НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ІМЕНІ ІГОРЯ СІКОРСЬКОГО
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ
КАФЕДРА АКУСТИЧНИХ ТА МУЛЬТИМЕДІЙНИХ ЕЛЕКТРОННИХ СИСТЕМ

«На правах рукопису»
УДК 621.004.522

«До захисту допущено»
Завідувач кафедри
_____С. А. Найда
«_07__» червня 2021 р.

Дипломна робота бакалавра

зі спеціальності 171 Електроніка

на тему: «Аудіо-модуль веб-сайту для інклюзивної освіти»

Виконав: студентка IV курсу, групи ДГ-72

Онисько Ольга Олегівна _____
(підпис)

Керівник доцент каф. АМЕС к.т.н. доц. Богданов О. В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____
(назва розділу) (посада, науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент доцент каф. КЕОА ФЕЛ к.т.н. доц. Кучернюк П. В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань
Студент _____
(підпис)

Київ – 2021 р.

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет Електроніки

Кафедра Акустичних та мультимедійних електронних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 171 «Електроніка»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_ С. А. Найда

«_07__» _червня_ 2021 р.

ЗАВДАННЯ

на дипломну роботу студенту

Онисько Ользі Олегівни

1. Тема проекту (роботи) «Аудіо-модуль веб-сайту для інклюзивної освіти»
керівник проекту (роботи) Богданов Олексій Вікторович, доц. к.т.н.

затверджені наказом по університету від «24» травня 2021 р. № 1316-с

2. Строк подання студентом роботи 01 червня 2021 року

3. Вихідні дані до роботи
існуючі способи синтезу мовлення та його фізичний принцип, розроблення сайту синтезу та розпізнавання мовлення.

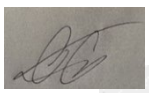
4. Зміст дипломної роботи (перелік завдань, які потрібно розробити): розглянути існуючі програмні реалізації TTS; обрати метод формування голосу з тексту на веб сторінці; реалізувати програмний модуль.
5. Перелік графічного (ілюстративного) матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) Презентація PowerPoint
6. Консультанти розділів проекту (роботи)

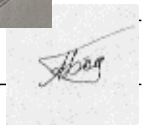
Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 12 квітня 2021 року

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз поточних реалізацій синтезу мовлення	12.04 – 18.04	Виконано
2	Написання 1-го розділу	19.04 – 26.04	Виконано
3	Написання 2-го розділу	27.04 – 09.05	Виконано
4	Оформлення звіту та щоденника	10.05 – 16.05	Виконано
5	Написання 3-го розділу	17.05 – 31.05	Виконано
6	Оформлення роботи	01.06 – 03.06	Виконано

Студент  О. О. Онисько

Керівник проекту (роботи) _____  О. В. Богданов

РЕФЕРАТ

Онисько, О. О. *Аудіо-модуль веб-сайту для інклюзивної освіти* : дипломна робота бакалавра : 171 Електроніка. – Київ, 2021. – 42 с.

Ключові слова: веб-сервіси, віртуальні асистенти, інтерфейси прикладного програмування, адаптивне навчання, засоби комунікації, акустичні програми, синтез мови.

Одним з актуальних питань сьогодення є проблема людей із вадами слуху, зору або дефекти мовлення. Люди, які народилися з обмеженими можливостями потребують інклюзивного навчання. Головне завдання інклюзії полягає в тому, що кожна людина повинна себе відчувати на рівні з іншими людьми.

За допомогою мобільних телефонів, комп’ютерів, що говорять, можна значно підвищити рівень інклюзивної освіти. На сьогоднішній день один з найефективніших методів створення комп’ютерного озвучування мови є Text To Speech (TTS).

В ході виконання даної роботи було проведено дослідження у сфері синтезу мовлення, розглянуто методи синтезу мовлення, а також основні сфери використання. Проведено аналіз сучасних синтезаторів мовлення вказуючи на переваги та недоліки.

Дана робота пропонується, як рішення для інклюзивного навчального контенту. За допомогою Web Speech API було розроблено веб-застосунок з можливістю перетворення тексту в звук та навпаки. Представлений великий вибір мов для обох варіантів.

ABSTRACT

Keywords: web services, virtual assistants, application programming interfaces, adaptive learning, communication aids, acoustic applications, speech synthesis.

One of the most pressing issues today is the problem of people with hearing, vision or speech disorders. People born with disabilities need inclusive education. The main task of inclusion is that everyone should feel on equal terms with other people.

With the help of mobile phones and talking computers, you can significantly increase the level of inclusive education. Meantime one of the most effective methods of creating computer speech is Text-to-Speech (TTS).

During implementation of this work a study was undertaken in the field of the speech synthesis, the methods of speech synthesis, and also basic spheres of the use are considered. The analysis of modern speech synthesizers is conducted pointing their advantages and disadvantages.

This work is offered as a solution for inclusive learning content. The Web Speech API has been used to develop a web application that can convert text to sound and vice versa. A large selection of languages for both options is presented.

ЗМІСТ

ВСТУП

За останні кілька десятиліть комунікаційні засоби були розроблені від калькуляторів, що говорять, до сучасних тривимірних аудіовізуальних додатків. Область застосування синтезу мовлення постійно стає все ширшою, що також приносить більше коштів у галузі досліджень та розробок. У майбутньому, якщо методи розпізнавання мовлення досягнуть належного рівня, синтезоване мовлення може також використовуватися в інтерпретаторах мови або декількох інших системах зв'язку, таких як відеофони, відеоконференції, тощо.

Однією з актуальних проблем, яка вирішується при розробці таких систем, є проблема людей із вадами слуху, зору або дефекти мовлення. За допомогою мобільних телефонів, комп’ютерів, що говорять, можна значно підвищити зручність використання даними приборами для користувачів із цими вадами.

На сьогоднішній день один з найефективніших методів створення комп’ютерного озвучування мови є Text To Speech (TTS). Багато світових компаній проводять дослідження у даній галузі, і створюють нові продукти, починаючи від телефоні закінчуючи роботами, які можуть говорити. Це такі корпорації, як: Google©, IBM®, Microsoft©, Amazon® та багато інших.

1. Синтез мовлення

1.1. Актуальність теми

Як вказано в джерелі [1], синтез мовлення, більш конкретно відомий як перетворення тексту в мовлення (TTS), є комплексною технологією, яка включає багато дисциплін, таких як акустика, лінгвістика, цифрова обробка сигналів та статистика. Основне завдання — перетворення будь-якої текстової інформації в акустичний сигнал. З розвитком технологій цифрової обробки сигналів мета дослідження синтезу мовлення еволюціонувала від зрозумілості та чіткості до природності та виразності.

Відомо з джерела [2], що синтез мови можна знайти в безлічі додатків. Однак важливо знати, що ця технологія спочатку була розроблена, щоб допомогти людям з обмеженими можливостями (особливо людьми із вадами зору) у їх повсякденному житті. Наприклад, дуже відомий Стівен Хокінг через свою важку інвалідність використовував TTS для спілкування з оточуючими людьми. Сьогодні дуже легко знайти сліди TTS у нашому використанні:

- ми їздимо на машинах за допомогою комп’ютеризованих навігаторів;
- слухаємо операторів телефонного зв’язку;
- комп’ютеризовані вибачення на залізничних станціях, коли наші поїзди затримуються, тощо.

