МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: **Розробка мобільного додатку «Android аудиоплеєр»**

Виконала студентка групи **К-21і**
спеціальності 122 Комп'ютерні науки
Головіна Олександра Андріївна

Керівник **д.ф., доцент**
Бучинська І.В.

Консультант

Рецензент **к.ф.-м.н., професор**
Ковальчук В.В.

Одеса 2023

ЗМІСТ

ВСТУП

Смартфони та інші мобільні пристрої стали не тільки частиною нашого повсякденного життя, вони – повноцінне продовження людини. Люди за допомогою мобільних телефонів не тільки спілкуються один з одним, але й замовляють товари з магазинів, купують квитки, викликають таксі, бронюють житло, використовують телефони як навігатори, фото- і відеокамери, онлайн-банки, або використовують як спосіб розваг чи для скорочення часу. Згідно статистики, опублікованою в Datareportal, біля 67% дорослих людей усього світу щодня використовують смартфони, а це близько 5,2 млрд осіб. З кожним роком тенденція до переходу з простих мобільних пристроїв на багатофункціональні смартфони тільки збільшується[1].

Актуальність розробки мобільних додатків зростає не те що з кожним роком, а й з кожним місяцем, бо сотні нових мобільних додатків виходять на онлайн-майданчиках кожен день.

Розробка мобільних додатків є відносно новою областю дослідження, яка продовжує розвиватися. Незважаючи на свою молодість, завдання, які виконують мобільні додатки, стають все більш складними та різноманітними. До них належать загальні наукові завдання, такі як розпізнавання зображень і звуків, штучний інтелект, паралельне програмування, а також спеціальні дослідження методів інтерактивної взаємодії людей, адаптивних інтерфейсів користувача та мобільних кіберфізичних систем. Мобільні програми можливо попередньо інсталювати на пристрої, завантажувати через платформи розповсюдження програмного забезпечення або реалізовувати як клієнт-серверні програми. Новітні технології, такі як голосове керування, віртуальні помічники (наприклад, Siri на iOS і Google Now на Android), розпізнавання тексту та об’єктів, легко доступні на мобільних пристроях [6].

Метою роботи є розробити аудиоплеєр з інтуїтивно зрозумілим інтерфейсом, який може відтворювати аудиофайли, та керувати відтворюванням, створювати плейлісти та працювати з метаданими.

Завдання:

- проаналізувати існуючі аудиоплеєри;
- розглянути основні поняття Android-розробки;
- обрати програмні засоби для розробки;
- розробити аудиоплеєр.

Кваліфікаційна работа містить вступ, 4 розділи, 35 рисунків, 56 сторінок, список літератури.

1 АНАЛІЗ І ПОРІВНЯННЯ ІСНУЮЧИХ ПЛЕЄРІВ

1.1 Огляд найпопулярніших аудіо плеєрів

1. Плеєр Simple MP3 (рис. 1) – це простий програвач, розроблений спеціально для пристроїв Android. Він є безкоштовним і зручним для користувача, що робить його зручним варіантом для прослуховування аудиофайлів, які зберігаються на SD-карті пристрою. Деякі переваги цього програвача включають:

– підтримку популярних форматів, таких як MP3, AVI, FLVC і MKV;

– здатність передавати онлайн-аудіо на пристрої;

– не потребує встановлення будь-яких додаткових плагінів.

Однак є кілька недоліків:

– програвач містить небажану рекламу, яка може перешкодити перегляду;

– програвач підтримує лише обмежену кількість основних форматів.

Рисунок 1 – Плеєр Simple MP3

2. VLC для Android (рис. 2) – це безкоштовний плеєр із відкритим кодом, розроблений спеціально для пристроїв Android. Це дозволяє

користувачам відтворювати широкий спектр відео та аудіофайлів, а також мережевих потоків [3]. Цей програвач пропонує кілька переваг:

– підтримка різних форматів, включаючи MKV, MP4, AVI та FLAC;

– можливість відтворювати локальне аудіо, DVD ISO та спільні диски;

– висока якістю звуку;

– підтримка Blu-ray;

– швидка навігація по файлах.

Однак є кілька недоліків:

– можуть бути проблеми зі зміною мови;

– були випадки, коли видалені відео не зникали повністю.

Рисунок 2 – Плеєр VLC для Android

3. MX Player (рис. 3) – чудовий програвач, призначений для пристроїв Android. Це зручне рішення для прослуховування аудіо різних форматів на пристрої Android. Можна легко завантажити безкоштовну версію додатку і користуватися його функціями. Також, програвач пропонує настроювані параметри для зміни стилю тексту та кольорів відповідно до уподобань[3].

Переваги:

– MX Player підтримує широкий спектр форматів, включаючи такі популярні, як MP3, MKV, SWF, AVI, MP3 і FLAC;

– програвач дозволяє налаштувати параметри, щоб покращити враження від перегляду. Це дозволяє апаратному прискоренню апаратного прискорення декодера HW+, покращуючи продуктивність відтворення аудіо.

Недоліки:

– можуть виникнути проблеми з продуктивністю, якщо графічний процесор чи процесор вашого пристрою недостатньо потужні;

– безкоштовна версія містить рекламу.

Рисунок 3 – Плеєр MX Player

4. Player All Format (рис. 4) – плеэр MP3 для Android, який який маэ високу якысть програвання аудіо. Він підходить для відтворення файлів всіх форматів на пристроях Android, має зручний та інтуітивно зрозумілий інтерфейс. Також, цей плеєр Android забезпечує ефективне керування аудіо та можливість обмінюватися файлами[3]. Переваги:

– підтримка всіх аудіоформатів :MKV, MP3, M4V, AVI тощо;

– регульовані параметри для субтитрів і звуку, що забезпечують персоналізовані параметри перегляду;

– покращена конфіденційність із можливістю збереження файлу в особистій папці.

Недоліки:

– містить випадкові рекламні оголошення.

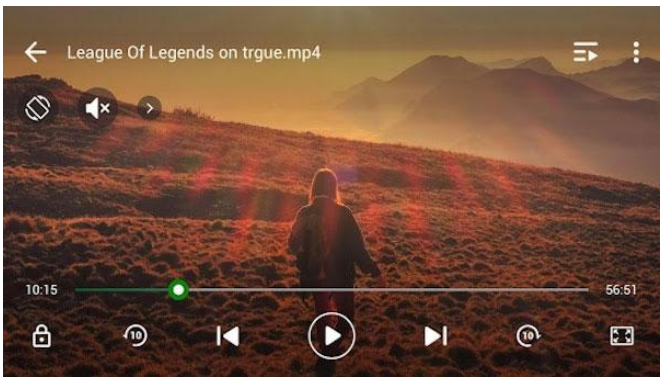

Рисунок 4 – Плеєр Player All Format

5. BSPlayer Free (рис. 5) – це безкоштовний MP3-плеєр, розроблений для пристроїв Android. Він зібрав понад 70 мільйонів користувачів у всьому світі. Завдяки мовним перекладам, доступний на 90 різних мовах, до нього можна отримати доступ і використовувати по всьому світу[3]. Переваги:

– підтримує широкий спектр форматів відеофайлів, таких як MP4, AVI, FLV і 3GP;

– сумісний з пристроями, що працюють на архітектурах ARMv5 і ARMv6.

Недоліки:

– наявність небажаної реклами може потенційно вплинути на взаємодію з користувачем.

Рисунок 5 – Плеєр BSPlayer Free

6. Одним із найпопулярніших додатків є KMPlayer (рис. 6), відомий чудовою продуктивністю MP3-плеєра Android. Дозволяє користувачам насолоджуватися широким спектром аудіо. Ця програма користується прихильністю серед користувачів завдяки своїм функціям. Це дозволяє отримати приємні враження від використання програвача[3].

Переваги:

– пропонує можливість контролювати швидкість відтворення аудіо;

– підтримує різні формати субтитрів, забезпечуючи зручність для користувачів;

– доступний понад 30 різними мовами.

Недоліки:

– іноді програма може автоматично зупинятися під час відтворення відео 1080p;

– безкоштовна версія KMPlayer містить рекламу, яка може бути нав’язливою для деяких користувачів.

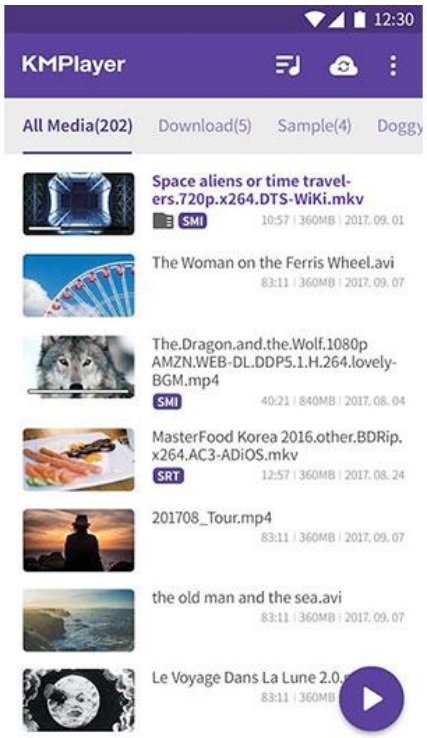

Рисунок 6 – Плеєр KMPlayer

7. Perfect Video Player (рис. 7) на Android – це надійний MP3-плеєр. За відгуками користувачів, він виділяється як естетично привабливий музичний плеєр, пропонуючи виняткову якість аудіо. Переваги:

– автоматично виявляє та відображає всі музичні файли на пристрої Android;

– підтримує широкий спектр популярних аудіо форматів;

– забезпечує автоматичне обертання та підлаштовується під зміни сторінки.

Недоліки:

– містить небажану рекламу;

– часто пропонує користувачам оцінити додаток.

Рисунок 7 – Плеєр Perfect Video Player

8. ATHPlayer для Android (рис. 8). Завдяки розширеному апаратному прискоренню та підтримці субтитрів ця програма гарантує зручне прослуховування аудио. Плеєр споживає мінімум пам’яті та дозволяє відтворювати у фоновому режимі. ATHPlayer має зручні елементи керування, швидко запускається і плавно відтворює аудіо[3].

Переваги:

– підтримує широкий спектр форматів файлів, включаючи AVI, MP3, WAV, MP4 тощо;

– дозволяє відтворювати різні аудіоформати, включаючи аудіо високої чіткості;

– підтримує кілька форматів субтитрів і забезпечує автоматичну синхронізацію.

Недоліки:

– містить рекламу;

– можуть виникнути проблеми з сортуванням відео.

