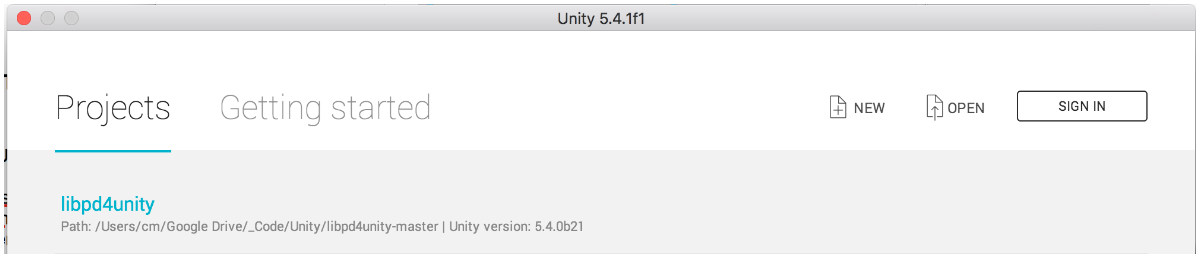
Tutorial - Week 6

This week we'll start looking into Unity 3D as well as the Pd + Unity integration project.

LOADING THE BASIC PROJECT

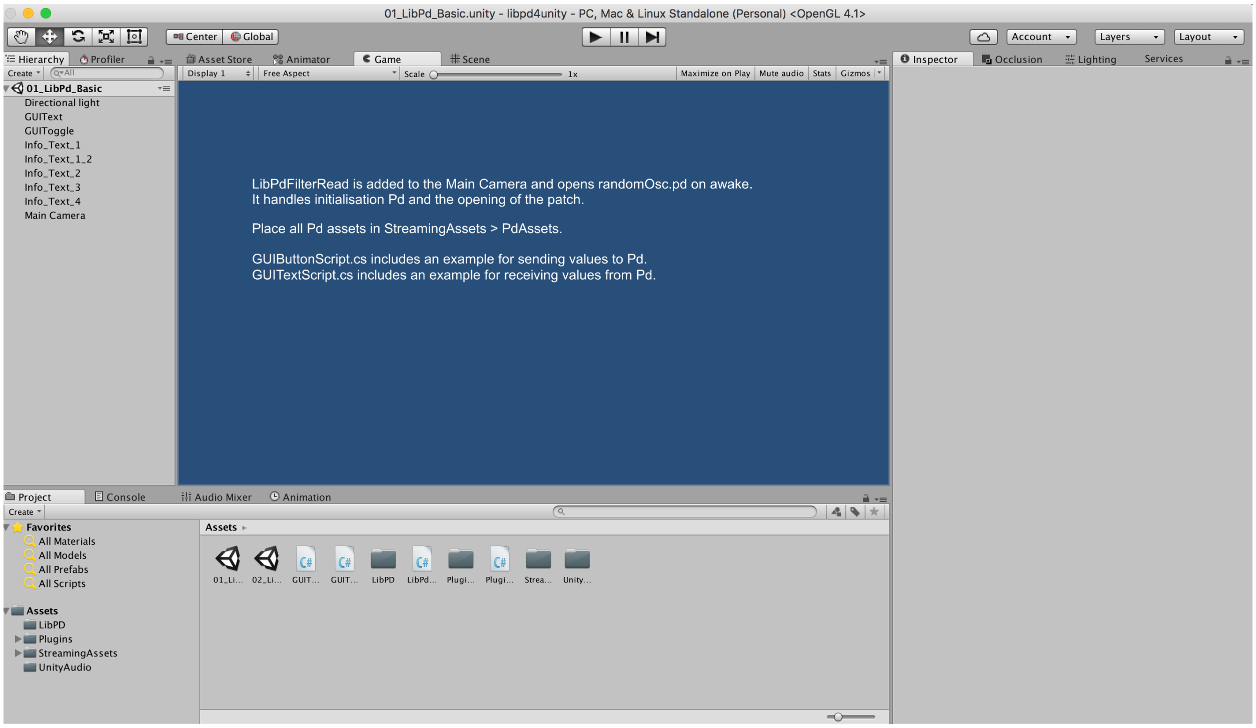
1) Download the "libpd4unity-master.zip" file from Blackboard. Unzip it and double-click the following file: libpd4unity>Assets>01_LibPd_Basic.unity. This will load the basic integration Pd+Unity scene, making sure that everything works properly. On the Unity window click on the project "libpd4unity":



libpd4unity

Path: /Users/cm/Google Drive/_Code/Unity/libpd4unity-master | Unity version: 5.4.0b21

2) If loaded properly you should be able to see this:



LibPdFilterRead is added to the Main Camera and opens randomOsc.pd on awake.
It handles initialisation Pd and the opening of the patch.