1.2. Історія синтезу мовлення

1.2.1. Від механічного до електричного синтезу

Перші зусилля створення синтетичної мови були зроблені понад двісті років тому. У Санкт-Петербурзі 1779 р. російський професор Крістіан Кратценштейн пояснив фізіологічні відмінності між п'ятьма довгими голосними (/ a /, / e /, / i /, / o /, / u /) та виготовив апарати для їх штучного відтворення. Він побудував акустичні резонатори, подібні до голосового тракту людини і активував резонатори вібруючим язичком, як у музичних інструментах(наприклад волинка, дримба, сурма). (рис.1)

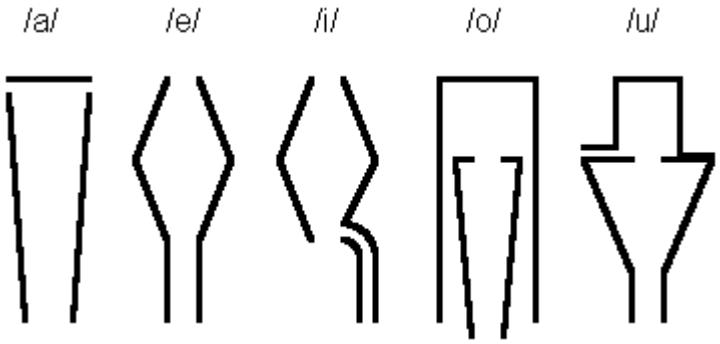


Рис. 1.1 Резонатор Кратценшейта [3]

Кілька років потому, у Відні 1791 року, Вольфганг фон Кемпелен представив свою "Акустично-механічну мовленнєву машину", яка вміла видавати окремі звуки та деякі звукові комбінації [4], [3]. Насправді, Кемпелен розпочав свою роботу до Краценштейна, в 1769 році, і після понад 20 років досліджень він також опублікував книгу, в якій описав свої дослідження з формування людської мови та експерименти з його мовною машиною. Основними частинами машини були барокамера для легенів, вібраційний язичок, який виконував роль голосових зв’язок та шкіряна трубка для дії голосових шляхів. Маніпулюючи формою шкіряної трубки, машина могла видавати різні голосні звуки. Його дослідження привели до теорії, що голосовий тракт, порожнина між голосовими зв’язками і губами, є головним місцем звукової артикуляції. До демонстрацій фон Кемпелена гортань, як правило, розглядалася як центр «виробництва» мови.

Приблизно в середині 1800-х рр. Чарльз Уітстоун сконструював свою знамениту версію мовної апаратури фон Кемпелена (Рис. 1.1). Це було дещо складніше і могло видавати голосні та більшість приголосних звуків.

Зв'язок між певним голосним звуком та геометрією голосових шляхів був виявлений Уіллісом у 1838 р. [3]. Він синтезував різні голосні з трубчастими резонаторами, як органні труби. Також було виявлено, що якість голосних залежить лише від довжини трубки, а не від її діаметра (див. Рис. 1.2).

Наприкінці 1800-х років Олександр Грехем Белл зі своїм батьком, натхненний мовною машиною Уітстона, побудував такий же мовний апарат. Дослідження та експерименти з механічними та напівелектричними аналогами голосової системи проводились до 1960-х рр.

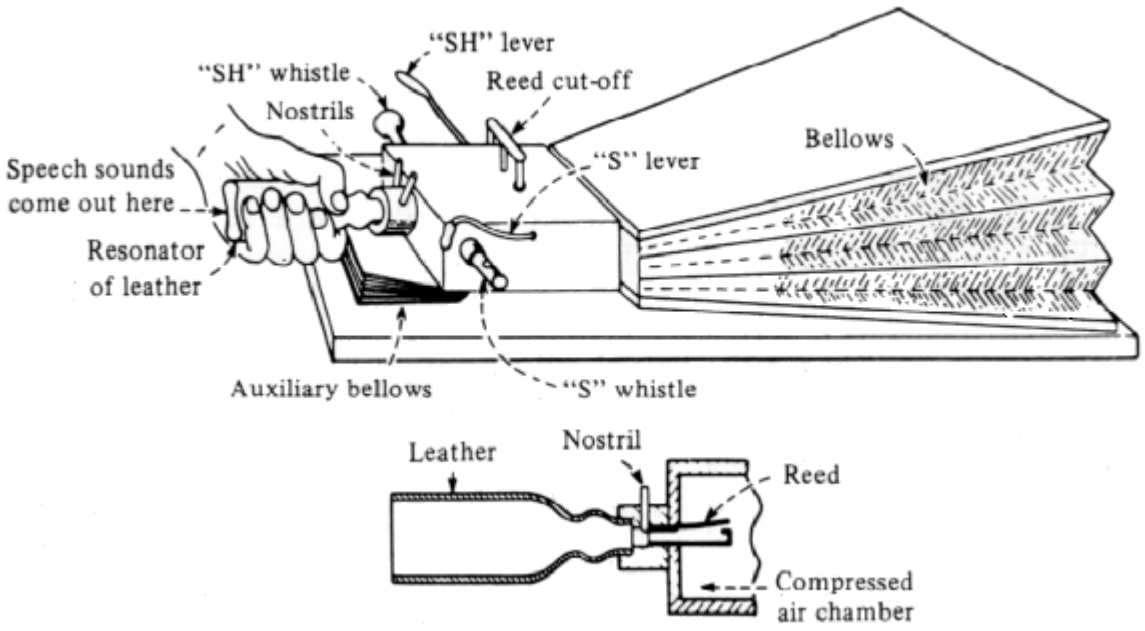


Рис. 1.2 Реконструкція Чарльза Уітстона [5]

1.2.2. Розробка електричних синтезаторів мовлення

Перший пристрій повного електричного синтезу був представлений Стюартом в 1922 році [4]. Синтезатор мав зумер в якості збудження та два резонансні контури для моделювання акустичних резонансів голосового тракту.

А першим пристроєм, що розглядався як синтезатор мови, був VODER (Voice Operating Demonstrator), представлений Гомером Дадлі на Всесвітній виставці в Нью-Йорку 1939 р. (Рис. 1.3)

Перший формантовий синтезатор, PAT (Parametric Artificial Talker), був представлений Уолтером Лоуренсом в 1953 р. [4]. Він складався з трьох паралельно з'єднаних електронних формантових резонаторів. [6]

1.3. Основні напрямки використання

1.3.1. Для осіб з обмеженими можливостями

Як вказано в джерелі [7], у всьому світі близько 2,2 млрд людей страждають порушенням ближнього або далекого зору. І більше 5% населення світу, або 430 мільйонів людей потребують реабілітації для вирішення проблеми втрати слуху, відомо з джерела [7]. Тому, мабуть, найважливішою та найкориснішою галуззю застосування в синтезі мовлення є засоби читання та спілкування для сліпих.

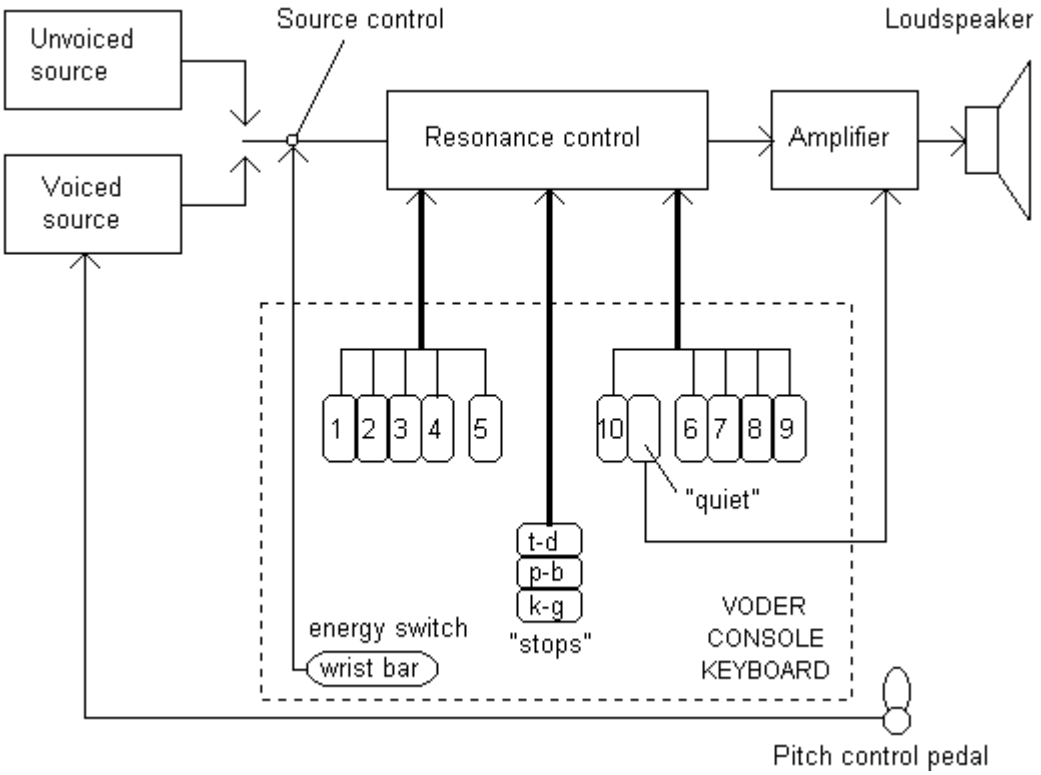


Рис. 1.3 Синтезатор мови VODER [4]

Відомо з джерела [8], що до синтезованого мовлення використовувались конкретні аудіокниги, де вміст книги читався на аудіокасету. Зрозуміло, що виготовлення такої розмовної копії будь-якої великої книги займає кілька місяців і є дуже дорогим.