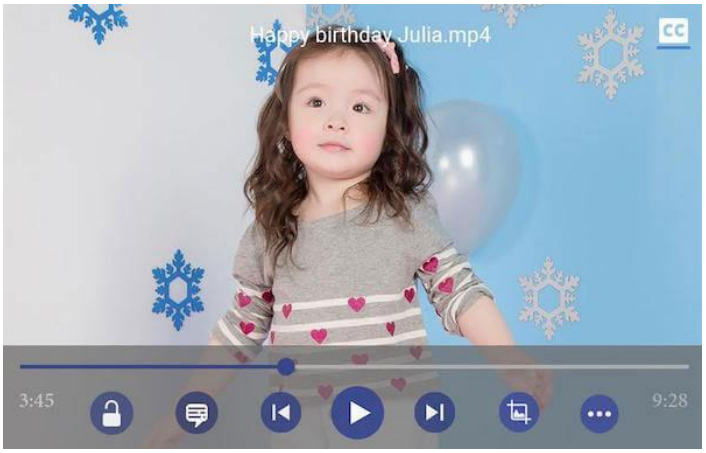

Рисунок 8 – Плеєр ATHPlayer

9. Плеєр MP3-4-5 (рис. 9) підходить для користувачів Android, які хочуть прослуховувати різноманітні аудіо на своєму пристрої. Ця програма пропонує легке налаштування яскравості та гучності екрана. не займає багато пам’яті та усі файли зберігає у хмарі. Крім того, підтримує такі популярні формати, як MP3, FLV, AVI та MKV. Плеєр має обмеження: оскільки він може відтворювати лише основні формати та містить рекламу, яка може набриднути.

Переваги:

– дозволяє регулювати яскравість екрана;

– менший розмір, тому він не займатиме багато місця на пристрої Android;

– підтримка майже всіх популярних форматів, таких як MP4, FLV, AVI, MKV тощо.

Недоліки:

– можна відтворювати тільки базові формати;

– містить рекламу.

Рисунок 9 – Плеєр Плеєр MP3-4-5

10. 321 Player – один із найпопулярніших MP3-плеєрів Android, який має зрозумілий інтерфейс. Підходить для платформи Android, добре контактує з оновленнями ОС. переведений на 30 різних мов і моє користувачів по всьому світу. Додаток можна завантажити безкоштовно, а також є преміям-версія[3].

Переваги:

– підтримка майже всіх аудіо форматів;

– програма може керуватись за допомогою гарнітури.

Недоліки**:**

– містить рекламу;

– може не ідентифікувати файли, якщо ви відкриваєте занадто багато аудіо за один раз .

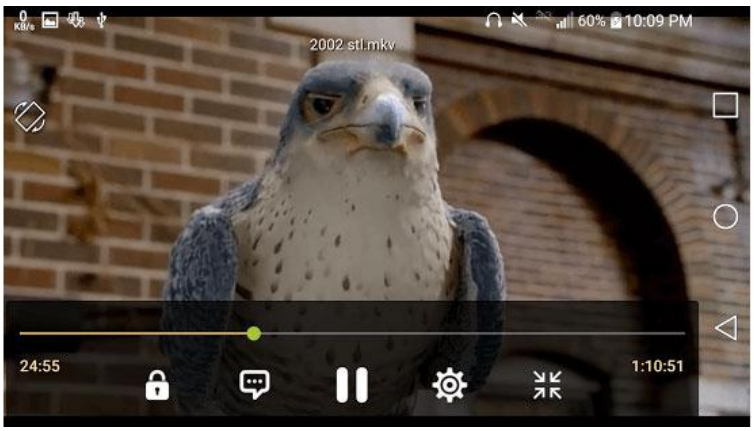

Рисунок 10 – Плеєр 321 Player

1.2 Огляд роботи аудіоплеєра

Програвач Blu-ray AnyMP3 (рис. 11) – один з найпопулярніших програм для відтворення MP3 на Android. Програвач Blu-ray AnyMP4 доступний як для Windows, так і для Mac OS, включаючи Windows 10 і останню версію Mac OS.

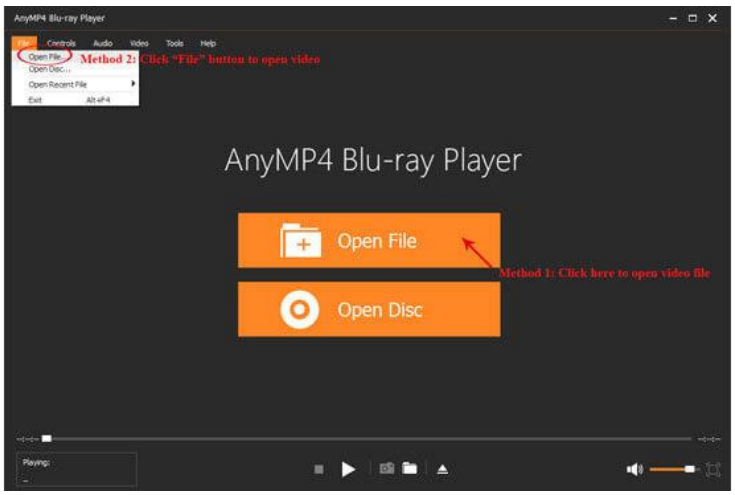

Рисунок 11 – Плеєр Blu-ray AnyMP3

Додаток може відтворювати аудіофрмати MP3 і відео MP4, MOV, AVI, WMV, M4V, MTS, MKV, DivX, MXF тощо. Він має розширення до 4k Blu -ray і ultra HD Blu -ray, тож можна відтворювати фільми 4k Blu-ray із дисків Blu-ray, папок або файлів ISO. Ця програма сумісна з широким діапазоном медіа- та аудіоформатів, включаючи відеофайли, різні диски та аудіо формати[4]. Робота з плеєром:

1. Треба завантажити, інсталювати та запустити AnyMP3 Blu-ray Player на пристрої.

2. Підключається диск операційної системи до пристрою та можна імпортувати файл.

3. Після натискання на кнопку «Відкрити файл», щоб завантажити його.

4. Є можливість обирати звукову доріжку та субтитри.

5. Для початку прослуховування, треба натиснути на кнопку «Відтворити» (рис. 12).

Рисунок 12 – Відтворення файлу

2 ПРЕДМЕТНА ОБЛАСТЬ ТА ПОСТАНОВКА ЗАДАЧІ

2.1 Характеристика предметної області

Розробка мультимедійних програм для Android-пристроїв – це предметна область, пов'язана зі створенням програмного забезпечення для Android-пристроїв, яке призначене для відтворення аудіофайлів різних форматів. Ця програма має можливість відтворювати музичні композиції, подкасти, аудіокниги та інші аудіофайли, а також пропонує додаткові функції, наприклад створення плейлистів, використання еквалайзера, включення та заміну треків та інших звуків[2].

В області розробки мобільного додатка аудіоплеєра є різні технічні аспекти, наприклад, вибір відповідних аудіоформатів, створення ефективного інтерфейсу користувача зі зручною навігацією по трекам і налаштуваннями предметного звуку, а також оптимізація відтворення для швидкої і швидкої передачі аудіофайлів.

У цій предметній області також важливі параметри безпеки, пов'язані з обробкою та зберіганням особистої інформації користувачів, такі як персональні звуки та дані про програні пісні. Таким чином, розробка програмного забезпечення аудіоплеєра вимагає збалансованого відношення до технологічної та користувальницької сторін, а також урахування особливостей, відносин з безпекою та конфіденційністю зовнішнього вигляду.

Щоб надати користувачам найкращий досвід, додаток має пропонувати мобільне розширення, яке є зручним для щоденного використання. Це розширення має бути не лише інформативним, але й швидким у використанні, забезпечуючи ефективну та безпроблемну взаємодію. Щоб забезпечити оптимальну продуктивність і точність, важливо використовувати останні досягнення в галузі інформаційних технологій. Служба звітування про продуктивність буде спеціально розроблена для

роботи в операційній системі Android, яка зберегла свої позиції як найпоширеніша операційна система для смартфонів[1].

Додаток має відображати для користувача наступну інформацію:

– список плей-листів;

– музичні композиції;

– плеєр програвання композиції;

– бібліотеку композицій та ін.

Інтерфейс мобільних додатків відіграє дуже важливу роль в процесі створення програми, адже інтерфейс є сполучною ланкою між апаратним та програмним забезпеченням мобільного пристрою і фокусом користувацької взаємодії. Тому для розробки мобільного додатку аудиоплеєр необхідно використовувати сучасну інтегровану середу розробки Android Studio, що містить усі необхідні інструменти та бібліотеки для роботи з XML розміткою (дизайн, інтерфейс), Java-кодом (логіка взаємодії даних) та побудови проектів, зручного та інтуїтивно зрозумілого розширення для прослуховування аудиофайлів.

Щоб досягти успіху як розробника мобільних додатків, важливо володіти поєднанням технічного досвіду та особистих якостей. У цій конкретній професії є потреба в різних навичках м’якого спілкування, які включають наступне:

– володіння структурним та аналітичним мисленням має вирішальне значення для ефективної навігації в складнощах розробки мобільних додатків;

– необхідна уважність, щоб звернути пильну увагу на деталі та забезпечити створення якісних програм;

– сильні комунікативні навички корисні для співпраці з членами команди та ефективного передачі ідей і вимог;

– творчий підхід і здатність мислити нестандартно є цінними для розробки інноваційних і захоплюючих мобільних додатків;

– важливо мати бажання та здатність брати участь у самонавчанні, оскільки розробка додатків – це сфера, яка постійно розвивається, і потрібно бути в курсі останніх досягнень;

– відповідальність за свою роботу та дотримання зобов'язань є життєво важливими для того, щоб створювати мобільні програми, які відповідають очікуванням.

Володіючи цими навичками спілкування разом із технічними знаннями, розробник мобільних додатків може збільшити свої шанси на досягнення успіху у своїй галузі. Простий додаток може створити кожен середньостатистичний розробник, але для то щоб виділитись, необхідно попрацювати над тим, щоб створити щось дійсно цікаве і оригінальне, а головне, якісне. Для цього знадобляться наступні навички:

- CSS і HTML, середовище розробки Cocoa / Xcode, мови програмування Objective-C / C ++, Java;
- HTTP, XML, принципи об'єктно-орієнтованого програмування, СУБД;
- знання Android, iOS SDK, шаблони проектування, поширені бібліотеки і архітектуру iOS, Android, скриптові мови програмування (Ruby, Python), принципи клієнт-серверної моделі взаємодії додатків;
- знання вимог до релізів додатків в AppStore і Google Play;
- затребувані навички роботи з Core Data (фірмової локальною базою даних від Apple, яка побудована за типом SQL);
- уміння розбиратися в чужому коді[5].

В першу чергу для підходу до визначення етапів розробки необхідно грунтуватися на принципах Human-centered design (HCD), тобто орієнтуватися на проблеми людини (користувача), і робити ставку на інтерфейс.