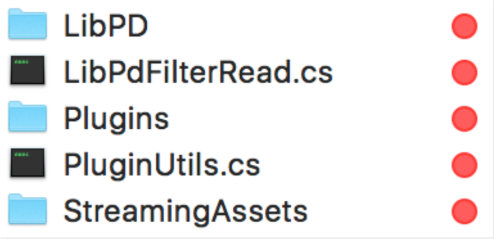
Place all Pd assets in StreamingAssets > PdAssets.

GUIButtonScript.cs includes an example for sending values to Pd.
GUITextScript.cs includes an example for receiving values from Pd.

3) Click the Play window at the center on the top. You should be able to hear a sine wave playing back random pitches. This means that everything works properly!

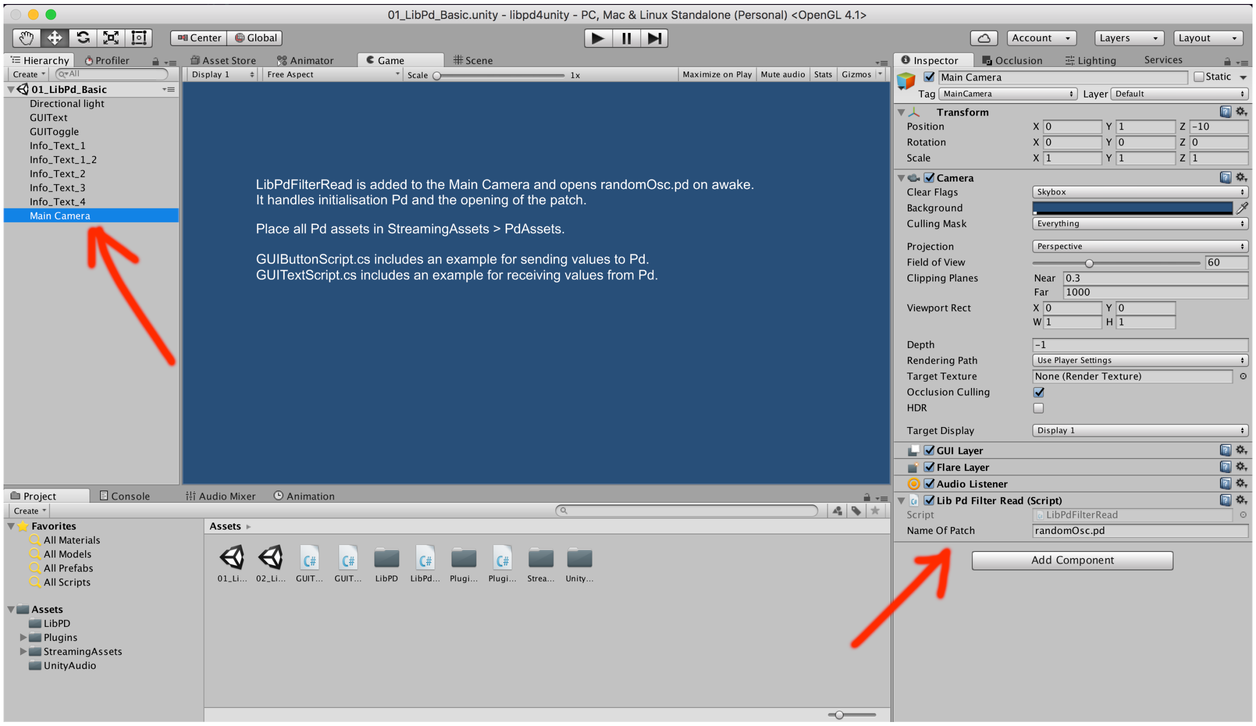
PROJECT'S ANATOMY

The integration of Pd within Unity is possible through the following folders, located in the basic project's Assets folder:



LibPD
LibPdFilterRead.cs
Plugins
PluginUtils.cs
StreamingAssets

Every time you create a new Pd + Unity project, make sure to copy these folders in the Assets folder of the newly created project. Back to the basic project, if you click on the "Main Camera" on the left, you'll notice that it contains a script Component named "Lib Pd Filter Read". It is there where you type in the name of the patch you want to be using, in this case "randomOsc.pd", which plays back those random pitches.



01_LibPd_Basic.unity - libpd4unity - PC, Mac & Linux Standalone (Personal) <OpenGL 4.1>

01_LibPd_Basic
Directional light
GUIText
GUIToggle
Info_Text_1
Info_Text_1_2
Info_Text_2
Info_Text_3
Info_Text_4
Main Camera

LibPdFilterRead is added to the Main Camera and opens randomOsc.pd on awake.
It handles initialisation Pd and the opening of the patch.

Place all Pd assets in StreamingAssets > PdAssets.

GUIButtonScript.cs includes an example for sending values to Pd.
GUITextScript.cs includes an example for receiving values from Pd.