Люди, які народилися глухими, не можуть навчитися говорити правильно, а люди з вадами слуху, як правило, мають проблеми з мовленням. Синтезована мова дає глухим та людям з обмеженими можливостями можливість спілкуватися з іншими людьми, які не розуміють мову жестів.

1.3.2. Освітні програми

Синтезоване мовлення може використовуватися також у багатьох навчальних ситуаціях. Комп’ютер із синтезатором мови може навчати 24 години на добу та 365 днів на рік. Він може бути запрограмований на спеціальні завдання, такі як навчання правопису та вимови для різних мов. Його також можна використовувати з інтерактивними освітніми програмами. Наприклад, як вказано в джерелі [9], у Китаї розроблено робота, який може виконувати роль вчителя. Робот запрограмований викладати низку предметів, включаючи природничі науки, англійську мову, тощо. Особливо у людей, які погано читають

(дислексики), синтез мовлення може бути дуже корисним, оскільки деякі діти можуть відчувати себе дуже незручно, коли їм повинен допомогти вчитель. Також навчитися писати та читати без розмовної допомоги практично неможливо. За умови належного комп'ютерного програмного забезпечення непідконтрольне навчання цих проблем легко і недорого організувати.

1.3.3. Інші програми та майбутні вказівки

Відомо з джерела [8], що синтез мови може бути використаний у всіх видах взаємодії людина-машина. Наприклад, у системах попередження та сигналізації синтезована мова може використовуватися для отримання більш точної інформації про поточну ситуацію. Синтезатор мови також може використовуватися для отримання деяких повідомлень на робочому столі з комп'ютера, таких як діяльність принтера або отримана електронна пошта.

Висновки

Перший розділ був присвячений огляду задачі синтезу мовлення, його історії. Також було розглянути основні напрямки використання. Найосновніший напрямок — це допомога людям з вадами слуху та зору. По світовій статистиці можна визначити, що це достатньо поширена проблема: людей з вадами зору в світі близько 2,2 млрд, а з вадами слуху більше 430 млн.

2. Огляд методів реалізації.

2.1. Способи синтезу мовлення

Найважливішими якостями системи синтезу мовлення є природність та зрозумілість. Природність описує, наскільки близький результат до людської мови, тоді як зрозумілість — це легкість розуміння результату. Ідеальний синтезатор мови є як природним, так і зрозумілим. Системи синтезу мовлення зазвичай намагаються максимізувати обидві характеристики.

Дві основні технології, що генерують синтетичні мовні форми, — це конкатенативний та формантний синтез. Кожна технологія має сильні та слабкі сторони, і передбачуване використання системи синтезу, як правило, визначає, який підхід застосовується. [10]

2.2. Конкатенативний або синтез компіляції (компілятивний)

Як вказано в джерелі [10], метод ґрунтується на складанні текстів із заздалегідь записаного "словника" елементів. Розмір елемента системи повинен бути не менше слова. Зазвичай запас елементів обмежується декількома сотнями слів, а зміст синтезованих текстів — обсягом словника. Основна проблема в компілятивному синтезі — обсяги пам'яті для зберігання словника. У зв'язку з цим використовуються різноманітні методи стиснення/кодування мовного сигналу. Цей метод синтезу мови широко використовується у повсякденному житті — як правило, в різних довідкових службах і техніці, що вимагає обладнання системами голосової відповіді, також для військової техніки.

2.2.1. Синтез з одиниць довільного розміру (Unit selection synthesis)

З опублікованої праці [11], відомо, що синтез *виділення одиниць* використовує великі бази даних записаної мови. Під час створення бази даних кожне записане висловлювання сегментується на деякі або всі наступні: окремі телефони, дзвінки, напівтелефони, склади, морфеми, слова, фрази та речення. Як правило, поділ на сегменти здійснюється за допомогою спеціально модифікованого розпізнавача мови, встановленого в режим "вимушеного

вирівнювання" з деякою ручною корекцією після цього, з використанням візуальних зображень, таких як форма сигналу та спектрограма. Потім створюється індекс одиниць у базі даних мовлення на основі параметрів сегментації та акустики, таких як основна частота (висота тону), тривалість, положення в складі та сусідні телефони. Під час виконання бажане цільове висловлювання створюється шляхом визначення найкращого ланцюжка одиниць-кандидатів з бази даних (вибір одиниць). Зазвичай цей процес досягається за допомогою спеціально зваженого «дерева» рішень.

Вибір одиниці забезпечує найбільшу природність, оскільки він застосовує лише невелику кількість цифрової обробки сигналу (Digital signal processing — DSP) до записаної мови. DSP часто робить записаний звук менш природним, хоча деякі системи використовують невелику кількість обробки сигналу в точці конкатенації, щоб згладити форму сигналу. Результати найкращих систем вибору одиниць часто неможливо відрізнити від справжніх людських голосів, особливо в контексті, на який була налаштована система TTS. Однак максимальна природність, як правило, вимагає, щоб бази даних мовлення з вибором одиниць були дуже великими, в деяких системах від гігабайта записаних даних, що представляє десятки годин мови.

2.2.2. Дифонний синтез

З опублікованої праці [11], відомо, що синтез дифонів використовує мінімальну базу даних мови, що містить усі дифони (звукові переходи), які зустрічаються у мові. Кількість дифонів залежить від фонотактики мови: наприклад, іспанська має близько 800, а німецька — близько 2500. При синтезі дифонів у базі мовлення міститься лише один приклад кожного дифона. Під час виконання цільова просодія речення накладається на ці мінімальні одиниці за допомогою таких технологій цифрової обробки сигналів, як лінійне передбачуване кодування, PSOLA або MBROLA або новіші методи, такі як модифікація висоти тону у вихідній області за допомогою дискретного косинусного перетворення. Синтез страждає від звукових збоїв конкатенативного синтезу та роботизованого характеру синтезу формантів, і має лише деякі

переваги будь-якого з підходів, крім малих розмірів. Таким чином, його використання в комерційних програмах зменшується, хоча воно продовжує використовуватися в дослідженнях, оскільки існує ряд вільно доступних програмних реалізацій.

2.2.3. Доменно-специфічний синтез

З опублікованої праці [11], відомо, що цей синтез є специфічний для предметної області і поєднує попередньо записані слова та фрази для створення повних висловлювань. Він використовується в додатках, де різноманітність текстів, які система видаватиме, обмежена певним доменом, наприклад, оголошення про розклад руху або звіти про погоду. Ця технологія дуже проста у впровадженні і тривалий час використовувалась у комерційних цілях на таких пристроях, як годинник, що говорить, та калькулятор. Рівень природності цих систем може бути дуже високим, оскільки різноманітність типів речень обмежена, і вони тісно відповідають просодії та інтонації оригінальних записів.

Оскільки ці системи обмежені словами та фразами у своїх базах даних, вони не універсальні та можуть синтезувати лише ті комбінації слів та фраз, за допомогою яких вони були попередньо запрограмовані. Однак поєднання слів у природно розмовній мові все одно може спричинити проблеми, якщо не враховувати багато варіацій.