Щоб розробити мобільний додаток, можна виділити кілька етапів. Ці етапи включають наступне:

– по-перше, це передбачає проведення ретельного дослідження та мозкового штурму для визначення головної ідеї та концепції програми;

– коли ідея сформульована, наступним кроком є визначення основної функціональності, яку програма запропонує своїм користувачам;

– після цього дизайнер працюватиме над створенням візуальних елементів та інтерфейсу програми;

– після завершення дизайну важливо протестувати макети та прототипи, щоб переконатися, що вони зручні та візуально привабливі;

– наступний етап – власне розробка мобільного додатку та будь-яких додаткових сервісів, які можуть знадобитися;

– після завершення етапу розробки виконується ретельне тестування функціональності програми для виявлення та виправлення будь-яких помилок або помилок[6].

Нарешті, після внесення всіх необхідних удосконалень і налаштувань, мобільний додаток готовий до виходу.

Завдання створення програми не просто завершується її розробкою. Крім того, необхідна постійна технічна підтримка для забезпечення його оптимальної продуктивності, а також для вдосконалення та розширення його функціональних можливостей.

Процес розробки мобільного додатку – це тривалий і складний процес. Немає попереднього плану, і цей етап потребує динамічного плану розробки мобільних додатків, який можна адаптувати та розвивати відповідно до вимог проекту.

2.2 Аналіз існуючих мобільних додатків

На сьогоднішній день існує багато мобільних додатків для прослуховування музики. Всі вони мають приблизно схожий функціонал. Для аналізу буде використано два додатка: Google Play Music (рис. 13) та Vivo Music (рис. 14)

Рисунок 13 – Іконка додатку Google Play Music

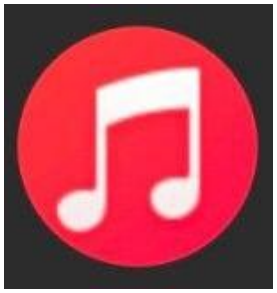

Рисунок 14 – Іконка додатку Vivo Music

Додаток Google Play Music має простий і організований інтерфейс, у якому легко орієнтуватися. На головному екрані відображається всі пісні, які є на пристрої (рис. 15). Є можливість пересуватись по вкладкам.

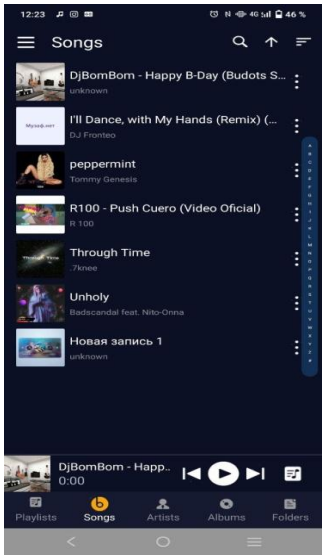

Рисунок 15 – Головне меню додатка Google Play Music

Vivo Music має темний і локанічний інтерфейс, який включає обкладинки альбомів і фоновий екран відтворення. На головному екрані відображаються нещодавно відтворені, папки, списки відтворення та кнопка еквалайзера для налаштування звуку (рис. 16).

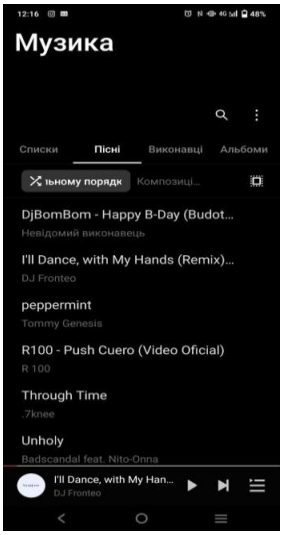

Рисунок 16 – Інтерфейс додатку Vivo Music

Обидва додатки дають змогу компонувати плейлісти. У Google Play Music одразу доданій плейліст «Favorite» (улюблені пісні). Кількість створених плейлістів не обмежується (рис. 17).

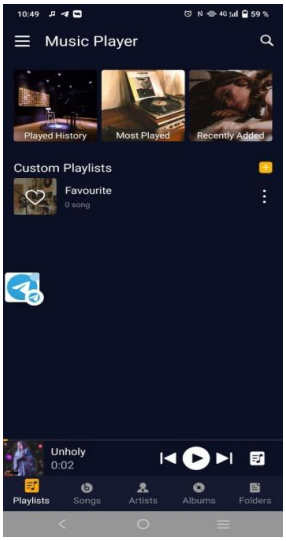

Рисунок 17 – Плейлісти додатку Google Play Music

За замовченням у додатку є три плейлісти:

– Played History (зіграні пісні);

– Most Played (пісні, які найчастіше програються);

– Recently Added (нещодавно додані пісні).

У додатку Vivo Music за замовченням є три плейлісти (рис. 18):

– нещодавно відтворене;

– нещодавно додане;

– список папок (сортовані пісні).

І також можна створити необмежену кількість нових плейлістів. Для улюблених пісень створено «Мої вподобання».

У додатку Google Play Music плейлісти проілюстровані, у Vivo Music ця частина інтерфейсу відсутня.

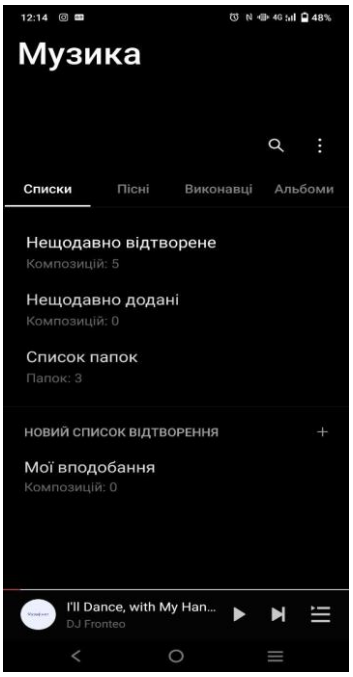

Рисунок 18 – Плейлісти додатку Vivo Music

Обидва додатка моють сортування композицій за виконавцями та альбомами (рис. 19, 20). Сортування за альбомами – це як композиція була завантажена на смартфон (з відомого джерела, або з невідомого).

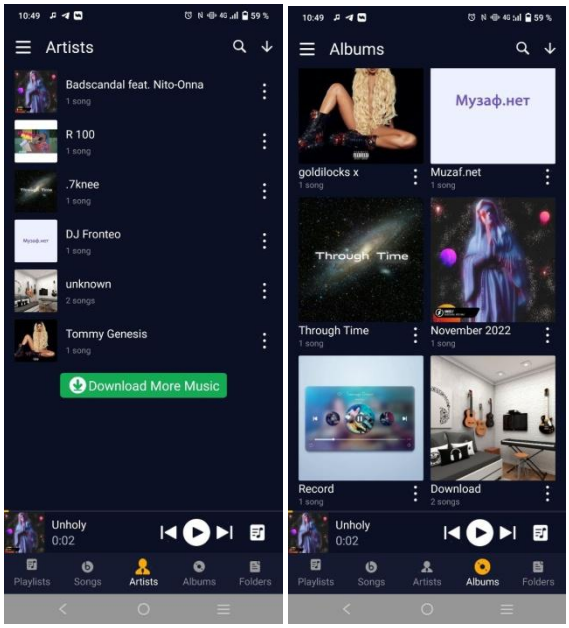

Рисунок 19 – Сортування за виконавцями та альбомами у додатку Google Play Music

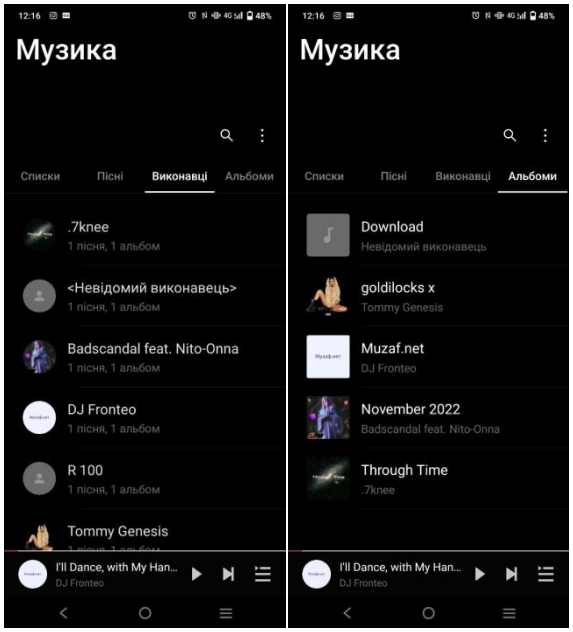

Рисунок 20 – Сортування за виконавцями та альбомами у додатку Vivo Music

Google Play Music пропонує широкий спектр функцій, включаючи безперебійну синхронізацію з Google Play Store та інтеграцію з Google Assistant для голосових команд. Він також має музичний магазин, де

користувачі можуть купувати пісні та альбоми, а також доступ до радіостанцій, кураторами яких є ді-джеї.

Vivo Music має темний і локанічний інтерфейс, який включає обкладинки альбомів і фоновий екран відтворення. На головному екрані відображаються нещодавно відтворені, папки, списки відтворення та кнопка еквалайзера для налаштування звуку.

Функціональні можливості Vivo Music включають відтворення популярних аудіоформатів, таких як MP3, WAV, FLAC тощо, а також синхронізацію з каталогами телефону користувача для безперебійного відтворення музики. Він також має еквалайзер для покращення якості звуку та підтримку настроюваних списків відтворення.

3 ВИБІР ПРОГРАМНИХ ЗАСОБІВ

3.1 Інтегроване середовище розробки Android Studio

Android Studio – це інтегроване середовище розробки (IDE) для платформи Android, яке замінило плагін Eclipse ADT. Середовище розробки базується на вихідному коді продукту IntelliJ IDEA Community Edition, який розробляється JetBrains. Android Studio розроблено за відкритою моделлю розробки та поширюється за ліцензією Apache 2.0. Двійкові збірки підготовлені для Linux (Ubuntu використовується для тестування), Mac OS X і Windows. Середовище надає інструменти для розробки програм не лише для смартфонів і планшетів, але й для носимих пристроїв на базі Android Wear, телевізорів (Android TV), окулярів Google Glass і автомобільних розважальних систем (Android Auto)[8].

Для програм, спочатку розроблених з використанням Eclipse і плагіна ADT, підготовлено інструмент для автоматичного імпорту існуючого проекту в Android Studio. Середовище розробки адаптовано для виконання типових завдань, які вирішуються в процесі розробки додатків для платформи Android. Це і інструменти для спрощення тестування програм на сумісність з різними версіями платформи, і інструменти для розробки програм, які працюють на пристроях з екранами різної роздільної здатності (планшети, смартфони, ноутбуки, годинники, окуляри тощо). На додаток до функцій, присутніх у IntelliJ IDEA, Android Studio реалізує кілька додаткових функцій, таких як нова уніфікована підсистема для складання, тестування та розгортання додатків на основі системи збірки Gradle і підтримки використання інструментів безперервної інтеграції.

