



Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №2
Технології розроблення програмного забезпечення

*Діаграма варіантів використання. Сценарії варіантів використання. Діаграми UML.
Діаграми класів. Концептуальна модель системи
Варіант №5*

Виконав
студент групи ІА-13:
Засінець А. Є.

Київ 2023

Тема: Діаграма варіантів використання. Сценарії варіантів використання. Діаграми UML. Діаграми класів. Концептуальна модель системи.

Хід роботи:

Перед початком виконання роботи, визначимо тему. Я 5-й за списком у групі, тому обрав тему під номером 5:

..5 Аудіо редактор (singleton, adapter, observer, mediator, composite, client-server)

Аудіо редактор повинен володіти наступним функціоналом: представлення аудіо даних будь-якого формату в WAVE-формі, вибір і подальші операції копіювання / вставки / вирізання / деформації по сегменту аудіозапису, можливість роботи з декількома звуковими доріжками, кодування в найбільш поширених форматах (ogg, flac, mp3).

Тепер можна перейти до завдання даної лабораторної роботи:

Завдання:

1. Ознайомитися з короткими теоретичними відомостями.
2. Проаналізуйте тему та намалюйте схему прецеденту, що відповідає обраній темі лабораторії.
 - а. Аналіз теми.

Згідно з описом теми, Аудіо редактор матиме наступний функціонал:

Представлення аудіо даних у WAVE-формі. Можна зрозуміти, що дана функція вимагає здатності читання та відображення аудіо у WAVE-формі.

Операції з сегментами аудіозапису.

- Копіювання
- Вставка
- Вирізання
- Деформація

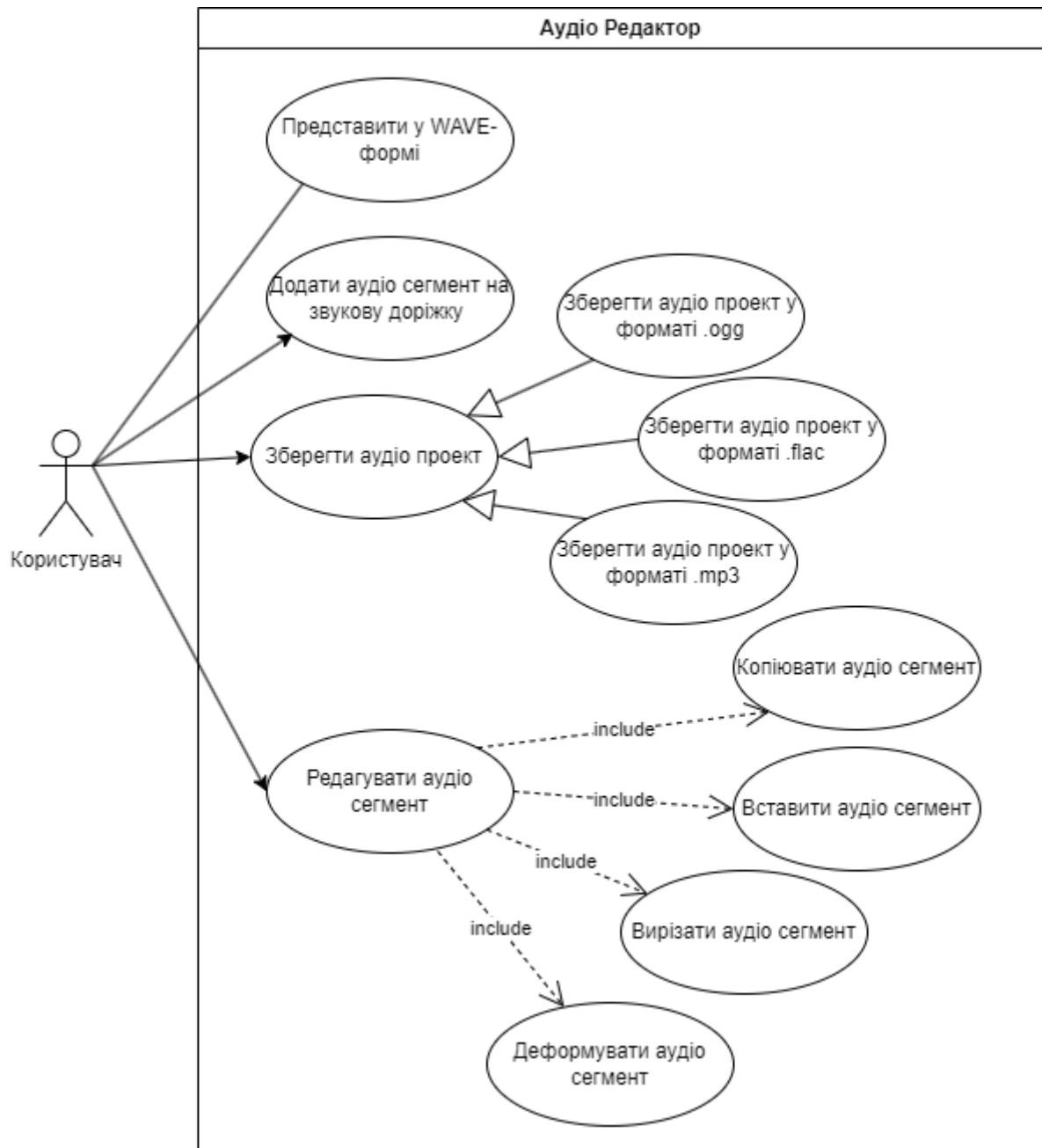
Робота з декількома звуковими доріжками. Дана функція дозволяє користувачеві працювати з кількома звуковими доріжками разом або окремо.