2.2.4. Формантний синтез

Як вказано в джерелі [2], цей тип синтезу мовлення відомий як формант, оскільки форманти — це 3–5 ключових (резонансних) частот звуку, які голосовий апарат людини генерує та поєднує, щоб створити звук мови або співу. На відміну від синтезаторів мови, які використовують конкатенацію, яка обмежується перестановкою попередньо записаних звуків, формантові синтезатори мови можуть говорити абсолютно все — навіть слова, які не існують, або іноземні слова, з якими вони ніколи не стикалися. Це робить формантові синтезатори хорошим вибором для супутникових (навігаційних) комп’ютерів GPS, які повинні мати можливість читати тисячі різних (і часто незвичних) топонімів, які важко запам’ятати. [10]

2.2.5. Артикуляційний

Як вказано в джерелі [2], це найскладніший підхід до утворення звуків, він «змушує» комп'ютери говорити, моделюючи дивовижно складний голосовий апарат людини. Теоретично це повинно дати найбільш реалістичний людський голос з усіх методів. Незважаючи на те, що численні дослідники експериментували з імітацією людського голосового апарату, артикуляційний синтез все ще є найменш дослідженим методом, здебільшого через його складність. Найбільш складною формою артикуляційного синтезу було б розробити робота, що говорить «головою», з рухомим ротом, який видає звук, подібний до людини, поєднуючи за необхідності механічні та електронні компоненти.

2.2.6. Синтез на основі HMM (Hidden Markov Model)

Як вказано в джерелі [12], це метод синтезу, заснований на прихованих марковських моделях, який також називають статистичним параметричним синтезом. У цій системі частотний спектр (голосовий тракт), основна частота (джерело голосу) і тривалість (просодія) мови одночасно моделюються HMM. Форми мовних сигналів генеруються з самих HMM на основі критерію максимальної вірогідності.

2.2.7. Синусоїдальний синтез

Як вказано в джерелі [12] синтез синусоїд — це техніка синтезу мовлення шляхом заміни формантів на чисті тонові свистки.

2.3. Різниця між формантним та конкатенативним синтезом

Відомо з джерела [2], що теоретично, *формантові* синтезатори можуть легко переключитися з чоловічого на жіночий голос (приблизно вдвічі збільшивши частоту) або на дитячий голос (збільшити втричі), і вони можуть розмовляти будь-якою мовою. На практиці синтезатори конкатенації тепер використовують величезні бібліотеки звуків, тому вони теж можуть сказати майже все. Більш очевидна різниця полягає в тому, що синтезатори конкатенації

звучать набагато природніше, ніж формантні, які, як і раніше, звучать відносно штучно і роботизовано.

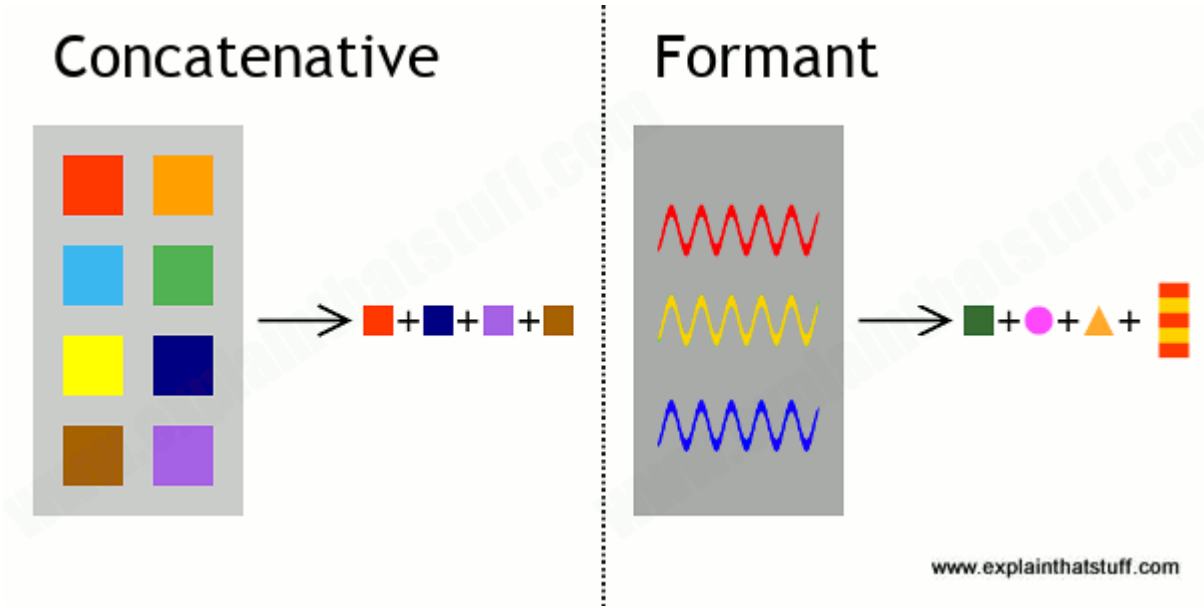


Рис. 2.1 Схематичне уявлення конкатенативного і формантового синтезу [2]

На Рис. 2.1 зліва: конкатенативний синтезатор створює мовлення з попередньо збережених фрагментів. Справа: Як і музичний синтезатор, формантовий синтезатор використовує генератори частоти для створення будь-якого звуку.

2.4. Фізичний принцип системи синтезу мови

Наприклад, у вас є параграф письмового тексту, який ви хочете, щоб ваш комп'ютер вимовляв вголос. Як він перетворює написаний текст в слова, які ви дійсно чуєте? Отже узагальнена схема представлена на Рис. 2.2, тут задіяні три етапи, які я назву:

1. перетворенням тексту в слова;
2. слова в фонеми;
3. фонеми в звук, як вказано в джерелі [2].

2.4.1. Текст в слова

Читання слів звучить легко, але якщо ви коли-небудь чули як маленька дитина читала книгу, яка для неї була важкою, ви знатимете, що це не так тривіально, як здається. Основна проблема полягає в тому, що письмовий текст є

неоднозначним: одна і та ж письмова інформація часто може означати більше, ніж одне значення, і, як правило, вам потрібно зрозуміти сенс або зробити обгрунтовану здогадку, щоб правильно прочитати. Отже, початковий етап синтезу мовлення, який зазвичай називають *попередньою обробкою* або *нормалізацією*, полягає у зменшенні двозначності: мова йде про звуження багатьох різних способів, як ви могли прочитати фрагмент тексту, до найбільш підходящого.

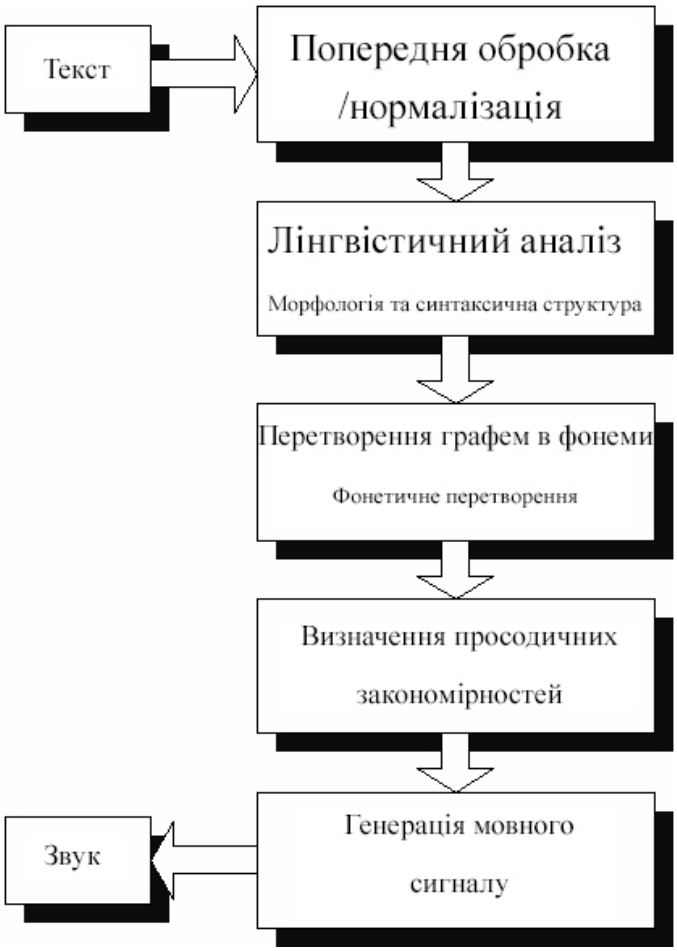


Рис. 2.2 Схема фізичного принципу синтезу мовлення [13]

Попередня обробка включає перегляд тексту та його очищення, щоб комп’ютер робив менше помилок, коли фактично читає слова вголос. Такі речі, як цифри, дати, час, абревіатури та спеціальні символи (символи валют тощо) потрібно перетворити на слова — і це складніше, ніж звучить. Число 1843 може означати кількість предметів ("тисяча вісімсот сорок три"), рік чи час ("вісімнадцять сорок три") або комбінацію навісних замків ("вісім вісім чотири три"), кожен з який читається дещо інакше. Хоча люди слідують відчуттю написаного і таким чином з’ясовують вимову, комп’ютери, як правило, не в силах

це зробити, тому їм доводиться використовувати статистичні прийоми ймовірності (зазвичай прихована марковська модель(ПММ), англ. Hidden Markov Model) або нейронні мережі (комп’ютерні програми, структуровані як масиви клітин мозку, які вчаться розпізнавати закономірності), щоб натомість отримати найімовірнішу вимову. Отже, якщо слово "рік" зустрічається в тому самому реченні, що і "1843", можливо, буде розумно вгадати, що це дата. Якби перед цифрами стояла десяткова крапка (".843"), їх потрібно було б читати інакше як "вісім чотири три".

