**Major Programming Project Package**

**Table of Contents**

**Scratches**

* Scratch 1 *(Scratch/Scratch 1/Basic\_FFT\_Band/Basic\_FFT\_Band.pde)*
  + This scratch draws a linear FFT Band on screen that reacts to the frequencies pulled by the player
* Scratch 2 *(Scratch/Scratch 2/Scratch\_1\_3\_BeatDetect/Scratch\_1\_3\_BeatDetect.pde)*
  + This scratch tests out the beat detect feature by drawing a circle on screen and exponentially increasing the size upon beat detection
* Scratch 3 *(Scratch/Scratch\_3/Scratch\_3.pde)*
  + This scratch demonstrates the use of classes by creating a function that draws a circle and upon keyPress exponentially increases the size of the circle. This is all done in a separate class and later called within main.

**Releases**

* Alpha 1.2 *(Releases/Alpha1\_2/Alpha1\_2.pde)*
* Alpha 1.25 *(Releases/Alpha1\_25/Alpha1\_2.pde)*
* Release 2.0 *(Releases/Release\_2\_0\_Circular\_Waveform/Release\_2\_0\_Circular\_Waveform.pde)*
* Beta 1.2 *(Releases/Beta\_1\_2/Beta\_1\_2.pde)*
* Beta 1.3 *(Releases/BETA\_1\_3(STABLE)/Beta\_1\_3/Beta\_1\_3.pde)*
* Beta 1.35 *(Releases/Beta\_1\_35/Beta\_1\_35.pde)*

**Documents**

*(Documents)*

* Initial Spec
* WIP 1
* WIP 2
* WIP 3
* Release Schedule

User Manual

**Progression of Specifications**

* Progression throughout the Alpha stage of the release schedule saw little to no changes
* The Beta stage saw heavy revisions as feedback resulted in many new goals
* This saw the change of Beta 1.2 to involve the randomization of colors, alphas and position as well as the addition of an entirely new release, Beta 1.3 that included the implementation of object classes
* Most of the stages leading up to Beta 1.3 progressed relatively smooth, however arraylists causes a significant bump in the road
* As of right now the program is %80-90 complete. The only additions remaining would be a working arraylist to control the decay of randomly spawned blooms as well as a UI Menu for user upload of music easily
* Possible soundcloud integration could also be a goal

**Program Overview**

* The program is designed to create aesthetically pleasing visualizations for music
* It utilizes the minim library for processing in order to implement various functions unique to the subject matter
* Firstly, it implements the Audio Player class (in minim by default) to easily play a certain audio file
* Then it uses the playerBuffer to display lines along
* It also implements the beatdetect function to display a circle and create cool blooming effects
* The user can then switch between songs and even upload their own music
* The program will also pull and display the meta-data of the track being played
* The project is somewhat of an art piece as opposed to a game etc

**List of Known Bugs**

* Implementing the blooms into an object causes some weird colouration within other objects
* The inside of the main circular audio band seems to take on the fill of the random blooms that spawn
* Beta 1.35 contains a failed attempt at implementing an array list for the random spawning of blooms

**User Manual**

* The user can use the “p” button to pause playback of the track
* The user can also use the “a” and “d” keys to switch between tracks inside the program folder
* The user can also copy and paste their own tracks inside the main program folder (found at *Documents\Processing\Beta\_1\_3).* The program will only recognize tracks named Music(*insert number here)*, it will also play in that order upon running the program. Without modification to the code, the program will play a maximum of 4 tracks. However this can be easily modified.

**Notes to Future Programmers**

* Given the nature of the program it can obviously be improved infinitely by simply adding more elements that react to the audio differently
* It is important to note that having a screen that is too busy is also a detriment to the overall design of the project
* This project hinges heavily on a minimalistic design philosophy although future programmers can modify the project into whatever the prefer as aesthetic is purely subjective