Кодування в поширених форматах.

- ogg
- flac
- mp3

По суті, можна виділити двох акторів: Система та Користувач. Часто, під час будування діаграм варіантів використання саму Систему не виділяють і не позначають як окремого актора. Отже, для реалізації даної теми нам буде достатньо виділити всього одного актора - Користувача.

На даному етапі, після аналізу, можна намалювати схему прецедентів:

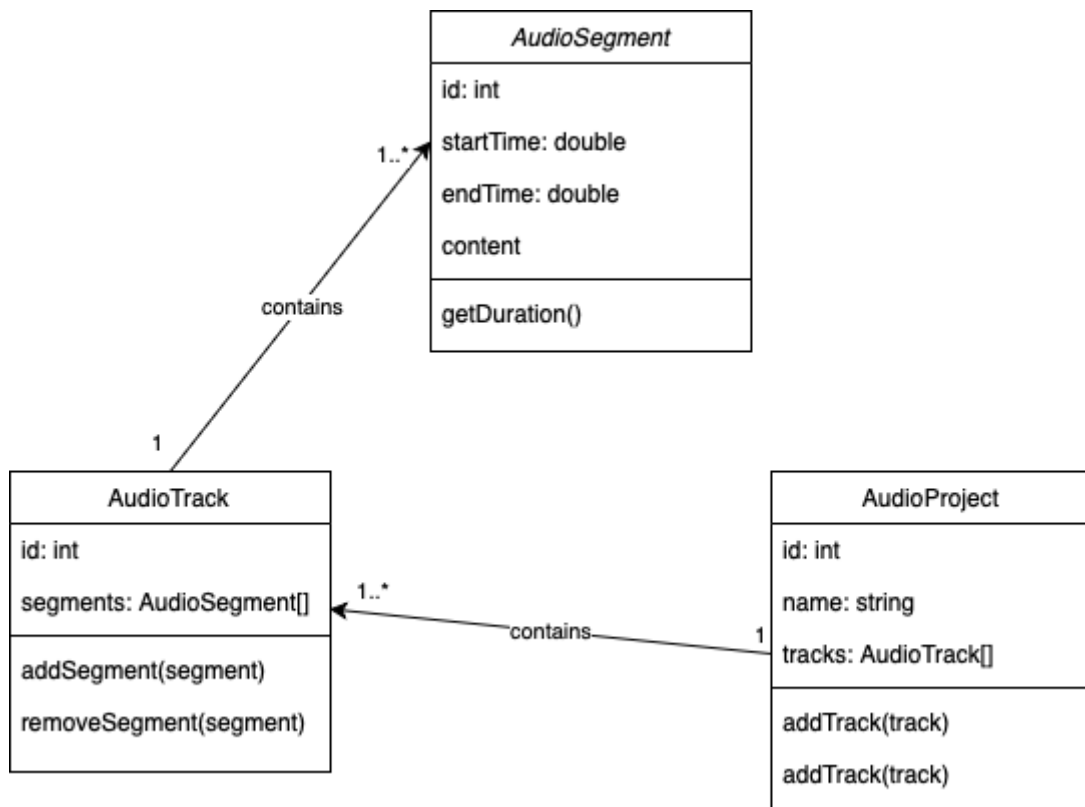


3. Намалуйте діаграму класів для реалізованої частини системи.

