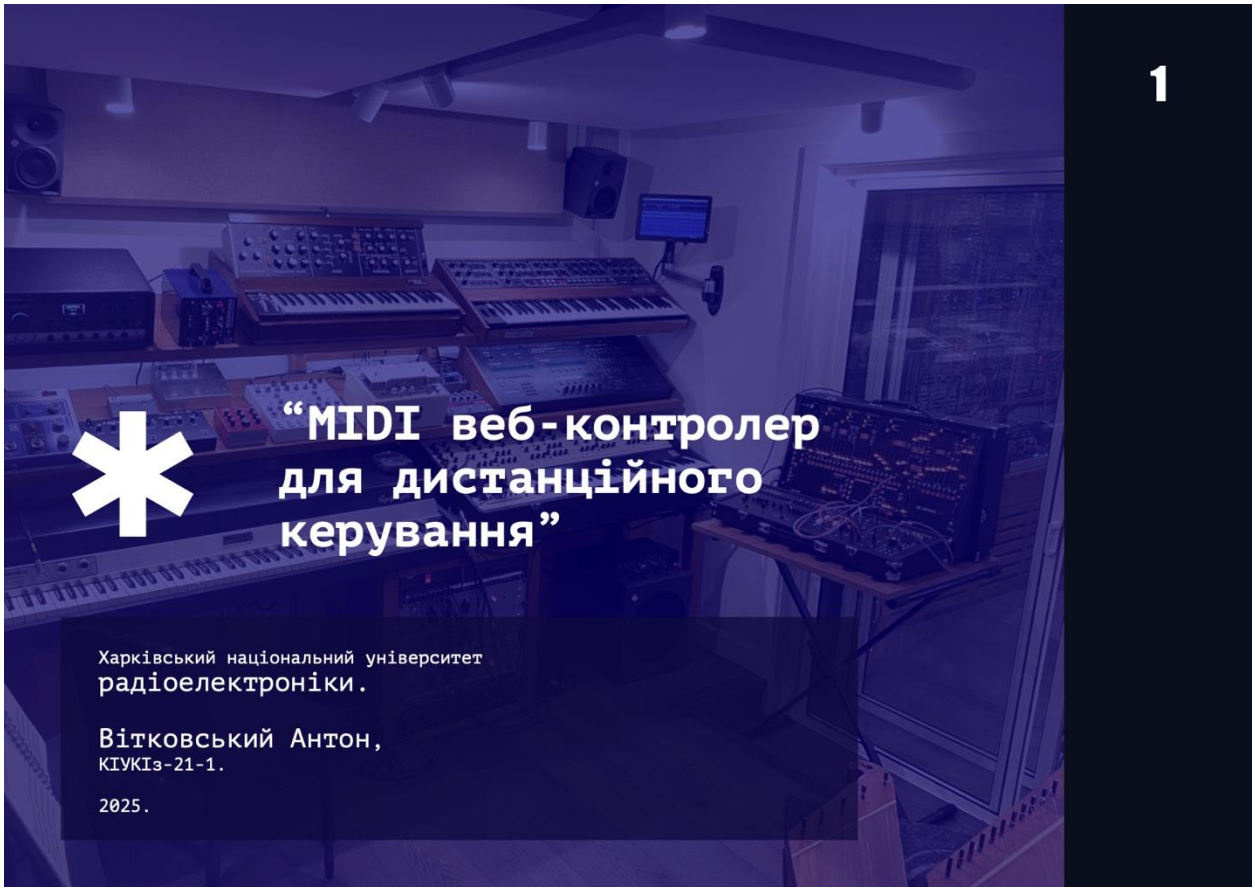
ДОДАТОК А

Графічний матеріал кваліфікаційної роботи



Що таке MIDI?

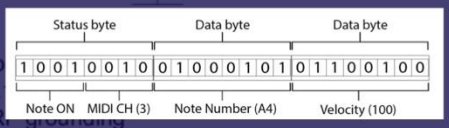
2

MIDI (Musical Instrument Digital Interface)

Специфікація MIDI (Musical Instrument Digital Interface, 1987) визначає як апаратний інтерфейс, так і протокол зв'язку для цифрових музичних інструментів та інших пристроїв.

MIDI протокол - це подієвий механізм, що працює на концепції повідомлень. Кожне повідомлення містить потрібно інформацію для конкретного типу дії, номер пристрою та його каналv.

