OJVR - ‘ _BaseProject’

StartScene.unity - The Base Scene

Player - the player object, provided by the [SteamVR plugin]

Basic Objects - Unity default lighting and plane, with teleport area script (steamVR) allows the area to be teleported to.

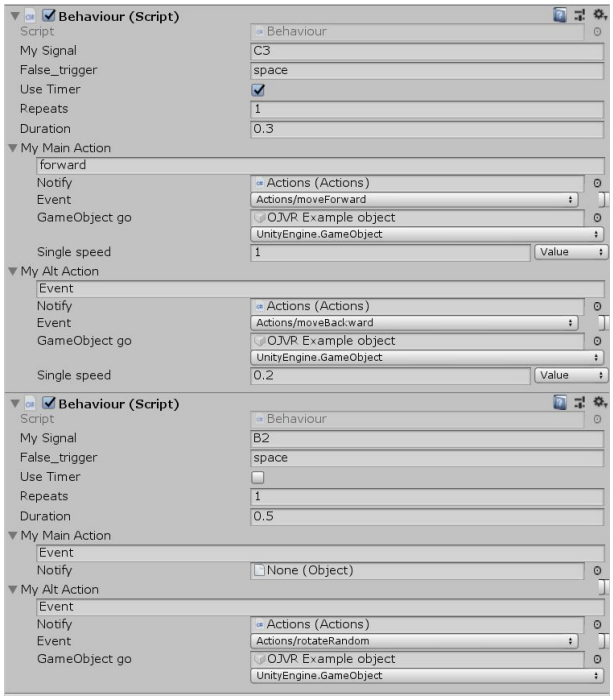
OJVR/TimingsManager - Script that controls the firing of *signals* (signals represent a midi note, and will be fired at the appropriate time as defined in the midi file). Assign the MidiTool output file here.

OJVR/Actions - Contains Actions.cs, the script defining all possible actions for OJVR objects.

OJVR Example Object - A cube object, that moves backwards a bit every time C3 signal is received, and rotates in a random direction every time B2 signal is received. (see bottom right screenshot)

OJVR Assets/Actions/Behaviour.cs

Defines the on signal received behaviour of an object.



My Signal - the signal that this object responds to

False_trigger - Used for testing, defines a key that the script will also respond to, simulating receiving the appropriate signal.

Use Timer - Should the alternate action run every frame for the duration of a timer? Or:

Repeats - should the alternate action run a limited amount of times

Duration - length (in seconds) of the timer.

My Main Action - the action that the object performs when no signal is received. Eg. Actions/moveForward.

My Main Action/GameObject go - the object the action is performed to (in this case, itself)

My Alt Action - the action that runs when a signal is received (eg. moveBackward). Will run every fame for x frames (*Repeats*) or for y seconds (*Duration*)

OJVR Assets/Actions/Actions.cs

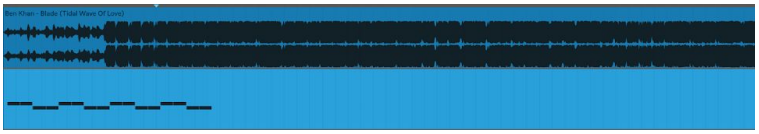
Defines the actions OJVR objects can perform, add new actions here.

```
//simple movement
public void moveForward(GameObject go, float speed){
    go.transform.Translate(Vector3.forward * Time.deltaTime * speed);
}

public void moveBackward(GameObject go,float speed){
    go.transform.Translate(Vector3.forward * Time.deltaTime * speed * -1);
}

//rotation
public void rotateRandom(GameObject go){
  go.transform.rotation = Random.rotation;
}
```

Workflow

Starting Your Project:

1. Create a copy of _BaseProject / rename, this will be your project folder.
2. Open StartScene.unity and delete the example objects (optional)
3. Create a MIDI file with appropriate timings for your song. This can be updated whenever.



4. Export midi file (suggestion: export it to **MidiTool** folder (OJVR_Assets/MidiTool, named test.mid)
5. In MidiTool/config, change settings for output file location (or leave default), file input (the above midi file, test.mid by default) and BPM (this is very important)
6. Run miditool with run.bat, this will create “output.json” in the output file location.
7. In your unity project: click OJVR/TimingsManager and set the path of output.json on the script object in the inspector
8. Set the AudioSource’s audio clip to your song mp3
9. If changes are made to the .midi file, run miditool again.

Creating an object that responds to signals.

1. Add behaviour.cs to the object
2. Configure behaviour.cs in the unity inspector. (Signal, use timer etc.)
3. Configure actions in the inspector (main/alt), set the game object parameter of each action to the game object it’s on.

Creating new actions.

1. Add a new function to Actions.CS