Попередня обробка також повинна вирішувати омографи, слова, що вимовляються по-різному відповідно до їх значення(наприклад, слово орган), відомо з джерела [2].

2.4.2. Слова до фонем

Як вказано в джерелі [2], з’ясувавши слова, які потрібно вимовити, синтезатор мови тепер повинен генерувати мовні звуки, з яких складаються ці слова. Теоретично це достатньо просто: усе, що потрібно комп’ютеру — це величезний алфавітний список слів та деталей, як вимовляти кожне з них (так само, як це можна знайти у типовому словнику, де вимова вказана до або після визначення). Для кожного слова нам потрібен список фонем, що складають його звук.

Що таке фонеми?

Грубо кажучи, фонеми для розмовної мови — це те ж саме, що і букви. Для письмової мови: вони є атомами розмовного звуку — звукових компонентів, з яких ви можете зробити будь-яке вимовлене слово, яке вам подобається.

Теоретично, якщо комп’ютер має словник слів і фонем, все, що йому потрібно зробити, щоб прочитати слово, — це знайти його у списку, а потім прочитати відповідні фонеми, чи не так? На практиці це складніше, ніж здається. Як може продемонструвати будь-який хороший актор, одне речення може бути прочитане різними способами відповідно до значення тексту, людини, яка говорить, та емоцій, які він хоче передати (у лінгвістиці ця ідея відома як просодія, і вона одна з найскладніших проблем для вирішення синтезаторів мови).

Альтернативний підхід передбачає розбиття письмових слів на їхні графеми (одиниці письмових компонентів, як правило, складаються з окремих букв або складів, що складають слово), а потім генерує фонеми, які відповідають їм за допомогою набору простих правил. Це трохи схоже на те, що дитина намагається прочитати слова, з якими вона ніколи раніше не стикалася. Перевага цього полягає в тому, що комп'ютер може зробити розумну спробу прочитати будь-яке слово, незалежно від того, чи це справжнє слово, що зберігається у словнику, іноземне слово чи незвичне ім'я чи технічний термін. Недоліком є те, що такі мови, як англійська, мають велику кількість неправильних слів, які вимовляються зовсім по-іншому від того, як вони пишуться.

2.4.3. Фонеми в звук

Список фонем, які комп'ютер читає вголос, перетворюючи текст на мову, можна отримати трьома різними підходами. Перший — використовувати записи людей, які вимовляють фонеми, другий — комп'ютер сам створює фонеми, генеруючи основні звукові частоти (трішки схоже, як музичний синтезатор), а третій підхід — імітувати механізм людського голосу, як вказано в джерелі [2].

2.5. Порівняльна характеристика деяких сучасних синтезаторів мовлення

Вибираючи, слід оцінити кілька речей: вартість (є безкоштовні послуги, і, як правило, послуги TSS оплачуються за літеру), якість виведення (якщо вам потрібен справді реалістичний голос, схожий на людину), складність його налаштування та інтегруючи його, за потреби, з іншими API або системами, послугами, які він пропонує (наприклад, скільки мов та розмір текстів він може обробити).

2.5.1. Google Text To Speech

Програма розроблена Google© для операційної системи Android™. Функція синтезу мовлення може використовуватися такими програмами, як Google Play Books, для читання книг вголос, Google Translate для озвучування перекладів та вимови слів, Google Talkback та іншими програмами, що базуються на

доступності голосових відгуків. Це найпопулярніший API для синтезу мовлення зі всіх доступних.

Плюси

- Великий список мов, які підтримуються: https://cloud.google.com/text-to-speech/docs/voices
- Стійкий до побічних шумів в звуці
- Можлива транскрипція тексту

Мінуси

- Платно: https://cloud.google.com/text-to-speech/pricing
- Інколи є затримка відтворення звуку 3-4сек

2.5.2. IBM Watson Text To Speech

Платформа, що використовується для створення природного звучання та плавної якості голосу. Це програмне забезпечення переважно використовують малі та середні компанії.

Плюси

- API може визначати тон речення. Наприклад, якщо це буде питання, то буде відповідний акцент на кінці речення(працює тільки на англійській мові)
- розпізнавання деяких скорочень(наприклад, символи валют)

Мінуси

- Маленький вибір мов
- Платно: https://cloud.ibm.com/catalog/services/text-to-speech

2.5.3. Microsoft Azure Bing Speech API

Це програмне забезпечення когнітивних мовних послуг, розміщене у хмарі, яке пропонує можливість перетворення мови в текст, а також перетворення тексту в мову.

Плюси

- Фільтрація ненормативної лексики
- Нормалізація тексту в реальному часі

- Також підтримує різні умови розмови(бесіда, диктування і т.д.)

Мінуси

- Відтворення інколи може бути не точним
- Платно: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/speech-services/

2.5.4. Amazon Polly

Сервіс, що дозволяє перетворювати текст в мову як в пакетному режимі, так і в режимі реального часу. Він також є частиною інфраструктури Amazon Web Services.

Плюси

- Великий вибір функціоналу
- Простота використання

Мінуси

- Малий вибір мов та голосів для відтворення
- Платно: https://aws.amazon.com/polly/pricing/?nc=sn&loc=4

2.5.5. Web Speech API

Це стандарт у веб-програмуванні. API має на меті дозволити веб-розробникам можливість надавати у браузері функції перетворення мови в текст і перетворення тексту в мову.

Плюси

- Конфіденційність користувача(наприклад, потрібне надання доступу до мікрофону)
- Великий список мов, які підтримуються
- Безкоштовно

Мінуси

- Немає української мови
- Інколи є затримка у відтворенні 2-3 секунди

Висновок

У другому розділі визначено основні види сучасних синтезаторів мовлення та їх аналіз. Також було описано фізичний принцип системи синтезу мовлення.

Після проведеного порівняння різних API, для розробки сайту було обрано Web Speech API. API (Application Programming Interface) — прикладний програмний інтерфейс.

3. Розробка сайту синтезатора мовлення

3.1. Вибір мови програмування та API для розробки

Обрано таку мову програмування, як JavaScript, а саме, було використано бібліотеку для створення користувацьких інтерфейсів — React.

Чому саме такий вибір?

1. На даний момент, JS вважається одною з найкращих та найбільш розвиваючих мов програмування;
2. JavaScript насамперед відома як скриптова мова для веб-сторінок.

У попередньому розділі виділено декілька найпопулярніших API для синтезу мовлення. На розробленому сайті використано – Web Speech API.

3.2. Концепції та викоритсання Web Speech

Всі терміни використані відповідно до [14].

Web Speech API дозволяє веб-додаткам керувати голосовими даними. Існує два компонента до цього API:

- Розпізнавання голосу. Доступ забезпечується через SpeechRecognition інтерфейс, який в свою чергу забезпечує можливість розпізнавати текст з вхідного аудіо потоку (зазвичай через пристрій розпізнавання телефону за замовчуванням). Скориставшись конструктором інтерфейсу можна створити новий SpeechRecognition об'єкт, у якого є ряд подій для виявлення початку мовлення через мікрофон пристрою. SpeechGrammar інтерфейс — надає контейнер для певного набору граматики, який ваший застосунок буде використовувати. Граматика визначається за допомогою JSpeech Grammar Format (JSGF.) Відомо з джерела [15], що формат граматики JSpeech (JSGF) — це незалежне від платформи та від постачальника текстове представлення граматик для використання в розпізнаванні мови. Граматики використовуються розпізнавачами мови, щоб визначити, що розпізнавач повинен слухати, і тому описують висловлювання, які може сказати користувач. JSGF застосовує стиль та

конвенції мови програмування JavaTM на додаток до використання традиційних граматичних позначень.

