For milestone 3 the features I decided to implement were a splash loading screen and some background music. These are generally cosmetic features that don't improve the gameplay itself but do instead make it look more professional and aesthetically pleasing to the player.

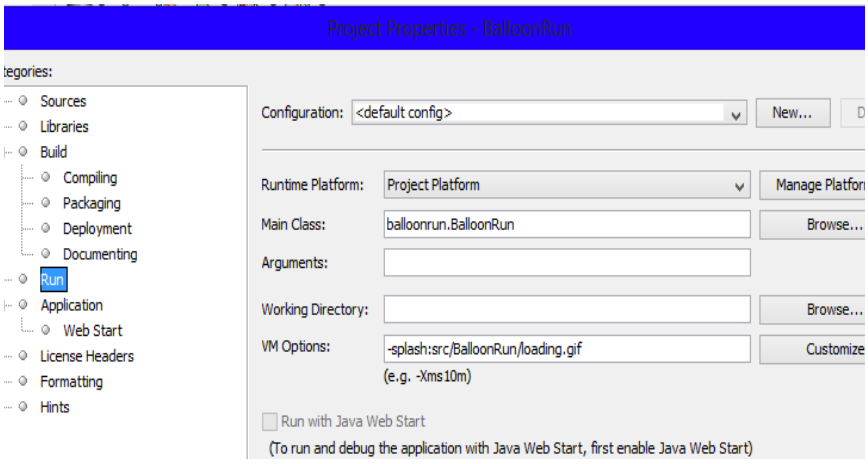
Splash screen

http://wiki.netbeans.org/Splash_Screen_Beginner_Tutorial – Tutorial followed

To help find out how to implement a splash screen I used the Netbeans wiki website which showed a tutorial of how to add one into a project. There are a number of different ways of doing it but I felt this was the best and easiest way because Netbeans provides the means within the software to add a loading screen in very easily. First I had to create a GIF as I wanted my splash screen to be animated so I chose two pictures that I thought would work well together.

These are the two images that I chose to use in my GIF which alternate between the two.

After creating the gif I had to add it into my project so it could be accessed correctly and I then needed to go into the manifest.mf file and add in this line: splashscreen-image: BalloonRun/loading.gif. This points to my GIF file so Netbeans knows which image I want.



I then needed to go to my project properties and open up the run category and add this line into the VM options for debugging purposes: -splash:src/BalloonRun/loading.gif.

```
public static void main(String[] args) throws InterruptedException {
    for (int i =0; i<=4; i++) {
        Thread.sleep(1000);
    }
```

This is the final part of the splash screen which is a for loop to control how long the splash screen lasts for. Every time the for loop runs it adds 1 second onto the timer until I becomes greater than 4. In my program, the splash screen lasts for 5 seconds.

Music

https://www.youtube.com/watch?v=SCf2x2qGcdI – Tutorial followed

My second feature was to add background music into my game. The main coding issues for this was to make sure there was a throw and catch for an exception that may occur. It also involved using some components I wasn't familiar with including the sun audio library.

```
import java.io.*;
import sun.audio.*;
```

Above are the two import statements required for this feature. I hadn't used the sun audio before so it required some reading and tutorial guidance to understand how it worked.

```
    try {
      InputStream audio = new FileInputStream("music/music.wav");
      BM = new AudioStream(audio);
      AudioPlayer.player.start(BM);
      musicPlayer.start(loop);
    } catch (IOException error) {
      System.out.println(error);
    }
```

This is the try and catch code for the music which will try to run the music and if an IOException occurs, there will be a message posted to the console. TO make the music play I had to add it to the main class where the worlds are created and then use the music() method to play the music when the world is created.