Period 1
Jalen Chen
Sound Squares
Final Document

Description
A rhythm game where a song plays and squares approach you in a 3D-like space. The notes match up with the song. You have to track and hit squares using your cursor. Maintain high accuracy and a high combo to get a higher score. Like Rhythia or Sound Space.

Functionalities:
- Mapping (table of coordinates of squares for each note in the song)
- Song browser to select a song to play
- Increase speed setting = harder = higher score multiplier
- Decrease speed setting = easier = lower score multiplier
- A "click" sound is played when a square is successfully hit, and an "error" sound is played when you miss a square
- Statistics (#hit/#total, #missed, accuracy%, #combo) shown on the side, and time remaining shown on the top
- Change the color/skin of the cursor for personalization

Libraries Needed:
- processing.sound.*: to play and manipulate songs (increase/decrease speed), also to play "click" and "error" sounds
- ControlP5: all the UI, buttons, song browser, and settings

How does it work?
Select a song from the song browser. You can then press confirm at the bottom if you would like to go through with your decision and play the song you selected. The objective of the game is to hit as many squares as possible while maintaining high accuracy and a high combo to get a higher score. You move your cursor onto a square to hit it. You just need your mouse to be hovering over the square at the right time, no clicking required. This game requires good tracking skills with the mouse and the ability to read the notes quickly and ahead of time.

UML Diagram

SoundManager

-clickSound : SoundFile
-errorSound : SoundFile
-musicTrack : SoundFile

+playClick() : void
+playError() : void
+getTime() : float
+pause() : void

1   1

SoundSquaresGame

-notes : ArrayList<Note>
-soundFile : SoundFile
-stats : StatsManager
-speedMultiplier : double
-selectedSkin : SkinOption
-selectedCursor : CursorOption

+setup() : void
+draw() : void
+loadMap(filePath : String) : void
+checkHit(cursorX : int, cursorY : int) : void
+displayStats() : void

1   0..*

Note

-x : int
-y : int
-timeStamp : double
-hit : boolean

+display() : void
+update(currentTime : double) : void
+checkCursor(cursorX : int, cursorY : int) : boolean

StatsManager

-totalNotes : int
-hits : int
-misses : int
-accuracy : float
-combo : int

+addHit() : void
+addMiss() : void

1   1