- Доступ до синтезу мови здійснюється за допомогою SpeechSynthesis (en-US) інтерфейсу, компонент text-to-speech дозволяє додаткам прочитати свій текстовий контент (зазвичай через дефолтний синтезатор мови пристрою). У SpeechSynthesisVoice (en-US) об'єктах є різні типи голосу, і різним частинам тексту можна призначати SpeechSynthesisUtterance об'єкти. Можна почати відтворення передавши їх методу SpeechSynthesis.speak () (en-US).

3.3. Існуючі інтерфейси Web Speech API

Що таке інтерфейс?

Як вказано в джерелі [16], інтерфейс — це сукупність методів і правил взаємодії елементів системи. Іншими словами, інтерфейс визначає як елементи будуть взаємодіяти між собою.

3.3.1. Розпізнавання мовлення

Всі терміни використані відповідно до [14].

- SpeechRecognition

  Інтерфейс контролера для служби розпізнавання; це також обробляє SpeechRecognitionEvent (en-US), відправлений зі служби розпізнавання
- SpeechRecognitionAlternative(en-US)

  Представляє одне слово яке було розпізнано службою розпізнавання голосу.
- SpeechRecognitionError(en-US)

  Представляє повідомлення про помилку із служби розпізнавання голосу.
- SpeechRecognitionEvent(en-US)

  Об'єкт події результату містить усі дані, пов'язані з тимчасовим або остаточним результатом розпізнавання мови.
- SpeechGrammar

  Слова або шаблони слів, які ми хочемо щоб служба розпізнала.

- SpeechGrammarList(en-US)

  Представляє список об’єктів SpeechGrammar

- SpeechRecognitionResult(en-US)

  Представляє один розпізнаний збіг, який може містити кілька об'єктів SpeechRecognitionAlternative (en-US).

- SpeechRecognitionResultList(en-US)

  Представляє список об'єктів SpeechRecognitionResult (en-US) або окремого, якщо результати фіксуються в безперервному режимі (en-US).

3.3.2. Синтез мовлення

Всі терміни використані відповідно до [14].

- SpeechSynthesis(en-US)

  Інтерфейс контролера для мовної служби; це може бути використано для отримання інформації про синтезатор голосів, доступних на пристрої, запуску та паузи мовлення та інших команд.

- SpeechSynthesisErrorEvent(en-US)

  Містить інформацію про будь-які помилки, які виникають під час обробки об’єктів SpeechSynthesisUtterance у мовленнєвій службі.

- SpeechSynthesisEvent(en-US)

  Містить інформацію про поточний стан об’єктів SpeechSynthesisUtterance, які були оброблені в мовленнєвій службі.

- SpeechSynthesisUtterance

  Представляє мовний запит. Він містить вміст, який повинна прочитати мовна служба, та інформацію про те, як його читати (наприклад, мову, швидкість та висоту тону).

- SpeechSynthesisVoice(en-US)

  Представляє голос, який підтримує система. Кожен SpeechSynthesisVoice має власну службу відносної мови, що включає інформацію про мову, ім'я та URI.

- Window.speechSynthesis(en-US)

  Виділяється як частина інтерфейсу [NoInterfaceObject], який називається SpeechSynthesisGetter, і реалізований об'єктом Window, властивість speechSynthesis забезпечує доступ до контролера SpeechSynthesis (en-US), а отже, і точку входу в функціонал синтезу мовлення. [14]

3.4. Створення проекту в WebStorm

Згідно до джерела [17], JetBrains WebStorm© 2020.1.1 — це інтегроване середовище розробки для HTML, CSS, JavaScript від компанії JetBrains©. WebStorm забезпечує автодоповнення, аналіз коду на льоту, навігацію по коду, рефакторинг та інтеграцію з системами управління версіями. Важливою перевагою інтегрованого середовища розробки WebStorm© є робота з проектами (у тому числі, рефакторинг коду JavaScript, що міститься в різних файлах і теках проекту, а також вкладеного в HTML). Підтримується множинна вкладеність (коли в документ на HTML вкладений скрипт на Javascript, в який вкладено інший код HTML, всередині якого вкладений Javascript) — в таких конструкціях підтримується коректний рефакторинг.

Щоб почати роботу у середовищі WebStorm© потрібно створити новий проект. Для цього необхідно зробити наступні кроки:

1. На екрані привітання WebStorm© обираємо пункт "Create New Project". Якщо потрібно відкрити готовий проект, вибираємо "Open" і вказуємо шлях до папки з проектом.
2. Якщо все ж таки метою є створення нового проекту, можна вказувати потрібні налаштування(є готові шаблони) або створити повністю пустий проект.

3.5. Створення програми

3.5.1. Create React App

Для початку потрібно установити пакет **create-react-app**. Це інструмент для швидкого старту React-додатків. За допомогою цього пакету ви не витрачаєте час на настройку більшості потрібних інструментів. Вони заздалегідь налаштовані і

заховані, так що зразу можна сфокусуватися на коді програми. Для цього в термінали потрібно прописати наступну команду: **npx create-react-app <<назва проекту>>**

3.5.2. Структура проекту

Після установки пакета create-react-app, буде створений такий проект:

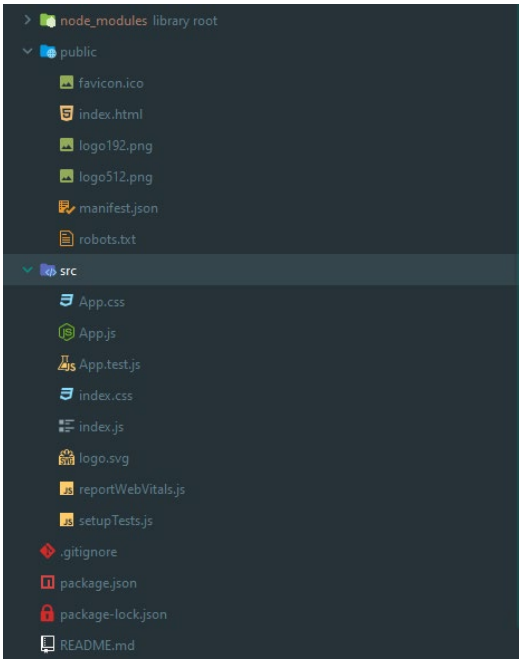


Рис. 3.1. Папки та файли створеного проекту

Детальний опис основних елементів, написаний на основні джерела [18].

- node_modules / містить всі зовнішні бібліотеки JavaScript використовувані додатком.
- Директорія public / містить базові файли HTML, JSON і зображень. Це кореневі ресурси проекту.
- В директорії src / міститься код React JavaScript для проекту. В основному працюють саме з цією директорією.
- src / index.js — саме цей файл виконує розгортання React-додатку. В нього вкладають головний компонент App, який представляє собою React-додаток.
- src / App.js – це код основного компонента на сайті, всі інші будуть приєднуватись саме до нього.

- Файл .gitignore містить кілька директорій і файлів за замовчуванням, які система контролю вихідного коду git ігноруватиме, в тому числі директорію node_modules. Ігноровані елементи — це великі директорії або файли журналу, які не потрібні при контролі вихідного коду.
- README.md — це файл розмітки, що містить багато корисної інформації про програму Create React App, в тому числі огляд команд і посилання на розширені опції конфігурації.
- Файл package-lock.json використовується npm для перевірки точної відповідності версій пакетів. За допомогою цього файлу можна бути впевненим, що якщо хтось інший встановить даний проект, у нього будуть ідентичні залежності.
- Файл package.json містить метадані про даний проект, включаючи заголовок, номер версії та залежності. Також він містить скрипти, які можна використовувати для запуску проекту.
- Крім цих файлів, в папці src лежать стилі, тести, кілька службових скриптів і svg файл логотипу.

3.5.3. Компоненти

React значно полегшує створення інтерфейсів завдяки розподіленню кожної сторінки на невеликі фрагменти. Ці фрагменти називаються компонентами.