MIDI повідомлення складаються з 3 частин: статусний байт(містить тип повідомлення, координати пристрою отримувача) та 2 байти даних(їх зміст формує тип повідомлення)



1 - Специфікація MIDI описує стандартний 5-контактний MIDI роз'єм для швидкої передачі вхідної(MIDI in) або вихідної(MIDI out) музичної інформації.

2 - На відміну від звичайного 5-контактного MIDI порту, USB має переваги: швидкість, пряме підключення до комп'ютеру без перехідників, додатковий функціонал.

Сучасні MIDI контролери.

3

Сучасні MIDI рішення мають широкий функціонал, але і мають певні недоліки.

Можливості:

- Гнучка система налаштувань, що дозволяє працювати з різними типами пристроїв.
- Можуть працювати з великою кількістю пристроїв підключених одночасно(multitrack).
- Реалізують унікальні способи генерації MIDI даних(збережені шаблони, алгоритми).
- Мають підтримку складної автоматизації відправки MIDI даних(CC parameter automation).
- Підтримують експорт та імпорт проектів.
- Інтеграція з плагінами іншими засобами контролю(VST).
- Часто мають функціонал не тільки MIDI керування, а і генерації звуку.
- Інші можливості, що дають багато можливостей для роботи з музичним матеріалом.

Недоліки:

- Хоча професійні DAW та апаратні секвенсори є потужними інструментами, вони можуть бути дуже важкими в розумінні і часто мають надлишковий функціонал для простих та швидких сценаріїв.
- Вартість ПЗ або апаратних рішень може бути дуже високою. Для ПЗ ціна може починатись від 300$, для апаратних - навіть більше.
- Для використання програмного забезпечення ви повинні переконатись, що ваша операційна система підтримує його.
- Апаратні засоби займають місце у робочому просторі, обмежені фізичним керуванням та вимагають додаткового обладнання.
- Вони не завжди зручні або доступні для швидких, мобільних або освітніх сценаріїв.

Що таке Web MIDI API?

4

Web MIDI API - це HTML стандарт, який реалізує JavaScript інтерфейс для доступу до підключених MIDI пристроїв з браузеру та взаємодію з ними через повідомлення.

Взаємодія сторінки з користувачем починається з запиту дозволу, Після чого, браузер надає список усіх пристроїв, готових до взаємодії.

```
navigator.permissions.query({ name: "midi", sysex: true }).then((result) => {
  if (result.state === "granted") {
    // Access granted.
  } else if (result.state === "prompt") {
    // Using API will prompt for permission
  }
  // Permission was denied by user prompt or permission policy
});
```

inputs/outputs - Усі підключені пристрої знаходяться через методи inputs() та outputs() наданого дозволу.

send() - Об'єкти output, повернені запитом доступу, реалізують методи send() для

> Web MIDI API підходить для цього проекту, оскільки дозволяє напряму взаємодіяти з MIDI-пристроями прямо з браузера, без потреби в установці додаткового програмного забезпечення. Він забезпечує кросплатформенність, простоту підключення, підтримку реального часу та гнучкість у створенні користувацьких інтерфейсів, що ідеально

Архітектура проекту - модель.

5

Для архітектури проекту було обрано класичну подієво-компонентну модель(Event-driven Modular Architecture), що легко масштабується.

- В основі логіки застосунку лежать **події**, які ініціюються компонентами системи, та відслідковуються компонентами, для яких важливі реакції на ці події.
- Наприклад, дії користувача зчитує компонент **інтерфейсу**, він і викликає події, які підхоплює компонент **даних**, щоб зберегти значення вводу користувача в системі.

Архітектура проекту - компоненти та інтерфейс.

6

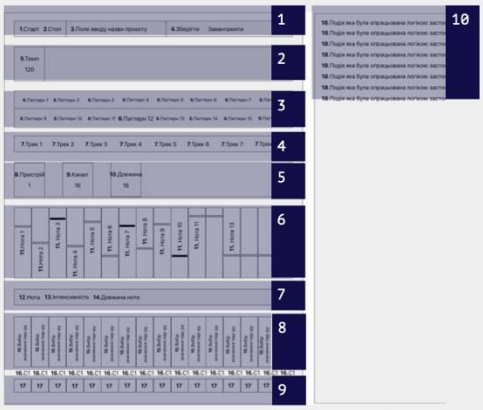
Компоненти подієвої системи:

- Дії користувача зчитує компонент **інтерфейсу(1)**.
- Компонент **даних(2)** застосунку містить усю інформацію про стан системи.
- Початкові налаштування, константи та обмеження містяться у компоненті **конфігурації(3)**.
- **Основна логіка** застосунку складається з компонентів:
- **метроному(4)**, який чітко надсилає події пульсу музичного темпу, для синхронізації пристроїв;
- **секвенсору(5)**, який відповідає за формат музичних даних та логіку її послідовності;
- **MIDI(6)** - реалізує адаптер до інтерфейсу Web MIDI API.