Програмне забезпечення для Android OS, розробляється за допомогою Android SDK – набору із засобів розробки, утиліт і документації, який дозволяє програмістам створювати прикладні програми за визначеною

технологією або для певної платформи (програмно-апаратної або програмної). Програмне забезпечення для Android OS, розробляється за допомогою Android SDK[6].

Android SDK використовується для розробки програмного забезпечення для операційної системи Android. SDK, розшифровується як Software Development Kit. Як правило, розробники отримують SDK безпосередньо від постачальника цільової технології або системи, і до них можна легко отримати доступ і завантажити їх онлайн. Іноді SDK доступні безкоштовно, щоб стимулювати використання певної технології чи платформи.

Android SDK – це інструмент, який можна використовувати для розробки мобільних програм спеціально для операційної системи Android. Він надає розширені функції, яких зазвичай немає у звичайних редакторах коду, наприклад можливість тестувати та налагоджувати вихідні коди, оцінювати продуктивність програми в режимі сумісності з різними версіями ОС Android і переглядати результати в реальному часі, якщо потрібно. Підтримуються різні мобільні пристрої, включаючи смартфони, планшети, розумні окуляри (наприклад, Google Glass), сучасні транспортні засоби, оснащені бортовими комп’ютерами з ОС Android, та інші подібні технічні пристрої[4].

Набір для розробки програмного забезпечення (SDK) часто є набором різноманітних інструментів, які використовуються для розробки програмного забезпечення. Ці інструменти дозволяють розробникам створювати програми для конкретного програмного пакета, апаратної платформи, комп’ютерної системи, операційної системи, ігрової консолі чи подібної платформи розробки. Щоб додати розширені функції до своїх програм, більшість розробників програмного забезпечення покладаються на окремі SDK. Певні пакети SDK мають вирішальне значення для додатків спеціальної розробки. Для прикладу: якщо ви збираєтеся створювати програму для Android на платформі Java, вам знадобиться Java Development Kit. Подібним чином для

додатків iOS необхідний iOS SDK, а для платформи Windows – .NET Framework SDK. (рис. 21).

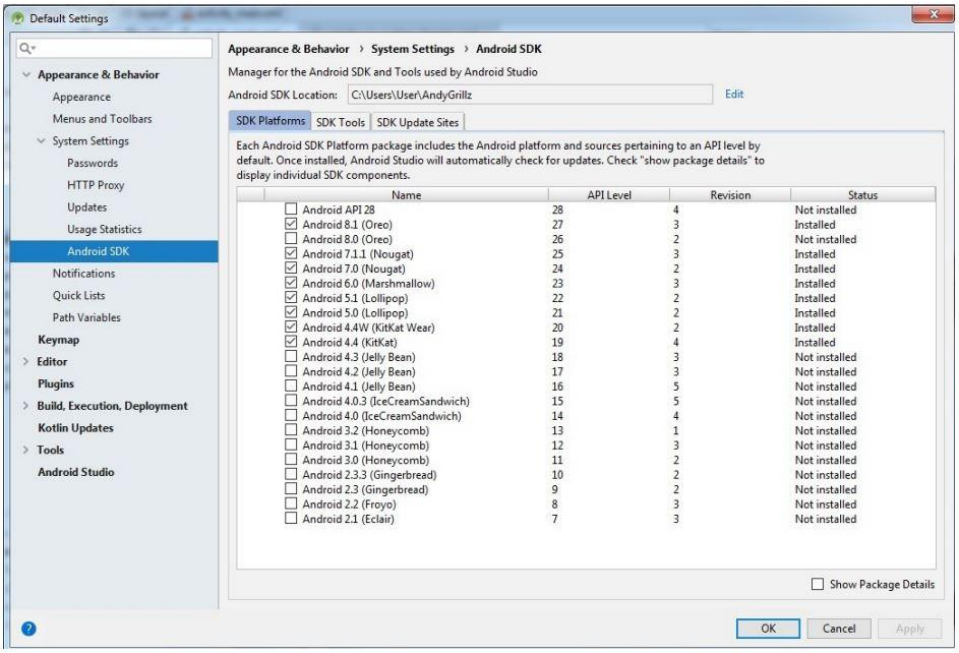

Рисунок 21 – Android SDK Manager

3.2 Загальні поняття Andriod-розробки

Android-розробка – це процес створення мобільних додатків, які працюють на операційній системі Android. Загальні концепції розробки Android включають:

– програми Android складаються з однієї або кількох дій, які представляють різні екрани програми;

– інтерфейс користувача програми Android створюється за допомогою макетів, які визначають розміщення та зовнішній вигляд елементів інтерфейсу користувача, таких як кнопки, текстові поля та зображення;

– наміри використовуються для полегшення зв’язку між діяльністю та іншими компонентами програми Android;

– служби використовуються для виконання фонових завдань, які не

включають інтерфейс користувача;

– приймачі широкомовної передачі використовуються для отримання та відповіді на загальносистемні події, такі як завершення завантаження файлу;

– постачальники вмісту використовуються для керування спільними даними, такими як контакти, події календаря та медіафайли;

– фрагменти – це багаторазово використовувані компоненти інтерфейсу користувача, які можна додавати до дій для створення складніших інтерфейсів користувача[7].

Розробники можуть використовувати різні інструменти та технології для створення додатків для Android, зокрема Android Studio, Eclipse та IntelliJ IDEA тощо. Крім того, доступна велика кількість сторонніх бібліотек і інструментів, які можна використовувати для спрощення розробки Android.

3.3 Розробка графічного інтерфейсу

Графічний інтерфейс користувача – тип інтерфейсу, який дозволяє користувачам взаємодіяти з електронними пристроями через графічні зображення та візуальні вказівки, на відміну від текстових інтерфейсів, заснованих на використанні тексту, текстовому наборі команд та текстовій навігації (рис. 22).

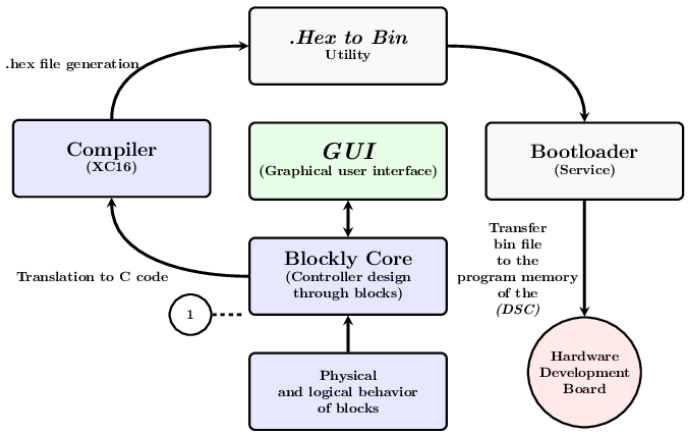

Рисунок 22 – Схема розробки графічного інтерфейса

Користувач має бачити перед собою інтуїтивно зрозумілий, при цьому ненав'язливий інтерфейс, який не викликатиме проблем або банального дратування у повсякденному використанні[5].

Існує безліч видів графічних інтерфейсів, але найпоширеніші включають:

1. Windows – це найпоширеніший тип графічного інтерфейсу, який заснований на збірнику вікон, кнопок та багатьох інших елементів для виконання завдань на ПК (рис. 23).

Рисунок 23 – Інтерфейс Windows

2. MacOS – це графічний інтерфейс для комп'ютерів Apple. Часто трапляється так, що програмне забезпечення та використання кількох вікон дозволяють користувачам швидко перемикатися між завданнями (рис. 24).

Рисунок 24 – Інтерфейс MacOS

3. Linux – це сімейство програмних платформ, які часто використовують графічні інтерфейси як основні форми спілкування з користувачами (рис. 25).

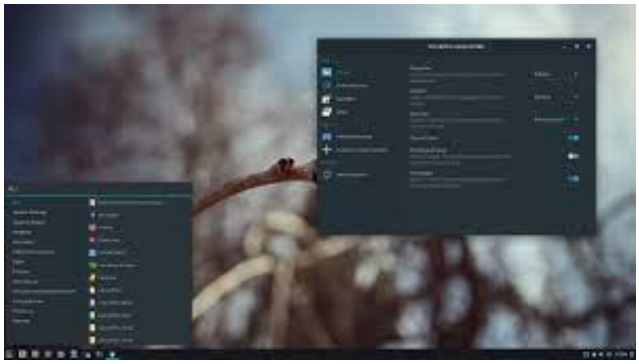

Рисунок 25 – Інтерфейс Linux

4. Android включає в себе домашній екран, стандартні програми, такі як телефон, SMS, віджети, меню. Багато програм Android мають свій стиль зовнішнього вигляду і пред'являють до них особливі вимоги, але в основному вони звертають увагу на Material Design (концепція розробки, яка використовується в Android) (рис. 26)[5].

Рисунок 26 – Інтерфейс Android

4 ЗАСОБИ І ТЕХНОЛОГІЇ РОЗРОБКИ

4.1 Створення структурної моделі майбутнього додатку

Взаємодія між користувачем, робочою станцією та додатком представлена структурною моделлю (рис. 27). Користувач звертається до програми через робочу станцію, яка за допомогою внутрішніх компонентів та додаткових бібліотек формує єдиний функціональний інтерфейс для користувача.

Формований інтерфейс включає п'ять основних типів компонентів: кнопки, повзунки, листбокси, таймери і написи. Кнопки є об'єктами, на які програмуються основні функції аудіоплеєра.

Повзунки дозволяють змінювати числові значення певних функцій аудіоплеєра. Лістбокси містять інформацію про аудіо файли, таку як довжина треку, назва треку, шлях до треку і т.д.

Таймери контролюють правильне виконання функцій програми, такі як відтворення, зупинка, пауза і т.д.

Написи відповідають за зручність та зрозумілість інтерфейсу програми, включаючи підказки, назви функцій тощо.

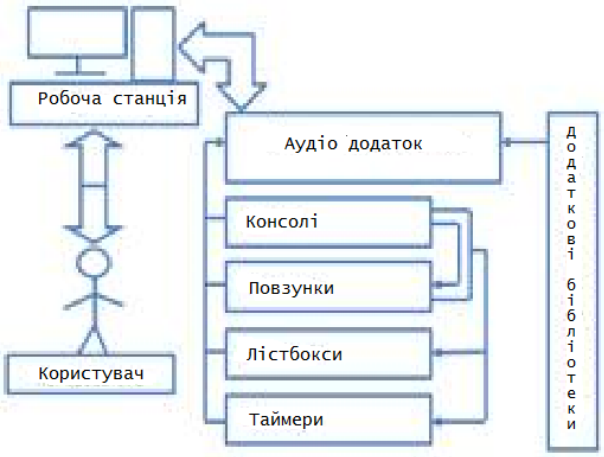

Рисунок 27 – Схема структурної моделі аудіоплеєра

4.2 Обґрунтування вибору середовища розробки

Android Studio – це інтегроване середовище розробки (IDE) для роботи з платформою Android, анонсована 16 травня 2013 року на конференції Google I/O.