Компонент React — це ділянка коду, яка представляє частину веб-сторінки. Кожен компонент — це JavaScript-функція, яка повертає шматок коду, що представляє фрагмент сторінки, як сказано в джерелі [19]

Компоненти проекту (всі вони зберігаються в директорії components) представлені на Рис. 3.3.

Короткий опис кожного компоненту:

AppBar — використовується для структурування документа. Це верхня панель, яку можна побачити на веб-сайті. AppBar це оболонка для інших компонентів. У нашому випадку всередині AppBar розміщується компонент Navigation. Це зроблено для семантичної верстки.

Рис. 3.2 Приклад розбивання сторінки на компоненти(кожен виділений фрагмент вважається компонентом) [19]

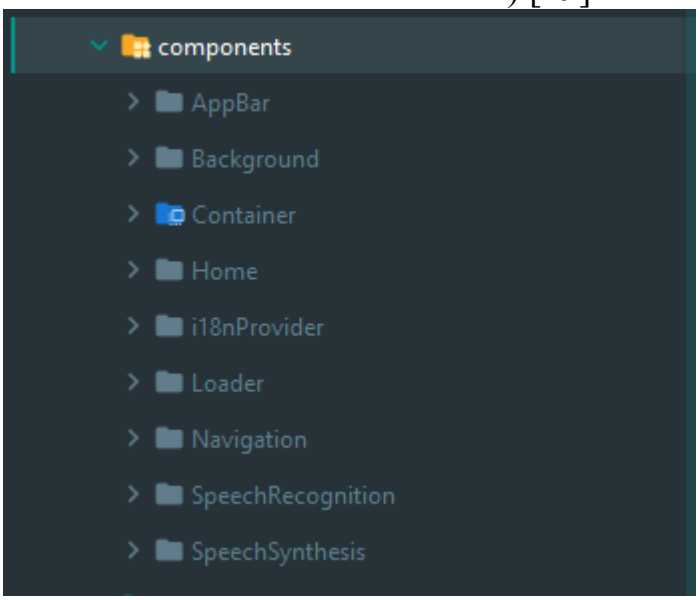


Рис. 3.3 Компоненти проекту

Що таке семантична верстка і чому це важливо?

Семантична верстка — підхід до розмітки, який спирається не на зміст сайту, а на смислове призначення кожного блоку і логічну структуру документа. Це важливо тому, що зрячі користувачі можуть без проблем з першого погляду зрозуміти, де яка частина сторінки знаходиться — де заголовок, списки або зображення. Для незрячих або частково незрячих все складніше. Основний інструмент для перегляду сайтів не браузер, який відмальовує сторінку, а скрінрідер, який читає текст зі сторінки вголос. [20]

Navigation – це компонент навігації. (зверху дві кнопки «Синтез мовлення» та «Розпізнавання мовлення»).

Container — це «коробка» для інших компонентів, він потрібний для стилізації і також для семантичної верстки.

Background – це компонент, який відповідає за задній фон на сайті.

Home – це «привітальна» сторінка. На ній можна обрати мову інтерфейсу.

Loader – це компонент, який відповідає за загрузку. Коли ми переключаємось на один з пунктів навігації, йде загрузка контенту, і в цей час появляється Loader.

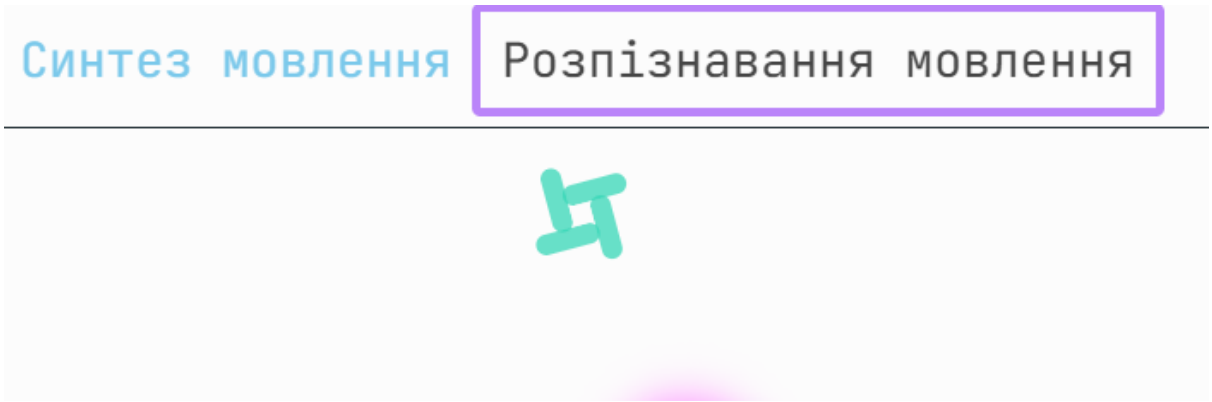


Рис. 3.4 Loader

i18nProvider — компонент, який відповідає за зміну мови на сайті.

Як це відбувається?

Було використано бібліотеку react-intl [21]. Вона надає багато можливостей і однієї з них є — інтернаціоналізація. React-intl дозволяє створити словник слів для кожної конкретної мови, а потім всередині компонентів замість тексту писати змінні з цих словників. При ініціалізації компонента вибираємо мову, а замість змінної автоматично підставляється слово з потрібного словника.

Так як на сайті використано 3 мови — українська, російська та англійська, було створено 3 окремих сховища(«словника») – uk-UA.JS, ru.js, en-US.js.

Необхідно відзначити, що назви ключів для тексту повинні збігатися у всіх словниках. І також окремо створена функція(файл translate.js), яка підставляє потрібну мову.

SpeechRecognition – компонент, який відповідає за розпізнавання мовлення.

SpeechSynthesis – компонент, який відповідає за синтез мовлення.

Написання коду в такому стилі, це дуже зручно, тому що кожен компонент можна перевикористовувати n-кількість раз, як і в цьому проекті так і в інших. Такі фрагменти, як: AppBar, Container, Navigation, Home(Main) — це стандартні компоненти, які присутні при кожній розробці веб-сайту.

3.5.4. Бібліотека react-speech-kit

Опис бібліотеки взято з [22]. Це інструмент для роботи react-додатку з Web Speech API.

Бібліотека має два хуки: useSpeechSynthesis, useSpeechRecognition. Саме за допомогою цих хуків, були написані такі компоненти, як SpeechSynthesis і SpeechRecognition.

Хуки — це нова технологія в React, їх використовують для роботи з локальним станом або методами життєвого циклу. Простими словами, це те, що дозволяє обробляти кліки, написання речення та багато інших інтерактивностей.

Всі терміни використані відповідно до [22]

useSpeechSynthesis побудований на основі SpeechSynthesis інтерфейсу, про який йшлась мова в 2 розділі.

Властивості:

- SpeechSynthesis.paused

  Логічне значення, яке повертає істину, якщо об'єкт SpeechSynthesis знаходиться в призупиненому стані.

- SpeechSynthesis.pending

  Логічне значення, яке повертає істину, якщо черга висловлювання містить ще невимовлені висловлювання.

- SpeechSynthesis.speaking

  Логічне значення, яке повертає істину, якщо в даний час звучить висловлювання, навіть якщо SpeechSynthesis знаходиться в призупиненому стані

```
const { speak, cancel, speaking, supported, voices } = useSpeechSynthesis({
  onEnd,
});
```

Рис. 3.5 Приклад useSpeechSynthesis в коді

Аргументи:

- onEnd — функція, вона є необов’язковим параметром.
  Викликається, коли SpeechSynthesis закінчує читати текст або скасовується.
- useSpeechSynthesis повертає об'єкт, який містить наступне:
  speak – функція з аргументами
  function({ text: string, voice: SpeechSynthesisVoice })
  Викликається, щоб браузер прочитав текст. Можна змінити голос, передавши доступний SpeechSynthesisVoice (із масиву голосів).
- cancel — функція
  Викликається, щоб SpeechSynthesis припинив читання.
- speaking — логічне значення
  Істина, коли SpeechSynthesis говорить.
- supported – логічне значення
  Істина, якщо браузер підтримує SpeechSynthesis.
- voices — масив
  Масив доступних голосів [SpeechSynthesisVoice], які можна передати функції speak. Приклад голосу SpeechSynthesisVoice має такі властивості.
  {
  default: true,
  lang: “en-AU”,
  localServiceL true,
  name: “Joe”,
  voiceURI: “Joe”
  }

useSpeechRecognition побудований на основі SpeechRecognition інтерфейсу, про який йшла мова в 2 розділі.