Ієрархія даних:

Застосунок підтримує **8 одночасно підключених пристроїв**, **для котрих є окрема доріжка(track)** з музичними даними. Користувач має **16 збережених послідовностей(pattern)**. Для кожної послідовності є **16 нот для керування**. Ноти мають такі керовані характеристики(яка саме **нота певної октави**, її **гучність** та **довжина програвання**).

Така ієрархія даних дозволяє писати комплексні музичні композиції, керуючі 8 пристроями одночасно.



Інтерфейс має послідовну структуру:

- **панель головного керування(1)**(кнопки старт та стоп, рядок назва проекту, кнопки збереження на завантаження проектів);
- **панель додаткового керування(2)**(налаштування музичного темпу);
- **панель обрання збережених послідовностей(3)**, з 16 доступних;
- **панель обрання доріжки(4)** одного з 8 пристроїв;
- **панель налаштування доріжки(5)** пристрою;
- **панель візуалізації(6)** музичної послідовності для 16 нот доріжки;
- **панель обрання керованої характеристики нот(7)**(впливає на те, чим керують **слайдери контролю нот(8)**);
- **кнопки активації та дезактивації(9)** кожної з 16 нот окремо;
- **панель моніторингу(10)** подій в системі.

Базовий сценарій використання.

7

1. користувач під'єднає пристрій через USB
2. відкриє застосунок у браузері
3. дозволить доступ до пристрою
4. обере MIDI налаштування, відносно пристрою
5. обере темп композиції
6. введе нотну послідовність
7. натисне кнопку початку відтворення
8. не зупиняючи відтворення, буде редагувати ноти для 16 збережених послідовностей
9. при необхідності, додасть більше пристроїв

Для тестування застосунку був обраний сучасний музичний пристрій **Elektron Digitakt.**

Він має багату MIDI підтримку - як через стандартний MIDI кабель, так і через USB.

Ключова відмінність цього пристрою - він **багатоканальний**, тобто ми можемо використовувати більше ніж 8 MIDI каналів(пристроїв) одночасно. Тож це дає нам можливість повністю протестувати усі можливі сценарії використання застосунку.

Особливості реалізації компонентів.

8

Interface - компонент взаємодії користувача з інтерфейсом.

- Кожен елемент керування представлено класом (кнопки, слайдери, візуалізація).

```
ScrollValueControl: class {
  constructor(element, scrollId, values, onChange = null, defaultValue = null) {}
```

- Кожен клас модулю Interface приєднується до відповідного об'єкту елементу HTML(DOM), отримуючи його як параметр.

```
this.UI.trackMidiChannelControl = new this.UI.ScrollValueControl(
  document.getElementById('midi-ch-control'),
);
```

- Класи елементів інтерфейсу мають методи типу render() або update(), які змінюють стилі, відповідно до логіки відображення інформації, яку вони містять.

```
updateVisual() {
  this.el.classList.toggle('active', this.value);
}
```

- Класи елементів прослуховують події користувача та викликають відповідні події системи, для того щоб інші компоненти в системі мали можливість реагувати на них.

```
button.addEventListener('click', () => {
  el.remove();
  document.dispatchEvent(new CustomEvent('app.ui.start'));
});
```

Дія користувача у браузері → JavaScript DOM події → Обробка події класом елементу → Оновлення стану застосунку

9

Data - модуль даних застосунку.

- Модуль Data, тримає в собі усі дані застосунку у багаторівневому JSON об'єкті. Це включає музичну інформацію нотних послідовностей та стан самого застосунку(активні елементи інтерфейсу, обрані налаштування, тощо).

```
data: {
  currentPattern: 0,
  currentTrack: 0,
  activeValueControls: 'note',
  tempo: 120,
  patterns: [],
  logs: []
},
```

- Модуль даних підписаний на події інтерфейсу, та змінюється, відповідно до вхідних даних. Наприклад, при натисканні однієї з кнопок послідовностей нот, подія несе в собі індекс цієї послідовності, цей індекс зберігається в Data.