IDE, починаючи з першого випуску в травні 2013 року з версією 0.1, була загальнодоступною. Бета-тестування почалося з версії 0.8 у червні 2014 року. Стабільна версія 1.0 була випущена в грудні 2014 року, що означало припинення підтримки плагіна Android Development Tools (ADT) для Eclipse.

Android Studio, офіційний інструмент розробки програм для Android, базується на програмному забезпеченні IntelliJ IDEA від JetBrains. Він доступний для операційних систем Windows, OS X і Linux. Під час щорічної конференції Google I/O, що відбулася 17 травня 2017 року, Google оголосив про підтримку Kotlin разом із Java і C++ як офіційної мови програмування для платформи Android на додаток до Android Studio.

Програмне забезпечення, яке використовується для розробки програм для ОС Android, створюється за допомогою Android SDK.

SDK (від англ. Software Development Kit) – набір із засобів розробки, утиліт і документації, який дозволяє програмістам створювати прикладні програми за визначеною технологією або для певної платформи (програмної або програмно-апаратної)[9].

```
buildscript {
    repositories {
        maven {
            url 'https://maven.google.com/'
            name 'Google'
        }
        jcenter()
        google()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:4.1.3'
        classpath           "org.jetbrains.kotlin:kotlin-gradle-
plugin:1.6.10"
```

```
    }}
allprojects {
    repositories {
        jcenter()
        maven {
            url 'https://maven.google.com/'
            name 'Google'
        }
        maven { url "https://jitpack.io" }
    }}
task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Цей код – файл збірки проекту для платформи Android. Він містить налаштування, які використовуються для налаштування проекту Android Studio та залежності, необхідні для збирання та виконання проекту.

Основні частини коду:

1. buildscript – плагін збирання, який задаєся для конкретних задач збирання проекту. Використовує:

– repositories – репозиторії, в яких шукаються залежності проекту. У цьому випадку, ми використовуємо кілька репозиторіїв, зокрема Google Maven та JCenter.

– dependencies – залежності бібліотек для плагіну збирання. В даному випадку, використовується плагін збирання для Android та плагін Kotlin.

2. allprojects – конфігурація, яка застосовується до всіх проектів в проекті Android Studio. В даному випадку, вона задає репозиторії, в яких шукаються залежності проектів.

3. task clean (type: Delete) – задача очищення, яка видаляє файли збірки проекту.

Цей код допомагає налаштувати залежності та репозиторії для проекту Android Studio з використанням плагіну збирання Gradle.

Коли розробник хоче працювати з певною технологією чи системою, він зазвичай отримує комплект розробки програмного забезпечення (SDK) безпосередньо від розробника. Ці SDK часто доступні для завантаження в

Інтернеті. Багато розробників пропонують свої SDK безкоштовно, щоб стимулювати інших використовувати їхню технологію чи платформу.

Android SDK – це універсальний інструмент, призначений для створення мобільних додатків для операційної системи Android. На відміну від звичайних редакторів коду, він надає широкий спектр функцій. Наприклад, це дозволяє розробникам тестувати та налагоджувати свій вихідний код, оцінювати продуктивність програми в різних версіях ОС Android і за бажанням переглядати результати в реальному часі. Android SDK підтримує різні мобільні пристрої, такі як смартфони, планшети, розумні окуляри (наприклад, Google Glass), сучасні автомобілі з бортовими комп'ютерами на базі Android та інші технічні гаджети.

Візуальний редактор макетів – це потужний інструмент, який дозволяє створювати складні макети за допомогою ConstraintLayout. Можна додати обмеження між видами та напрямними, забезпечуючи точне позиціонування. Також, є можливість попередньо переглянути макет на різних розмірах екрана, вибравши різні конфігурації пристрою або змінивши розмір вікна попереднього перегляду.

Аналізатор APK – це корисна функція, яка допомагає зменшити розмір програм Android. Це дозволяє перевіряти вміст файлу APK, навіть якщо він не був створений за допомогою Android Studio. Таким чином можна проаналізувати вміст програми та зробити оптимізацію, щоб зменшити її загальний розмір.

Швидкий емулятор – це зручний інструмент для швидкого встановлення та запуску програм порівняно з використанням фізичного пристрою. Він також дозволяє імітувати різні конфігурації та функції, включаючи ARCore, платформу Google для створення досвіду доповненої реальності. Емулятор дає змогу перевірити продуктивність програми в різних сценаріях без потреби у фізичних пристроях.

Інтелектуальний редактор коду розроблено для покращення досвіду кодування. Він надає пропозиції та доповнення коду для мов Kotlin, Java та

C++ / C. Використовуючи інтелектуальні функції, можна написати кращий код, збільшити швидкість і підвищити загальну продуктивність.

Профайлери в реальному часі – це вбудовані інструменти, які пропонують цінну інформацію про продуктивність вашої програми. Є можливість відстежувати статистику в режимі реального часу, пов’язану з використанням ЦП, пам’яті та мережевою активністю. Ці профайлери допомагають виявити вузькі місця продуктивності, записуючи трасування методів, перевіряючи купу та виділення, а також аналізуючи вхідні та вихідні дані мережі. Використовуючи профайлери, можна оптимізувати програму для кращої продуктивності.

Розширювана мова розмітки (широко відома як XML) — широко прийнятий стандарт, представлений Консорціумом Всесвітньої павутини, спрямований на полегшення створення мов розмітки, спеціально розроблених для організації ієрархічно структурованих даних. Цей стандарт був в основному розроблений для безперебійного обміну даними між різними програмами, з особливим акцентом на онлайн-спілкуванні. XML можна розглядати як спрощену та оптимізовану версію стандартної узагальненої мови розмітки (SGML), що зосереджується на основних елементах, необхідних для ефективного представлення даних.

Стандарт XML встановлює набір фундаментальних правил, що регулюють побудову мови, яка використовується для опису інформації за допомогою простих тегів. Цей формат забезпечує достатню гнучкість для обслуговування різноманітних галузей. Простіше кажучи, запропонований стандарт окреслює мета-мову, з якої можуть бути отримані специфічні предметно-орієнтовані мови розмітки даних шляхом накладення обмежень на структуру та зміст документів. Ці обмеження визначаються такими мовами Schema, як XML Schema (XSD), DTD або RELAX NG. Кілька прикладів мов на основі XML включають XSLT, XAML, XUL, RSS, MathML, GraphML, XHTML, SVG і XML Schema.

4.3 Опис роботи додатку

Для запуску додатку потрібно натиснути на ярлик (рис. 28).

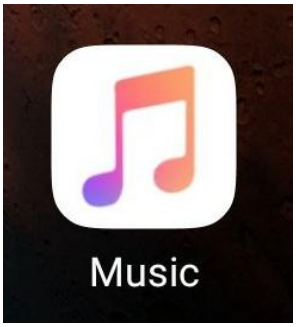

Рисунок 28 – Ярлик додатку

Після запуску додатку на екрані пристрою користувач бачить головний екран (рис. 29).

```
# Project-wide Gradle settings.
# IDE (e.g. Android Studio) users:
# Gradle settings configured through the IDE *will override*
# any settings specified in this file.
# For more details on how to configure your build environment
visit
#http://www.gradle.org/docs/current/userguide/build_environment.
html
# Specifies the JVM arguments used for the daemon process.
# The setting is particularly useful for tweaking memory
settings.
```

Цей код є файлом налаштувань Gradle для проекту. Gradle є системою збирання та автоматичної залежності, яка використовується для будівництва проектів на платформі Android. У цьому коді є декілька ключових налаштувань:

1. `android.enableJetifier=true`: Ця настройка дозволяє використовувати Jetifier, який автоматично конвертує залежності, що використовують бібліотеки підтримки, у їх відповідники в AndroidX. Це допомагає

забезпечити сумісність бібліотек з новішими версіями Android.

2. `android.useAndroidX=true`: Ця настройка вказує Gradle використовувати AndroidX бібліотеки замість застарілих бібліотек підтримки. AndroidX - це новий підхід до розробки Android, який замінює підтримку бібліотек. Використання AndroidX дозволяє отримати доступ до нових функцій і виправлення помилок.

3. `org.gradle.jvmargs=-Xmx1536m`: Ця настройка визначає аргументи JVM, які будуть використовуватися при запуску Gradle. У даному випадку, `-Xmx1536m` задає максимальний обсяг пам'яті, який може використовувати Gradle.

Ці налаштування дозволяють налаштувати поведінку Gradle при будівництві та залежності проекту на платформі Android.

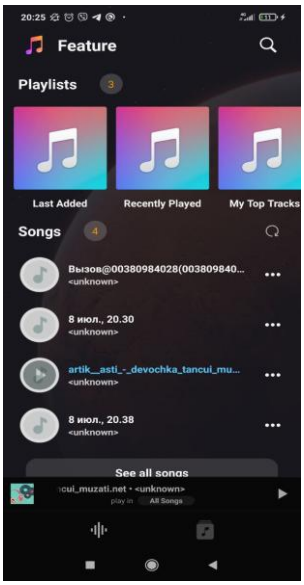

Рисунок 29 – Головний екран додатку

```
android.enableJetifier=true
android.useAndroidX=true
org.gradle.jvmargs=-Xmx1536m
# When configured, Gradle will run in incubating parallel mode.
# This option should only be used with decoupled projects. More
details, visit
#http://www.gradle.org/docs/current/userguide/multi_project_buil
ds.html#sec:decoupled_projects
```