Можемо виділити декілька класів, що будуть взаємодіяти:

AudioSegment, AudioTrack, AudioProject, детально вони розписані в пункті [5](#).

Отже, намалюємо діаграму класів з використанням саме цих класів.



4. Виберіть 3 прецеденти і напишіть на їх основі прецеденти.

Оберемо 3 основні прецеденти для аудіо редактора та опишемо їх:

1. Прецедент: "Додавання аудіо сегменту на звукову доріжку"

Опис: Користувач може додати новий аудіо сегмент на вибрану звукову доріжку в аудіо редакторі.

2. Прецедент: "Збереження аудіо проекту у вказаному форматі"

Опис: Користувач може зберегти свій поточний аудіо проект у вказаному аудіоформаті (наприклад, ogg, flac, mp3).

3. Прецедент: "Редагування аудіо сегменту"

Опис: Користувач може вибрати аудіо сегмент та виконати на ньому операції редагування, такі як зміна гучності, зміна швидкості тощо.

Також, на схемі прецедентів було позначено деякі прецеденти, які є або дочірніми до цих трьох, або доповнюють їх.

5. Розробити основні класи і структуру системи баз даних. Класи даних повинні реалізувати шаблон Репозиторію для взаємодії з базою даних.

Основні класи даних:

1. **AudioSegment** (Клас, що представляє аудіо сегмент):

- Властивості:

- `id`: унікальний ідентифікатор сегменту.
- `startTime`: початковий часовий показник сегменту.
- `endTime`: кінцевий часовий показник сегменту.
- `content`: вміст аудіо сегменту.

- Методи:

- `getDuration()`: отримання тривалості сегменту.

2. **AudioTrack** (Клас, що представляє звукову доріжку):

- Властивості:

- `id`: унікальний ідентифікатор доріжки.
- `segments`: список аудіо сегментів на доріжці.

- Методи:

- `addSegment(segment)`: додавання аудіо сегменту на доріжку.
- `removeSegment(segment)`: видалення аудіо сегменту з доріжки.

3. **AudioProject** (Клас, що представляє аудіо проект):

- Властивості:

- `id`: унікальний ідентифікатор проекту.
- `name`: назва проекту.
- `tracks`: список звукових доріжок у проекті.

- Методи:

- `addTrack(track)`: додавання нової звукової доріжки до проекту.
- `removeTrack(track)`: видалення звукової доріжки з проекту.

Структура системи бази даних:

В базі даних буде одна основна таблиця для кожного класу даних:

`audio_segments`, `audio_tracks`, та `audio_projects`.

1. Таблиця `audio_segments`:

- Поля:

- `id` (PRIMARY KEY)
- `start_time`
- `end_time`

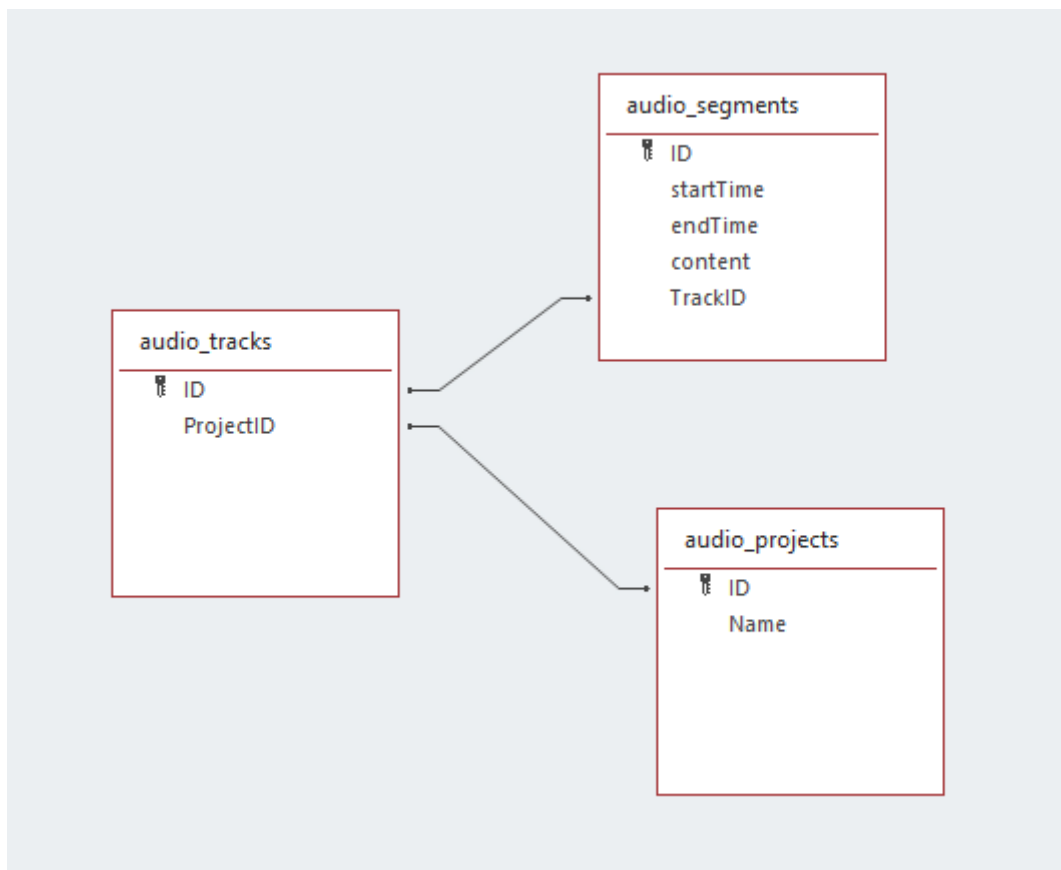
- `content`
- `track_id` (зовнішній ключ, пов'язаний з `audio_tracks.id`)

2. Таблиця `audio_tracks`:

- Поля:
- `id` (PRIMARY KEY)
- `project_id` (зовнішній ключ, пов'язаний з `audio_projects.id`)

3. Таблиця `audio_projects`:


- Поля:
- `id` (PRIMARY KEY)
- `name`



- Підготувати звіт про хід виконання лабораторних робіт. Звіт, що подається повинен містити: діаграму прецедентів, діаграму класів системи, вихідні коди класів системи, а також зображення структури бази даних.

Для розробки проекту було обрано мову програмування Java. Вихідні коди класів поки що виглядають так(поки що тільки приблизна візуалізація):

AudioProject

```
1  import java.util.ArrayList;
2
   no usages
3  public class AudioProject {
   1 usage
4      private int id;
   1 usage
5      private String name;
   3 usages
6      private ArrayList<AudioTrack> tracks;
7
   no usages
8      public AudioProject(int id, String name, ArrayList<AudioTrack> tracks){
9          this.id = id;
10         this.name = name;
11         this.tracks = tracks;
12     }
13
   no usages
14     public void addTrack(AudioTrack track) {
15         this.tracks.add(track);
16     }
17
   no usages
18     public void removeTrack(AudioTrack track) {
19          this.tracks.remove(track);
20     }
21 }
```

AudioTrack

```
1  import java.util.ArrayList;
2
3  4 usages
4  public class AudioTrack {
5      1 usage
6      private int id;
7      3 usages
8      private ArrayList<AudioSegment> segments;
9
10     no usages
11     public AudioTrack(int id, ArrayList<AudioSegment> segments)
12     {
13         this.id = id;
14         this.segments = segments;
15     }
16
17     no usages
18     public void addSegment(AudioSegment segment){
19         this.segments.add(segment);
20     }
21
22     no usages
23     public void removeSegment(AudioSegment segment){
24         this.segments.remove(segment);
25     }
26 }
```


AudioSegment

```
1 public class AudioSegment {  
    1 usage  
2     private int id;  
    2 usages  
3     private float startTime;  
    2 usages  
4     private float endTime;  
    1 usage  
5     private String content;  
6  
    no usages  
7     public AudioSegment(int id, float startTime, float endTime, String content) {  
8         this.id = id;  
9         this.startTime = startTime;  
10        this.endTime = endTime;  
11        this.content = content;  
12    }  
13  
    no usages  
14    public float getDuration() {  
15        return this.endTime - this.startTime;  
16    }  
17 }  
18
```

Висновки: Під час виконання даної лабораторної роботи ми вивчили інформацію про діаграми варіантів використання, діаграми класів, та концептуальну модель системи та застосували отриманні знання на практиці.