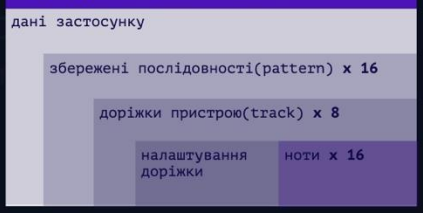
```
this.Utils.listen('app.ui.pattern.selected', e => {
  this.setCurrentPattern(e.detail.selectedIndex); # індекс зберігається в Data
  this.UI.update(); # інтерфейс оновлюється
});
```

Config - модуль початкових даних та налаштувань.

Модуль Config, несе дані, які записуються в Data як значення по замовчуванню, а також має деякі налаштування системи.

```
generateTrackPattern: function() {
  const pattern = {
    midiOut: this.config.defaultMidiOut,
    midiChannel: this.config.defaultMidiChannel,
```

Структура збережених даних.



Metronome - головний музичний рушій. 10

Metronome повинен:
- бути надійним та дуже чутливим до часу
- працювати без затримок
- чітко відповідати до встановлених налаштувань музичного

Проблема:

Вбудована Javascript функція **setInterval()**:
- кожен раз викликає вказану функцію у проміжку вказаного часу
- у браузерному середовищі ця функція оптимізована для роботи зі сторінками
- через цю оптимізацію вона може мати затримку, або взагалі не спрацьовувати

Рішення:

Побудувати власну функцію Metronome.start(), яка:
- вираховує удари метроному у майбутньому та порівнює теперішній в розрахунками
- якщо теперішній час співпадає з часом розрахованого удару, викликати системну подію
- викликати системну подію “app.engine.metronome.tick”, що означає удар метроному

Музичні особливості роботи метроному(системна подія ”app.engine.metronome.tick”):

Музичні події мають більш щільну концентрацію в часі, ніж темп ударів метроному.
Але метроном все одно є головним орієнтиром в часі.
Тому модуль Metronome на 1 удар викликає 4 події “кроку” в системі.
Це дає можливість розмістити 4 ноти між ударами метронома.

встановлений темп
виклик події "app.engine.metronome.tick"

Sequencer та MIDI - модулі логіки та передачі музичної MIDI інформації. 11

Функція Sequencer.advance() робить крок вперед по послідовності, збереженій в Data, вираховує відповідну ноту та підлаштовує її у зручний формат для MIDI:

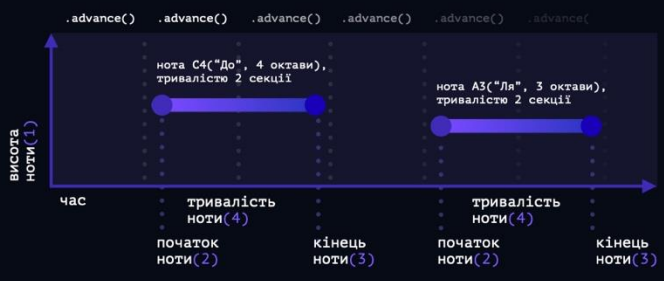
- Sequencer.advance() реагує на системну подію метроному “app.engine.metronome.tick”, та кожного разу вираховує ноту.

- Sequencer.advance() викликає системну подію “app.sequencer.note_played”, передає усі характеристики MIDI ноти, яку потрібно відтворити.

Функція MIDI.send() реагує на системну подію “app.sequencer.note_played” та формує MIDI повідомлення, яке буде передане в Web MIDI API.

app.engine.metronome.tick > Sequencer.advance() > app.sequencer.note_played > MIDI.send() > Web MIDI Api

Дані MIDI подій "app.sequencer.note_played":