```
# org.gradle.parallel=true
```

Цей код є частиною файла налаштувань Gradle для проекту Android. Він встановлює декілька параметрів конфігурації для Gradle:

1. `android.enableJetifier=true` - Цей параметр увімкнутий для Jetifier, який допомагає у дотриманні сумісності бібліотек, що використовують бібліотеки підтримки, з AndroidX. Jetifier перетворює вимоги бібліотек підтримки на відповідні вимоги AndroidX, що дозволяє використовувати бібліотеки, які ще не були оновлені до AndroidX.

2. `android.useAndroidX=true` - Цей параметр увімкнутий для використання AndroidX у проекті. AndroidX - це пакети бібліотек, які замінюють старий пакет бібліотек підтримки Android.

3. `org.gradle.jvmargs=-Xmx1536m` - Цей параметр встановлює обмеження пам'яті для виконання Java-коду Gradle. Тут обмеження становить 1536 мегабайт.

4. `org.gradle.parallel=true` - Цей параметр увімкнутий, якщо проект складається з декількох незалежних модулів. При увімкненні цього параметра Gradle виконуватиме компіляцію модулів паралельно, що може прискорити час збирання проекту.

Ці параметри дають можливість налаштувати Gradle для оптимальної роботи з проектом Android. Ad not mentioned.

Ці настройки дозволяють налаштувати поведінку Gradle при будівництві та залежності проекту на платформі Android.

У верхній правій частині вікна знаходиться іконка пошуку, яка при натисканні викликає рядок пошуку (рис. 30).

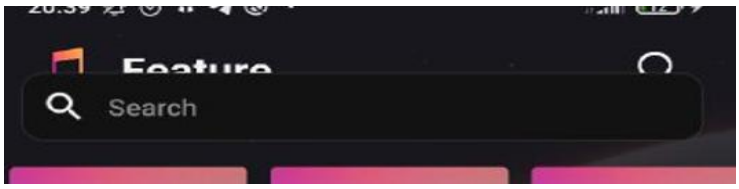

Рисунок 30 – Рядок пошуку

В першій частині інтерфейсу додатку знаходяться списки відтворення, які можна прокручувати вліво та вправо шляхом проведення пальцем.

За замовчуванням в додатку є 3 списки відтворення:

– "Останні додані" – цей список містить аудіофайли, які були знайдені останніми на вашому пристрої;

– "Нещодавно відтворені" – цей список містить аудіофайли з останньої черги відтворення;

– "Мої улюблені треки" – у цьому списку містяться аудіофайли, які були повністю прослухані більше 5 разів.

У наступній частині "Пісні" показується кількість аудіофайлів на вашому пристрої, а також 15 випадково обраних аудіофайлів. Щоб повторно вибрати файл, потрібно натиснути на кнопку оновлення у верхньому правому куті цієї частини. Кожен аудіофайл в списку (рис. 31) має такі відображення:

– зображення;

– назву аудіофайлу;

– виконавця;

– кнопку меню.

Рисунок 31 – Аудіофайл у списку

При натисканні на іконку меню (три крапки з правої сторони) з’являється меню (рис. 32), що дозволяє:

– встановити аудіозапис наступним у черзі програвання;

– програти коротке прев’ю (10 секунд з середини треку);

– програти повне прев’ю (30 секунд з середини треку);

– додати аудіозапис у чергу програвання;

– додати аудіозапис у плейлист (існуючий або створивши новий, окрім плейлистів, що створені за замовченням);

– перейти до усіх аудіозаписів виконавця;

видалити аудіофайл з пристрою.

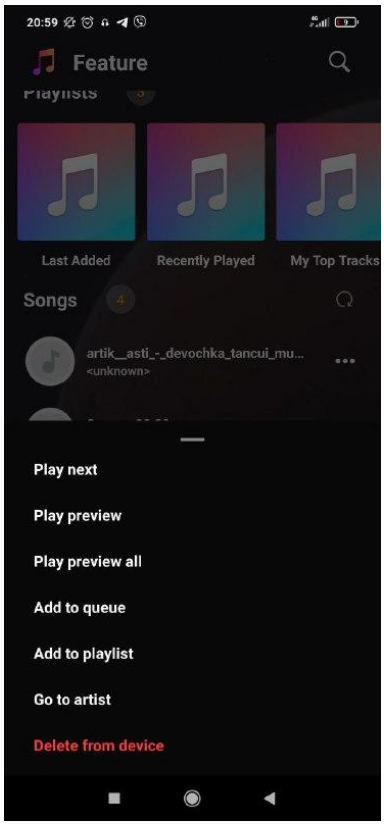

Рисунок 32 – Меню аудіо файлу

```
while [ -h "$PRG" ] ; do
    ls=`ls -ld "$PRG"`
    link=`expr "$ls" : '.*-> \(.*\)$'`
    if expr "$link" : '/.*' > /dev/null; then
        PRG="$link"
    else
        PRG=`dirname "$PRG"`"/$link"
    fi
done
SAVED="`pwd`"
cd "`dirname \"$PRG\"`/" >/dev/null
APP_HOME="`pwd -P`"
cd "$SAVED" >/dev/null

APP_NAME="Gradle"
```

```
APP_BASE_NAME=`basename "$0"`
```

Цей код є скриптом для визначення шляху до виконуваного файлу із символічним посиланням.

Основною метою цього коду є отримання шляху до скрипту, який викликається, навіть якщо він має символічне посилання. Код виконує перевірку, чи існує символічне посилання для виконуваного файла, і якщо так, то знаходить його шлях, доки не знайде фактичний файл.

У кінці цього коду змінні `APP_HOME`, `APP_NAME`, і `APP_BASE_NAME` містять шлях до каталогу, в якому знаходиться скрипт, назву виконуваного файлу та базове ім'я виконуваного файлу відповідно.

```
      # Add default JVM options here. You can also use JAVA_OPTS and
GRADLE_OPTS to pass JVM options to this script.
      DEFAULT_JVM_OPTS=""

      # Use the maximum available, or set MAX_FD != -1 to use that
value.
      MAX_FD="maximum"

      warn () {
          echo "$*"}
      die () {
          echo
          echo "$*"
          echo
          exit 1}
```

Цей код представляє собою shell-скрипт (bash-скрипт), який використовується для налаштування параметрів запуску Java Virtual Machine (JVM) та деяких внутрішніх функцій. Основні компоненти цього коду:

1. DEFAULT_JVM_OPTS: Змінна, яка визначає значення параметрів JVM за замовчуванням. За допомогою цієї змінної можна передати параметри JVM до скрипту, використовуючи JAVA_OPTS або GRADLE_OPTS.

2. MAX_FD: Змінна, яка визначає максимальну кількість доступних

файлових дескрипторів. Зазвичай використовується значення "maximum" для використання максимально доступної кількості дескрипторів.

3. warn() та die(): Функції для виводу повідомлень у консоль. Функція warn() просто виводить передане їй повідомлення, а функція die() виводить передане повідомлення, а потім завершує виконання скрипту з кодом помилки 1.

```
# OS specific support (must be 'true' or 'false').
cygwin=false
msys=false
darwin=false
nonstop=false
case "`uname`" in
  CYGWIN* )
    cygwin=true
    ;;
  Darwin* )
    darwin=true
    ;;
  MINGW* )
    msys=true
    ;;
  NONSTOP* )
    nonstop=true
    ;;
esac
```

Цей код перевіряє операційну систему, на якій він запускається, і встановлює відповідні значення змінних, що вказують на тип операційної системи. Змінні cygwin, msys, darwin, і nonstop приймають значення true або false в залежності від типу операційної системи.

Якщо виконується на операційній системі типу Cygwin, змінна cygwin отримує значення true, інші змінні залишаються зі значеннями false. Аналогічно, для операційної системи типу Darwin (MacOS), змінна darwin отримує значення true, а для операційної системи типу MINGW (наприклад, MSYS або Git Bash на Windows), змінна msys отримує значення true.

Цей код може бути використаний для визначення поведінки скрипту або програми залежно від типу операційної системи, на якій він запускається.

```
      CLASSPATH=$APP_HOME/gradle/wrapper/gradle-wrapper.jar

      # Determine the Java command to use to start the JVM.
      if [ -n "$JAVA_HOME" ] ; then
          if [ -x "$JAVA_HOME/jre/sh/java" ] ; then
              #  IBM's  JDK  on  AIX  uses  strange  locations  for  the
executables
              JAVACMD="$JAVA_HOME/jre/sh/java"
          else
              JAVACMD="$JAVA_HOME/bin/java"
          fi
          if [ ! -x "$JAVACMD" ] ; then
              die "ERROR: JAVA_HOME is set to an invalid directory:
$JAVA_HOME
      Please set the JAVA_HOME variable in your environment to match
the
      location of your Java installation."
          fi
      else
          JAVACMD="java"
          which java >/dev/null 2>&1 || die "ERROR: JAVA_HOME is not
set and no 'java' command could be found in your PATH.
```

Цей код встановлює CLASSPATH для програми, а також визначає команду, яку треба використовувати для запуску JVM.

Далі код перевіряє, чи є змінна JAVA_HOME встановлена, і в залежності від того, де знаходяться виконувані файли Java, встановлює змінну JAVACMD відповідно. Якщо JAVA_HOME не встановлена або шлях до виконуваного файлу java не знайдено в змінній PATH, генерується помилка. Основна мета цього коду - встановити вірний шлях до виконуваного Java-файлу для подальшого запуску JVM.

```
      Please set the JAVA_HOME variable in your environment to match
the
      location of your Java installation."
      # Increase the maximum file descriptors if we can.
      if [ "$cygwin" = "false" -a "$darwin" = "false" -a "$nonstop" =
"false" ] ; then
          MAX_FD_LIMIT=`ulimit -H -n`
          if [ $? -eq 0 ] ; then
              if [ "$MAX_FD" = "maximum" -o "$MAX_FD" = "max" ] ; then
                  MAX_FD="$MAX_FD_LIMIT"
              fi
```

```
            ulimit -n $MAX_FD
            if [ $? -ne 0 ] ; then
                warn "Could not set maximum file descriptor limit:
$MAX_FD"
            fi
        else
            warn "Could not query maximum file descriptor limit:
$MAX_FD_LIMIT"
```

Цей код є скриптом для налаштування оточення розробки в Java.

У першому рядку коду повідомляється, що потрібно встановити змінну JAVA_HOME у середовищі так, щоб вона вказувала на місцезнаходження вашої встановленої Java.

Наступні рядки коду виконують перевірку можливості збільшити максимальне значення файлових дескрипторів (file descriptors) для поточного користувача. В цьому випадку перевіряються різні умови, такі як не працюють чи не у OS/2, macOS (darwin) та NonStop. Якщо вони виконуються, то спробує встановити максимальну кількість файлових дескрипторів, використовуючи команду ulimit. Також, якщо змінна MAX_FD має значення "maximum" або "max", то заміняє її на MAX_FD_LIMIT.

Якщо скрипт не в змозі встановити максимальну кількість файлових дескрипторів або отримати її значення, видається відповідне попередження.

```
    # For Darwin, add options to specify how the application appears
in the dock
    if $darwin; then
        GRADLE_OPTS="$GRADLE_OPTS    \"-Xdock:name=$APP_NAME\"    \"-
Xdock:icon=$APP_HOME/media/gradle.icns\""
    fi

    # For Cygwin, switch paths to Windows format before running java
    if $cygwin ; then
        APP_HOME=`cygpath --path --mixed "$APP_HOME"`
        CLASSPATH=`cygpath --path --mixed "$CLASSPATH"`
        JAVACMD=`cygpath --unix "$JAVACMD"`
```

Цей код містить скрипти, які виконуються в залежності від операційної системи, на якій виконується код.

Перший блок коду встановлює додаткові параметри для Grandle. Ці параметри задають ім'я та іконку, які будуть використовуватися для додатку

в доку MacOS.

Другий блок коду перетворює шляхи до файлів у форматі Windows, якщо операційна система - Cygwin. Це може бути необхідно для правильного функціонування Java на цій системі.

