „TAB“. Další hodnoty atributu určují pořadí prioritního zobrazení elementu (např. první, druhý, až poslední).

Speech Focus může být aktivován uživatelem stisknutím klávesy „1“, a následně je uživateli potvrzeno zapnutí této funkce. Ovládání probíhá výhradně pomocí klávesy „TAB“, což umožňuje postupné proklikávání mezi jednotlivými focusAble elementy. Funkci lze vypnout dvěma způsoby: opětovným stiskem klávesy „1“ nebo klávesou „ESCAPE“ (která zároveň vypíná celou knihovnu a všechny její funkce).

Speech Focus analyzuje obsah na focus elementu a hledá texty k přečtení. Nejdříve o jaký typ elementu se jedná. To slouží k tomu, aby poté vybral z pole pořadí priorit, vyhledávání možných textů k přečtení. A poté postupně podle priorit zkouší, zda se tam daný text nachází a pokud ano přečte ho a zastaví se. Například pro elementy typu input, kde není přímo dostupný textContent, může algoritmus začít hledáním u popisků (labels) a pokračovat například u titulků. Naopak u odkazů (link) je pole prioritizováno jinak, přičemž textContent je na prvním místě, následovaný titulkem atd…

Obrázek kódu, pole priorit a switch s elementy

# Administrátorská část

V této kapitole se zaměříme na administrátorskou část aplikace, kde budou podrobněji rozebrány dvě klíčové podkapitoly: 4.1 Instalace Knihovny a 4.2 Nastavení a Spuštění Knihovny. Tyto sekce poskytnou uživatelům a správcům potřebné informace k úspěšnému nasazení, konfiguraci a správě funkcí knihovny.

## Instalace knihovny

Knihovna je dostupná na platformě npm, což umožňuje jednoduché stažení do vašeho projektu využívajícího node.js. Postup instalace je snadný – v konzoli ve vašem projektovém adresáři jednoduše zadejte následující příkaz:

*`npm install blind-friendly-library`*

Po provedení tohoto příkazu bude knihovna stažena a automaticky umístěna do složky s dalšími knihovnami, konkrétně do adresáře (node\_modules).

Obrázek kódu z konzole

## Nastavení a spuštění knihovny

Všechny kroky od instalace, přes nastavení, až po vysvětlení jednotlivých funkcí jsou detailně popsány v dokumentaci, kterou můžete najít v souboru *README.md* ve projektu knihovny nebo na webových stránkách knihovny(<https://www.npmjs.com/package/blind-friendly-library>). Po úspěšné instalaci je potřeba knihovnu importovat a spustit v souboru *main.js* projektu, ve kterém je knihovna využívána.

Konkrétněji importováním souboru *speaker.js* z knihovny. Poté, abyste knihovnu spustili, je potřeba zavolat funkci *setBFL()* a nastavte její základní parametry pro hlasového navigátora a celou knihovnu (volume, rate, pitch, lang).

Po těchto krocích bude například funkce SpeechFocus automaticky aktivní. Ostatní funkce vyžadují přidání vlastních atributů k elementům na webu. Pro funkci řečníka je třeba na tyto elementy přidat custom atribut, konkrétně *data-speaker.* Řečník poté, jakmile je aktivován, bude číst všechny elementy, které mají tento atribut. Autor má následně možnost mezi těmito elementy přepínat pomocí šipek.

Funkce ovládání hlasem…

Obrázek kódu importování knihovny v main.js

# Knihovna

V této kapitole se hlouběji ponoříme do samotné struktury a organizace knihovny. Kapitola 5 poskytne uživatelům důkladné informace o vzniku, publikaci a správě knihovny npm, aby ji mohl poté kdokoliv vytvořit.

## Založení knihovny

Knihovna je založena a publikována na npm (Node.js Package Manager). Npm slouží jako správce balíčků pro JavaScript a je výchozím správcem balíčků pro Node.js. Disponuje online databází knihoven, ke které lze přistupovat jak interně, tak programově. Na webových stránkách npm lze nalézt veřejné knihovny vytvořené komunitou.