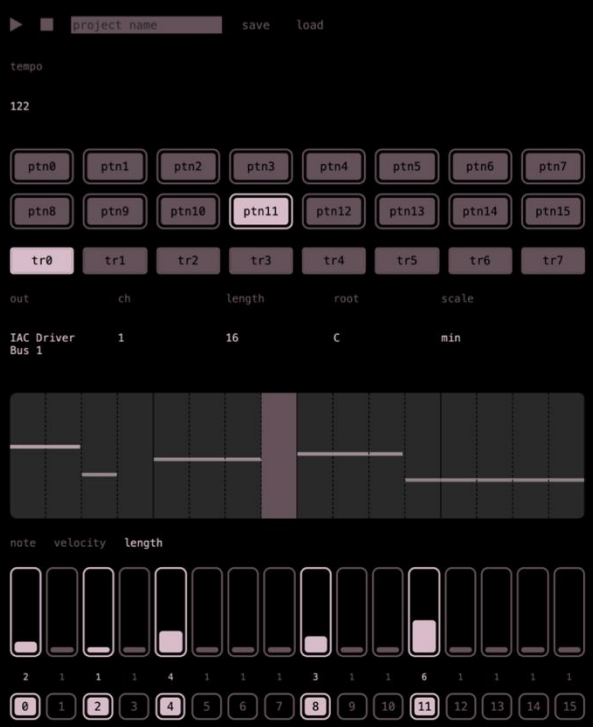
Це стандартне уявлення про **нотну послідовність**, використане в багатьох програмах та апаратних пристроях MIDI керування.

- (1) Висота ноти або номер ноти(note number)
- (2) Час початку програвання ноти
- (3) Час коли нота закінчується
- (4) Тривалість ноти(4)(length)
- (5) Гучність ноти(velocity)

Гучність ноти(velocity)(5) немає на графіку, але це такая сама характеристика як висота ноти, що вказує на те як інтенсивно MIDI пристрій повинен її програвати.

Результат.

12



Як результат маємо гнучкий та простий у використанні застосунок Web MIDI контролеру.

- Працює з 8 підключеними MIDI пристроями одночасно.

- Зберігає 16 послідовностей, які можна перемикати та редагувати у режимі реального часу.

- Візуалізує обрані дані музичних послідовностей у реальному часі.

- Дозволяє створювати повноцінні та комплексні музичні композиції та зберігати їх для відтворення у будь який час

- Доступний для використання у будь яких умовах та не вимагає додаткових налаштувань.

- Не залежить від ОС та обладнання.

- Має потенціал для розвитку та більших можливостей.

Тестування та висновки.

13

Тестування.

Під час тестування було пройдено ключовий сценарій крок за кроком, ураховуючи характеристики підключеного пристрою. Базовий функціонал застосунку працював відповідно до вимог. Підключений пристрій керувався з інтерфейсу застосунку швидко, без затримок.

Протягом тестування було знайдено деякі недопрацювання: зсув музичної послідовності після певних дій користувача та дефекти інтерфейсу. Всі дефекти були виправлені.

Потенціал розвитку.

У розробленому застосунку не вистачало декілька функцій, які могли б ще більше інтегрувати його у повноцінному музичному середовищі.

Наприклад:
- синхронізація з іншими пристроями по темпу
- відправка MIDI повідомлень для контролю параметрів музичного обладнання
- більше гнучке налаштування нотних послідовностей на основі алгоритмів, функцій генерації випадкових нот
- підтримка контролерів для фізичного керування

Потенціал розвитку.

У розробленому застосунку не вистачало декілька функцій, які могли б ще більше інтегрувати його у повноцінному музичному середовищі. Наприклад:
- синхронізація з іншими пристроями по темпу
- відправка MIDI повідомлень для контролю параметрів музичного обладнання
- більше гнучке налаштування нотних послідовностей на основі алгоритмів, функцій генерації випадкових нот
- підтримка контролерів для фізичного

Сценарії користування.

Застосунок дозволяє працювати у мінімальних умовах, без додаткових налаштувань та обладнання.

Це дає можливості для різних сценаріїв користування:
- живі виступи музикантів
- навчання та експерименти
- тестування музичного обладнання
- інтеграція з DAW(Digital Audio Workstation)

Висновки.

Розроблений додаток відповідає очікуванням у ключових сценаріях використання та працює стабільно з великою кількістю даних.

Застосунок коректно передає всі види музичної інформації - ноти та їх характеристики(номер ноти, тривалість та інтенсивність).

Віддалене керування апаратними пристроями на прикладі Elektron Digitakt практично продемонструвало ефективність Web MIDI API, тому застосунок готовий до повноцінної роботи з музичним матеріалом в умовах швидкого розгортання або складних апаратних конфігураціях.