```
          # We build the pattern for arguments to be converted via
cygpath
          ROOTDIRSRAW=`find -L / -maxdepth 1 -mindepth 1 -type d
2>/dev/null`
          SEP=""
          for dir in $ROOTDIRSRAW ; do
              ROOTDIRS="$ROOTDIRS$SEP$dir"
              SEP="|"
          done
          OURCYGPATTERN="(^($ROOTDIRS))"
          # Add a user-defined pattern to the cygpath arguments
          if [ "$GRADLE_CYGPATTERN" != "" ] ; then
              OURCYGPATTERN="$OURCYGPATTERN|($GRADLE_CYGPATTERN)"
          fi
          # Now convert the arguments - kludge to limit ourselves to
/bin/sh
          i=0
          for arg in "$@" ; do
              CHECK=`echo "$arg"|egrep -c "$OURCYGPATTERN" -`
              CHECK2=`echo        "$arg"|egrep        -c        "^-"`
### Determine if an option
```

Цей код виконує наступні дії:

1. Знаходить всі теки першого рівня від кореневого каталогу ("/") і зберігає їх у змінну `ROOTDIRSRAW`. Відповідна команда `find` використовується з параметрами `-L` (дотримуватись символічних посилань) та `-maxdepth 1 -mindepth 1 -type d` (шукати тільки теки першого рівня).

2. У циклі проходимо через кожну теку зі змінної `ROOTDIRSRAW`, додаємо її до змінної `ROOTDIRS`, розділяючи кожну теку символом `|`. Таким чином, у змінній `ROOTDIRS` зберігаються всі теки першого рівня, розділені символом `|`.

3. Формуємо шаблон `OURCYGPATTERN`, який містить регулярний вираз для визначення шляхів, що відповідають текам першого рівня. Використовуємо значення змінної `ROOTDIRS` для побудови виразу.

4. Якщо змінна `GRADLE_CYGPATTERN` не порожня, додаємо його значення до шаблону `OURCYGPATTERN`, розділяючи символом `|`.

5. У циклі проходимо через кожен аргумент командного рядка (`$@`) і перевіряємо, чи відповідає він шаблону `OURCYGPATTERN`. Для цього використовується команда `egrep -c`, яка повертає кількість співпадінь. Результат перевірки зберігається у змінній `CHECK`.

6. Також перевіряється, чи аргумент є опцією командного рядка, тобто починається з символу `-`. Результат перевірки зберігається у змінній `CHECK2`.

Код виконує різні перевірки і формує шаблон для конвертації шляхів з використанням програми `cygpath`. Зрозуміло, що цей код міститься в скрипті або файлі шелл-сценарію. Точна дія, що відбувається з результатами перевірок і параметрами `cygpath`, залежить від контексту використання і налаштувань середовища.

```
              if [ $CHECK -ne 0 ] && [ $CHECK2 -eq 0 ] ; then
### Added a condition
                  eval `echo args$i`=`cygpath --path --ignore --mixed
"$arg"`
              else
                  eval `echo args$i`="\"$arg\""
              fi
              i=$((i+1))
            save () {
                for    i    do    printf    %s\\n    "$i"    |    sed
     "s/'/'\\\\''/g;1s/^/'/;\$s/\$/' \\\\/" ; done
          echo " "
          done
              OURCYGPATTERN="$OURCYGPATTERN|($GRADLE_CYGPATTERN)"
          fi
          # Now convert the arguments - kludge to limit ourselves to
/bin/sh
          i=0
          for arg in "$@" ; do
              CHECK=`echo "$arg"|egrep -c "$OURCYGPATTERN" -`
              CHECK2=`echo        "$arg"|egrep        -c        "^-"`
### Determine if an option
```

Цей код є фрагментом скрипту або програми. Основна функція коду - це перетворити аргументи командного рядка і зберегти їх у змінні.

Умова `if [ $CHECK -ne 0 ] && [ $CHECK2 -eq 0 ]` перевіряє, чи значення змінних `$CHECK` та `$CHECK2` відповідають певним умовам. Якщо ця умова виконується, виконується блок коду після `then`, в іншому випадку виконується блок коду після `else`.

У блоку коду після `then` відбувається наступне:

– команда `eval` виконує вираз, що передається в аргументі. В даному випадку, виконується команда `echo args$i`=`cygpath --path --ignore --mixed "$arg"`, де `$i` та `$arg` є значеннями змінних, а `args` - префікс для назви змінної, яка буде створена через `eval`.

– команда `cygpath --path --ignore --mixed "$arg"` перетворює шлях файлу в формат, зрозумілий для системи Cygwin.

– результат виразу `cygpath --path --ignore --mixed "$arg"` присвоюється змінній з ім'ям `args$i`.

У блоку коду після `else` відбувається наступне:

– команда `eval` виконує вираз, що передається в аргументі. В даному випадку, виконується команда `echo args$i`="\"$arg\""`, де `$i` та `$arg` є значеннями змінних, а `args` - префікс для назви змінної, яка буде створена через `eval`.

– змінній з ім'ям `args$i` присвоюється значення `"$arg"`.

– також в коді є оголошення функції `save()`. Ця функція приймає аргументи `i`, які використовуються для перебору аргументів командної стрічки. Функція виводить кожен аргумент на новому рядку за допомогою команди `printf`, а також додає крапку та пробіл після останнього аргументу за допомогою команди `echo`. Всі символи одинарніх лапок `'` в тексті аргументів замінюються на `'\"'` за допомогою команди `sed`.

Щоб переглянути усі аудіозаписи на пристрої потрібно натиснути кнопку «See all songs», після чого відбудеться перехід до вікна бібліотеки аудіозаписів (рис. 33).

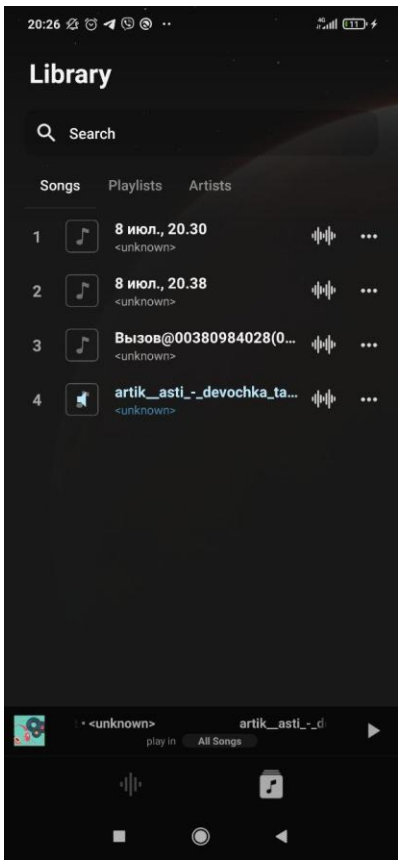

Рисунок 33 – Бібліотека аудіо записів

У бібліотеці аудіозаписів знаходяться три вкладки, які мають наступні назви і функції:

– "Пісні" – тут можна знайти всі аудіозаписи, які зберігаються на пристрої, вони відсортовані за назвою.

– "Плейлисти" – ця вкладка містить аудіозаписи, впорядковані за плейлистами.

– "Виконавці" – тут аудіозаписи впорядковані за виконавцями.

У підвалі вікна є дві частини:

– показується зменшене вікно плеєра, яке відображає назву поточно програваного аудіозапису. Натиснувши на цю назву, можна перейти до повноекранного вікна плеєра;

– є також можливість переміщатися між головним вікном і вікном бібліотеки аудіозаписів. (рис. 34).

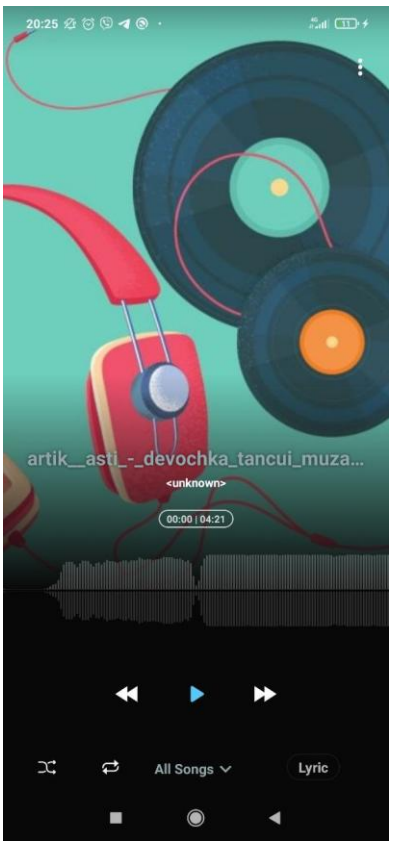

Рисунок 34 – Вікно плеєру

```
@rem Set local scope for the variables with windows NT shell
if "%OS%"=="Windows_NT" setlocal
set DIRNAME=%~dp0
if "%DIRNAME%" == "" set DIRNAME=.
set APP_BASE_NAME=%~n0
set APP_HOME=%DIRNAME%
@rem Add default JVM options here. You can also use JAVA_OPTS
and GRADLE_OPTS to pass JVM options to this script.
set DEFAULT_JVM_OPTS=
@rem Find java.exe
if defined JAVA_HOME goto findJavaFromJavaHome
set JAVA_EXE=java.exe
%JAVA_EXE% -version >NUL 2>&1
if "%ERRORLEVEL%" == "0" goto init
```

Цей код є скриптом командного рядка і виконує наступні дії:

1. Встановлює локальний контекст для змінних у системному шелі.

2. Визначає змінну `DIRNAME`, яка містить шлях до директорії, в якій знаходиться сценарій.

3. Перевіряє, чи `DIRNAME` не порожній рядок і встановлює його як

поточну директорію, якщо він є порожнім.

4. Визначає змінну `APP_BASE_NAME`, яка містить назву основного файлу (без розширення).

5. Визначає змінну `APP_HOME`, яка містить значення `DIRNAME`.

6. Визначає змінну `DEFAULT_JVM_OPTS`, яка може містити додаткові опції JVM (Java Virtual Machine).

7. Пошук встановленого JVM. Якщо змінна `JAVA_HOME` визначена, то відбувається пошук Java з використанням шляху з `JAVA_HOME`. Інакше, використовується значення `java.exe` (у випадку, якщо воно знаходиться в змінній середовища `PATH`).

8. Якщо Java знайдений, то виводиться його версія і виконується маркерний рядок `init`.

9. Даний код використовується для ініціалізації змінних середовища та встановлення опцій JVM перед виконанням певної Java-програми чи іншого скрипта.

