adrumdrum

Arkitekturbeskrivning

Projektet adrummdrum innehåller 6 stycken olika paket

1 kvhc.adrumdrum

Paketet adrumdrum innehåller bara programmets main klass.

1.1 MainActivity

Programmets startpunkt. Här görs allt som krävs för att appen ska starta

2 kvhc.gui

Här finns de klasser som används för att skapa ett GUI. Här finns det även klasser som används för att uppdatera och ändra i GUI:t

2.1 GUIController

Kontrollerar det mesta i programmet. Det är här allt som har med GUI:t att göra instansieras. Det är den här klassen som reagerar på input från användaren och ser till att rätt åtgärder utförs. Den här klassen skapar också nya songs och laddar in dem i en player.
Viktiga metoder:

- removeChannel
- toggleSolo
- reloadSong
- redrawChannels
- clearAllSteps

2.2 ChannelButtonGUI

En klass som ärver Button. GUI för att representera en knapp som man kan klicka på för att få upp kanalinställningar där man kan ändra volym och ljud.
Viktiga metoder:

- updateName

2.3 ChannelMuteButton

En klass som ärver Button. Klickar man på knappen så ska kanalen byta mellan att vara mutad eller inte. Ska även innehålla en ikon som reflekterar över hur hög volym kanalen har.

2.4 GUIStepButton

En klass som beskriver hur ett step ska se ut. Sköter aktivering och avaktivering av ett step när den blir klickad på. Vid ett ”långt” klick visas en StepDialog
Viktiga metoder:

- getChannelId
- getStepId
- getStep
- reverse

2.5 GUIUpdateObsever

En Observer som updaterar GUI:t med information som skickas av player
Viktiga metoder:

- Update

3 kvhc.gui.dialogs

I paketet dialogs ligger alla dialoger som används av programmet.

3.1 ChannelDialog

En dialog över olika kanalinställningar

3.2 StepDialog

En dialog som ändrar värden på ett Step. Startas av en GUIStepButton

3.3 SavaSongDialog

En dialog där användaren kan ange ett namn på låten som de vill spara

3.4 LoadSongDialog

En dialog där användaren kan se vilka låtar som finns sparade i en lista och välja vilken låt som ska laddas genom att klicka på den.

4 kvhc.model

Paketet player innehåller de klasser som styr spelaren och

4.1 Song

Song används för att modellera antalet kanaler och hur många steg varje kanal har.
Viktiga metoder:

- addChannel
- removeChannel
- getNumberOfChannels
- addSteps
- removeSteps
- getNumberOfSteps
- clearAllSteps
- getSounds

4.2 Channel

En Channel innehåller ett visst antal Steps. I en Channel finns det även variabler och metoder för att ändra kanalvolym, uppspelningsljud och panning
Viktiga metoder:

- setStep
- toggleStep
- getStepAt
- getSteps
- getNumberOfSteps
- resizeBy

4.3 Step

Ett Step är en modell för en ruta som man kan markera som aktiv eller inte. Step innehåller också ett värde för hur hög ljuduppspelningen ska vara.
Viktiga metoder:

- setActive
- isActive
- reset
- getId
- clone

4.4 Sound

Ett sound är en ”Wrapper klass” för ett ljud
Viktiga metoder:

- getId
- getSoundValue
- getName

5 kvhc.util

I paketet util finns de generell klasser som modellerar de verktyg som vi behöver för adrumdrum

5.1 Player

Player spelar upp en angiven Song. Player sköter vad som ska spelas upp och när det ska göras. En Player har en SoundManager som sköter uppspelningen av ljud Viktiga metoder

- play
- stop
- isPlaying
- setWaitTimeByBPM
- getActiveStep

5.2 SoundManager

SoundManager sköter all uppspelning av ljud
Viktiga metoder:

- playSound

5.3 AndroidTimer

En timer som kör en Runnable varje gång som "tiden går ut". Används av player för att styra tiden mellan varje uppspelning av nästa steg.
Viktiga metoder:

- start
- stop
- running
- changeJob

5.4 AssetManagerModel

En klass för att hantera ljud. Klassen är en singelton.
Viktiga metoder:

- getInstance

5.5 ISongLoader

Ett interface för att ladda en låt. Används för att kunna spela upp låten med olika bibliotek eller på olika sätt.

5.6 ISongRenderer

Ett interface för att rendera en song.

5.7 SoundPoolRenderer

Implementerar ISongRenderer. Renderar ljud och spelar upp dem genom att använda en SoundManager
Viktiga metoder:

- renderSong
- renderSongAtStep

6 kvhc.util.db

util.db innehåller de klasser som används för att skriva och läsa från databasen

6.1 SQLRenderer

Sparar en Song med allt innehåll (Channels och Steps, properties) till databasen. Sparar även ljud till databasen (loadSound sparar, olyckligt).

6.2 SQLSongLoader

Laddar en Song med allt innehåll (Channels och Steps, properties) från databasen. Laddar även ljud från databasen.

6.3 SoundSQLiteHelper

ärver från SQLiteOpenHelper. Klass som innehåller tabelldata för Sound. Skapar tabellen om den inte finns. Hanterar databas uppkoppling så att man kan köra queries mot databasen.
Kolumner:

- _ id
- sound_ value
- name

6.4 SoundDataSource

Data Access Object för Sound. Sparar, laddar och tar bort Sound-objekt från databasen. Viktigt: Kom ihåg att köra open() innan du använder någon metod och close() efter att du är klar, annars är det inte möjligt att köra queries mot databasen.
Viktiga metoder:

- save
- getAllSongs
- deleteSong

6.5 SongSQLiteHelper

ärver från SQLiteOpenHelper. Klass som innehåller tabelldata för Sound. Skapar tabellen om den inte finns. Hanterar databas uppkoppling så att man kan köra queries mot databasen.
Kolumner:

- _ id
- name
- bpm

6.6 SongDataSource

Data Access Object för Song. Sparar, laddar och tar bort Song-objekt från databasen. Viktigt: Kom ihåg att köra open() innan du använder någon metod och close() efter att du är klar, annars är det inte möjligt att köra queries mot databasen.
Viktiga metoder:

- save
- getAllSounds
- getSoundFromKey
- deleteSound

6.7 ChannelSQLiteHelper

ärver från SQLiteOpenHelper. Klass som innehåller tabelldata för Sound. Skapar tabellen om den inte finns. Hanterar databas uppkoppling så att man kan köra queries mot databasen.
Kolumner:

- _ id
- number
- volume

- leftpan
- rightpan
- mute

Foregn keys:

- song_ id
- sound_ id

6.8 ChannelDataSource

Data Access Object för Channel. Sparar, laddar och tar bort Channel-objekt från databasen. Viktigt: Kom ihåg att köra open() innan du använder någon metod och close() efter att du är klar, annars är det inte möjligt att köra queries mot databasen.
Viktiga metoder:

- save
- getAllChannelsForSong
- deleteChannel

6.9 StepSQLiteHelper

ärver från SQLiteOpenHelper. Klass som innehåller tabelldata för Sound. Skapar tabellen om den inte finns. Hanterar databas uppkoppling så att man kan köra queries mot databasen.
Kolumner:

- _ id
- number
- active
- velocity

Foregn keys:

- channel_ id

6.10 StepDataSource

Data Access Object för Channel. Sparar, laddar och tar bort Channel-objekt från databasen. Viktigt: Kom ihåg att köra open() innan du använder någon metod och close() efter att du är klar, annars är det inte möjligt att köra queries mot databasen.
Viktiga metoder:

- save
- getAllStepsForChannel
- deleteStep