Pro úspěšné publikování nové knihovny je nutné řádně nastavit soubor *package.json* ve vašem projektu. Tento soubor musí obsahovat následující informace:

* *Name* (jméno projektu)
* *Description* (podrobnější vysvětlení projektu)
* *Version* (aktuální verze knihovny, začíná se od 0.0.0)
* *Repository* (odkaz na git repositář, kde je knihovna uložena. Obvykle na GitHubu)
* *Keywords* (slova, podle kterých bude vyhledávána knihovna na internetu)
* *Author* (jméno autora)
* *License* (licence, pod kterou je chráněna)
* *Files* (specifikuje, které přesně soubory má obsahovat knihovna)

Obrázek kódu setup v package.json

Po nastavení *package.json* musí autor své nové knihovny vytvořit účet na *npm.com* a přihlásit se. Následně se v konzoli přihlásí do svého účtu pomocí příkazu *`npm login`* a zadá své uživatelské jméno a heslo. Pokud bylo přihlášení úspěšné, zobrazí se tato zpráva:

*Logged in as <uživatelské-jméno> on https://registry.npmjs.org/.*

Následně stačí jednoduše spustit příkaz *`npm publish`* v konzoli pro publikování nové knihovny. Pokud vše proběhne v pořádku, objeví se následující zpráva:

Obrázek z konzole po publikování nové verze

Tím je vaše knihovna úspěšně publikována a dostupná pro další vývojáře přes npm.

# 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

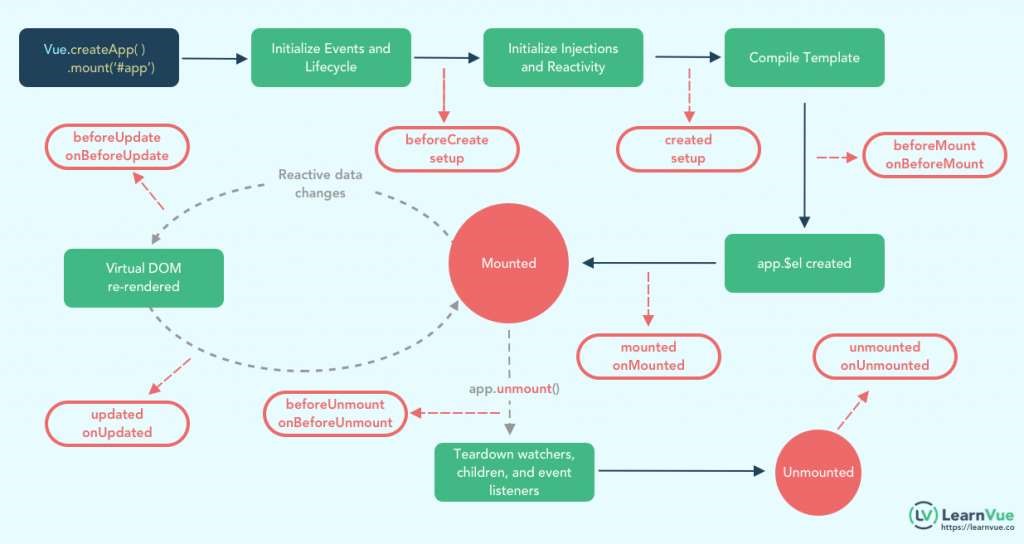
Při volbě frontend frameworku, který je postaven na HTML, CSS a JS jsme se rozhodovali ve finále mezi Vue.js a React.js. Vue.js je JS framework sloužící k vytváření uživatelských rozhraní. Pokrývá většinu běžných funkcí potřebných k vývoji frontendu a je navržen tak, aby byl flexibilní, rychlý a přizpůsobitelný. 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 aplika .............................................. 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)