```
      echo.
      echo ERROR: JAVA_HOME is not set and no 'java' command could be
found in your PATH.
      echo.
      echo Please set the JAVA_HOME variable in your environment to
match the
      echo location of your Java installation.
      goto fail
      :findJavaFromJavaHome
      set JAVA_HOME=%JAVA_HOME:"=%
      set JAVA_EXE=%JAVA_HOME%/bin/java.exe
      if exist "%JAVA_EXE%" goto init
      echo.
      echo ERROR: JAVA_HOME is set to an invalid directory:
%JAVA_HOME%
      echo.
      echo Please set the JAVA_HOME variable in your environment to
match the
      echo location of your Java installation.
      goto fail
      :init
```

Цей код перевіряє наявність змінної оточення JAVA_HOME та

перевіряє, чи існує команда "java" у шляху. Якщо змінна JAVA_HOME не встановлена або команда "java" не знайдена, з'являється повідомлення про помилку, яке просить користувача встановити змінну JAVA_HOME у своєму середовищі та вказати шлях до установлення Java.

Якщо змінна JAVA_HOME встановлена, код перевіряє, чи існує файл java.exe у визначеному шляху. Якщо файл існує, код продовжує своє виконання, якщо ні – з'являється повідомлення про помилку і рекомендація встановити коректний шлях до установлення Java.

Цей код скоріш за все належить до скрипта або пакету, пов'язаного з встановленням та налаштуванням Java на комп'ютері.

```
@rem Get command-line arguments, handling Windows variants
if not "%OS%" == "Windows_NT" goto win9xME_args
:win9xME_args
@rem Slurp the command line arguments.
set CMD_LINE_ARGS=
set _SKIP=2
:win9xME_args_slurp
if "x%~1" == "x" goto execute
set CMD_LINE_ARGS=%*
:execute
@rem Setup the command line
set CLASSPATH=%APP_HOME%\gradle\wrapper\gradle-wrapper.jar
@rem Execute Gradle
"%JAVA_EXE%" %DEFAULT_JVM_OPTS% %JAVA_OPTS% %GRADLE_OPTS% "-
Dorg.gradle.appname=%APP_BASE_NAME%" -classpath "%CLASSPATH%"
org.gradle.wrapper.GradleWrapperMain %CMD_LINE_ARGS%
:end
```

Цей код є скриптом для виклику і виконання програми Gradle.

1. `@rem Get command-line arguments, handling Windows variants` - Ця команда пропускає наступний рядок, якщо операційна система не є Windows_NT.

2. `:win9xME_args` - Ця мітка використовується для зазначення місця, куди можна перейти, якщо операційна система є Windows 9x або ME.

3. `@rem Slurp the command line arguments.` - Цей рядок пропускає наступні рядки із зчитуванням аргументів командного рядка.

4. `set CMD_LINE_ARGS=` - Цей рядок встановлює змінну CMD_LINE_ARGS на порожній рядок.

5. `set _SKIP=2` - Цей рядок встановлює змінну _SKIP на значення 2.

6. `:win9xME_args_slurp` - Ця мітка використовується для зазначення місця, куди можна перейти для зчитування аргументів командного рядка в операційних системах Windows 9x або ME.

7. `if "x%~1" == "x" goto execute` - Ця команда перевіряє, чи є перший аргумент командного рядка порожнім. Якщо так, вона переходить до мітки `execute`, інакше виконується наступний рядок.

8. `set CMD_LINE_ARGS=%*` - Цей рядок встановлює змінну CMD_LINE_ARGS на всі аргументи командного рядка.

9. `:execute` - Ця мітка використовується для зазначення місця, куди можна перейти після зчитування аргументів командного рядка.

10. `set CLASSPATH=%APP_HOME%\gradle\wrapper\gradle-wrapper.jar` - Цей рядок встановлює змінну CLASSPATH на шлях до файлу gradle-wrapper.jar.

11. `"%JAVA_EXE%" %DEFAULT_JVM_OPTS% %JAVA_OPTS% %GRADLE_OPTS% "-Dorg.gradle.appname=%APP_BASE_NAME%" -classpath "%CLASSPATH%" org.gradle.wrapper.GradleWrapperMain %CMD_LINE_ARGS%` - Цей рядок викликає програму Gradle з використанням встановлених опцій та переданих аргументів командного рядка.

Цей скрипт служить для виконання команд Gradle для будь-якого проекту, який використовує Gradle для збирання і управління залежностями. Він забезпечує виконання Gradle з конфігураціями, які зазначені у скрипті, і з аргументами командного рядка, переданими при запуску скрипту.

```
@rem End local scope for the variables with windows NT shell
if "%ERRORLEVEL%"=="0" goto mainEnd
:fail
rem Set variable GRADLE_EXIT_CONSOLE if you need the _script_ return code instead of
```

```
rem the _cmd.exe /c_ return code!
if  not "" == "%GRADLE_EXIT_CONSOLE%" exit 1
exit /b 1
:mainEnd
if "%OS%"=="Windows_NT" endlocal
:omega
```

Цей код є скриптом для виконання Gradle, який є інструментом збирання та управління проектами Java. Основна мета цього скрипта – запустити Gradle з необхідним JVM, шляхом до якого визначається змінною середовища JAVA_HOME.

i. Основні кроки, які виконує цей скрипт:

2. Встановлює локальний обсяг для змінних в середовищі Windows NT.

2. Визначає шлях до каталогу, в якому знаходиться скрипт.

3. Встановлює змінні, що описують ім'я базового додатка, домашній каталог додатка і т.д.

4. Пошук java.exe з використанням JAVA_HOME (якщо визначено) або шляхом за замовчуванням.

5. Ініціалізує змінні середовища та перевіряє, чи може виконатися команда 'java' з поточного середовища.

6. Встановлює змінні середовища, пов'язані з параметрами JVM.

7. Встановлює CLASSPATH для Gradle.

8. Виконує команду Gradle, передаючи необхідні параметри, шляхи класів і аргументи командного рядка.

9. Завершує виконання скрипта з відповідним кодом повернення.

10. Цей скрипт полегшує використання та автоматизацію Gradle-проектів в середовищі Windows NT.

Додаток має можливість працювати в фоновому режимі. Щоб не відчиняти вікно додатку для керування програванням достатню відкрити панель повідомлень Android, де знаходиться панель плеєру (рис. 35).

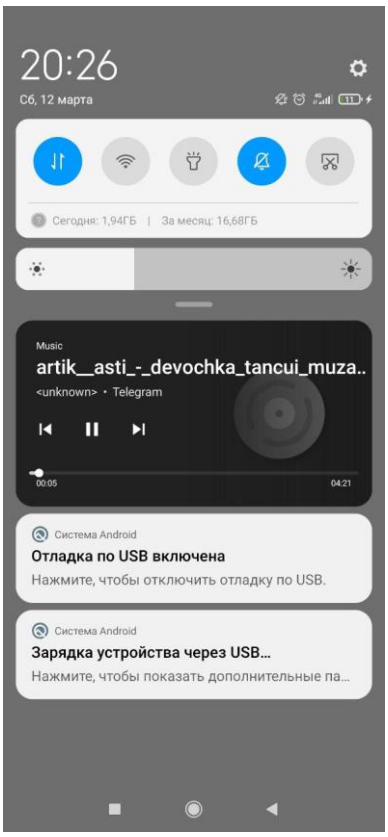

Рисунок 35 – Панель плеєру у панелі повідомлень Android

ВИСНОВКИ

Розробка мобільного додатку – це досить тривалий і трудомісткий процес. Не існує готового сценарію, і для даного етапу потрбен план розробки мобільного застосування, який може змінюватися, і коригуватися в залежності від проекту. Підсумуюючи тему, можна зробити наступні висновки:

− високий попит на мобільні аудіоплеєри: враховуючи зростання кількості користувачів мобільних пристроїв, створення аудіоплеєра для платформи Android є перспективним і вигідним проектом.

− функціональні можливості: мобільний додаток аудіоплеєра повинен мати базовий набір функцій, таких як відтворення музики, створення плейлистів, перемотування треків і налаштування еквалайзера. Однак, можна також додати додаткові функції, такі як підтримка бездротової передачі музики на інші пристрої, інтеграція зі стрімінговими сервісами або підтримка синхронізації з хмарними сховищами.

− інтерфейс та дизайн: користувацький інтерфейс додатку повинен бути зрозумілим та зручним для використання. Дизайн може бути орієнтованим на матеріальний дизайн, щоб відповідати стандартам Android, але також може мати свою унікальну концепцію, яка відповідає цілям додатку.

− оптимізація та продуктивність: додаток повинен бути оптимізований для роботи на різних пристроях і версіях Android, забезпечуючи стабільну роботу і мінімальне споживання ресурсів пристрою.

− тестування та відлагодження: важливим етапом розробки є ретельне тестування додатку на різних пристроях і налаштуваннях, щоб виявити та виправити помилки і недоліки перед випуском на ринок.

− безпека та захист: додаток повинен забезпечувати захист інформації користувача, так як у нього можуть бути особисті дані (логіни та паролі для підключення до стрімінгових сервісів або власних музичних бібліотек).

Отже, розробка мобільного додатку аудіоплеєра для платформи Android може бути вигідним та цікавим проектом, вимагаючим уваги до деталей, якості та користувацького досвіду. Забезпечення функціональності, ергономіки, оптимізації та безпеки повинно бути в основі розробки такого додатку.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1 Статистика Datareportal URL: https://ceusnj.org/2022/12/12/rozrobka-vlasnogo-mobil%D1%8Cnogo-dodatku-pid-android/ (Дата звернення 05.05.2023)

2 Як стати Андроїд-розробником? URL: https://vstbaz.com/jakstati-android-rozrobnikom-znulja-napal%D1%8Ccjah/ (Дата звернення 05.05.2023)

3 Розробка мобільних додатків від А до Я URL: https://dan-it.com.ua/uk/blog/rozrobka-mobilnih-dodatkiv-vid-a-do-ja-povnij-gajd/ (Дата звернення 058.05.2023)

4 ХНУМГ ім. О.М. Бекетова 2020 «XIII Всеукраїнської студентської науково-технічної конференції «Сталий розвиток міст»» URL: https://science.kname.edu.ua/images/dok/konferentsii/2020konf/4_2020.pdf (Дата звернення 04.05.2023)

5 Інтегроване середовище розробки URL: https://uk.unionpedia.org/ (Дата звернення 10.05.2023)

6 Наукова конференція молодих вчених ОДЕКУ URL: blob:https://web.telegram.org/2747359c-0b4e-49bd-a3e7-c800463c54bd (Дата звернення 05.05.2023)

7 Проблеми інформатизації навчального процесу в школі та вищому педагогічному навчальному закладі URL: https://enpuir.npu.edu.ua/bitstream/handle/123456789/38675/20informatyzatsii%20navchalnoho%20protsesu.pdf?sequence=1&isAllowed=y (Дата звернення 07.05.2023)

8 Збірник наукових праць студентів Кам’янець-Подільського національного університету імені Івана Огієнка URL: http://elar.kpnu.edu.ua:8081/123456789/3373/Zbirnyk.naukovykh.prats.studentiv.tamahistrantiv.KPNU.im.I.OhiienkaVyp.13.pdf?sequence=1&isAllowed=y (Дата звернення 09.05.2023)

9 Вчені записки ТНУ ім. В.І. Вернадського URL: https://vernadsky.in.ua/1_2022.pdf (Дата звернення 06.05.2023)