Властивості:

- SpeechRecognition.grammars

  Повертає та встановлює колекцію об'єктів SpeechGrammar – це граматичні правила, які будуть зрозумілі поточним SpeechRecognition.

- SpeechRecognition.lang

  Повертає та встановлює мову поточного розпізнавання.

- SpeechRecognition.continuous

  Визначає, чи повертаються результати для кожного розпізнавання.

- SpeechRecognition.interimResults

  Контролює, чи слід повертати проміжні результати (true) чи ні (false). Проміжні результати — це результати, які ще не є остаточними.

- SpeechRecognition.maxAlternatives

  Встановлює максимальну кількість альтернатив SpeechRecognitionAlternatives, наданих на результат.

- SpeechRecognition.serviceURI

  Вказує місце розташування служби розпізнавання мови, яка використовується поточним SpeechRecognition для обробки фактичного розпізнавання. За замовчуванням використовується мовна послуга агента користувача за замовчуванням.

У коді хук виглядає ось так:

```
const { listen, listening, stop, supported } = useSpeechRecognition({
  onResult,
  onEnd,
  onError,
});
```

Рис. 3.6 Приклад useSpeechRecognition в коді

Аргументи:

- onEnd – функція

  Викликається, коли SpeechRecognition перестає слухати.

- onResult – функція, яка приємає аргументом рядок function(string)

  Викликається, коли SpeechRecognition має результат. Він викликається за допомогою рядка, що містить розшифровку розпізнаної мови.

useSpeechRecognition повертає об’єкт, який містить listen – функція з аргументами function({ interimResults: boolean, lang: string })

Викликається, щоб браузер почав слухати введення. Кожен раз, коли він обробляє результат, він пересилає стенограму до наданої функції onResult. Можна змінити поведінку, передавши такі аргументи:

- lang – рядок

  Мова, за допомогою якої SpeechRecognition намагатиметься інтерпретувати введення.

- interimResults – логічне значення(по замовчуванню true)

  SpeechRecognition може надавати результати в режимі реального часу, оскільки намагається з’ясувати найкращий збіг для вхідних даних. Можна встановити значення false, якщо ви хочете лише остаточний результат.

- stop — функція

  Викликається, щоб SpeechRecognition припинив слухати. Це також викликає функцію onEnd.

- listening – логічне значення

  Істина, коли SpeechRecognition слухає.

- supported – логічне значення

  Істина, якщо браузер підтримує SpeechRecognition.

3.6. Запуск сервера

Для запуску сервера є два варіанти: викласти його на хостинг або відкрити термінал і ввести команду **npm start**

Висновок

В ході виконання даної роботи було проведено дослідження у сфері синтезу мовлення, розглянуто методи синтезу мовлення, а також основні сфери використання. Проведено аналіз сучасних синтезаторів мовлення з перевагами і недоліками.

За допомогою Web Speech API було розроблено веб-застосунок з можливістю перетворення тексту в звук та навпаки. Представлений великий вибір мов для обох варіантів.

Ця робота представляється як, рішення для інклюзивних освітніх програм. З не вирішених проблем залишається відсутність перетворення тексту в звук українською мовою. Проте Web Speech API зараз стрімко розвивається, тому цієї проблеми скоро не буде. Також потрібно зазначити, що плюсом даного сайту є те що він безкоштовний, і присутній вибір інтерфейсу з 3 мов: українська, російська, англійська.

Посилання на створений веб-сайт: https://o-text-to-speech.netlify.app/

Перелік посилань

[1] "Deep Learning Based Speech Synthesis," [Online]. Available: https://encyclopedia.pub/2975. [Accessed 10 03 2021].

[2] C. Woodford, “Speech synthesizer,” [Online]. Available: https://www.explainthatstuff.com/how-speech-synthesis-works.html. [Accessed 09 03 2021].

[3] M. R. Schroeder, "A brief history of synthetic speech," *Speech Communication,* vol. 13, no. 1-2, pp. 231-237, October 1993.

[4] D. H. Klatt, "Review of text-to-speech conversion for English," *The Journal of the Acoustical Society of America,* vol. 82, no. 3, p. 737, May 1987.

[5] J. L. Flanagan, Speech Analysis Synthesis and Perception, 2 ed., Springer-Verlag Berlin Heidelberg, 1972.

[6] “History and Development of Speech Synthesis,” [Online]. Available: http://research.spa.aalto.fi/publications/theses/lemmetty_mst/chap2.html. [Accessed 14 03 2021].

[7] WHO, "Deafness and hearing loss," who.int, [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss. [Accessed 15 05 2021].

[8] “Application of Synthesis Speech,” [Online]. Available: http://research.spa.aalto.fi/publications/theses/lemmetty_mst/chap6.html. [Accessed 09 03 2021].

[9] tribuneonlineng.com, “China Now Using Robots To Teach Pupils In Schools Due To Shortage Of Teachers,” [Online]. Available: https://tribuneonlineng.com/china-now-using-robots-to-teach-pupils-in-schools-due-to-shortage-of-teachers-2/. [Accessed 15 05 2021].

[10] Wikimedia Foundation, Inc., «Синтезатор мовлення,» [Онлайновий]. Available: https://uk.wikipedia.org/wiki/Синтез_мовлення. [Дата звернення: 08 03 2021].

[11] International Journal of Applied Information Systems, “Design and Implementation of Text To Speech Conversion for Visually Impaired People,” *IJAIS Journal,* 2014.

[12] Wikimedia Foundation, Inc., “Speech synthesis,” [Online]. Available: https://en.wikipedia.org/wiki/Speech_synthesis. [Accessed 08 03 2021].

[13] J. P. Teixeira, "The Text-to-Speech System Architecture," researchgate.net, [Online]. Available: https://www.researchgate.net/figure/The-Text-to-Speech-System-Architecture_fig2_221522994. [Accessed 15 03 2021].

[14] M. a. i. contributors, «Web Speech API,» [В Интернете]. Available: https://developer.mozilla.org/ru/docs/Web/API/Web_Speech_API. [Дата обращения: 09 03 2021].

[15] Sun Microsystems, Inc., “JSpeech Grammar Format,” w3.org, [Online]. Available: https://www.w3.org/TR/jsgf/. [Accessed 02 05 2021].

[16] «ИНТЕРФЕЙС,» [В Интернете]. Available: https://www.russianpromo.ru/wiki/interface/. [Дата обращения: 24 04 2021].

[17] Wikimedia Foundation, Inc., «WebStorm,» [Онлайновий]. Available: https://uk.wikipedia.org/wiki/WebStorm. [Дата звернення: 24 04 2021].

[18] J. Morgan, «Создание и настройка проекта React с помощью приложения Create React App,» [В Интернете]. Available: https://www.digitalocean.com/community/tutorials/how-to-set-up-a-react-project-with-create-react-app-ru. [Дата обращения: 04 25 2021].

[19] Stepik, «React,» stepik.org, [В Интернете]. Available: https://stepik.org/lesson/370446/step/1. [Дата обращения: 26 04 2021].

[20] Интерактивные обучающие технологии, «Что такое семантическая вёрстка и зачем она нужна,» [В Интернете]. Available: https://htmlacademy.ru/blog/boost/frontend/semantics. [Дата обращения: 26 04 2021].

[21] npm, Inc, “React Intl,” npmjs.com, [Online]. Available: https://www.npmjs.com/package/react-intl. [Accessed 24 04 2021].

[22] npm, Inc, «react-speech-kit,» npmjs.com, [В Интернете]. Available: https://www.npmjs.com/package/react-speech-kit. [Дата обращения: 15 03 2021].

[23] WHO , "Blindness and vision impairment," who.int, [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-

impairment. [Accessed 15 05 2021].

ДОДАТКИ

Додаток А. Лістинг програми

Весь код програми знаходиться на електронному ресурсі:

https://github.com/olyaonysko/text-to-speech