Main Camera
Tag MainCamera Layer Default

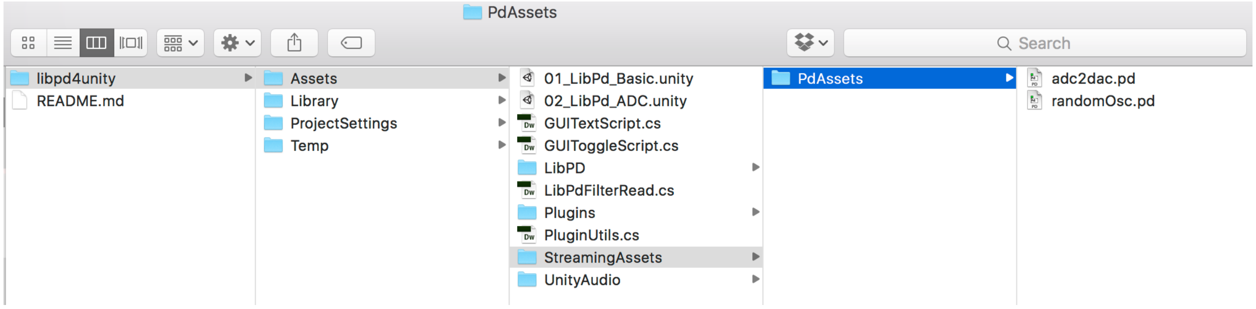
Transform
Position X 0 Y 1 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Camera
Clear Flags Skybox
Culling Mask Everything
Projection Perspective
Field of View 60
Clipping Planes Near 0.3 Far 1000
Viewport Rect X 0 Y 0 W 1 H 1
Depth -1
Rendering Path Use Player Settings
Target Texture None (Render Texture)
Occlusion Culling ☑
HDR □
Target Display Display 1

GUI Layer
Flare Layer
Audio Listener

Lib Pd Filter Read (Script)
Script LibPdFilterRead
Name Of Patch randomOsc.pd

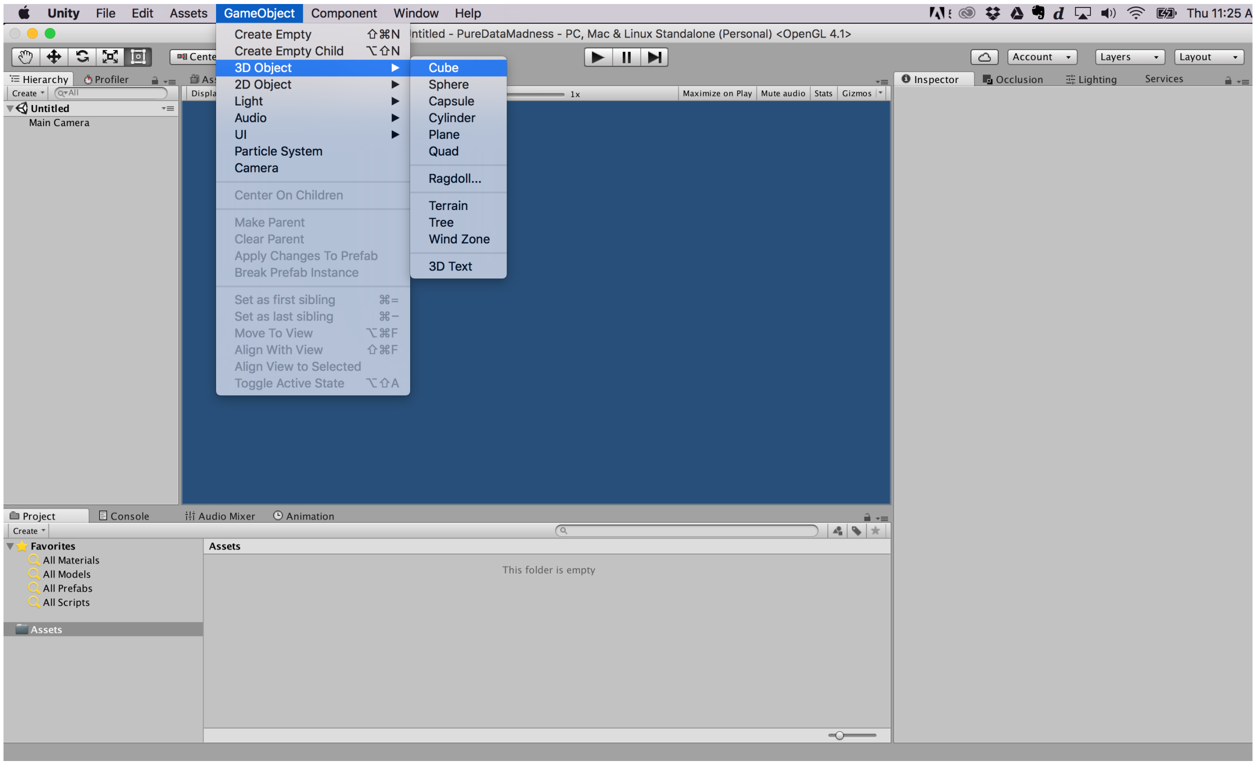
Assets
LibPD
Plugins
StreamingAssets
UnityAudio



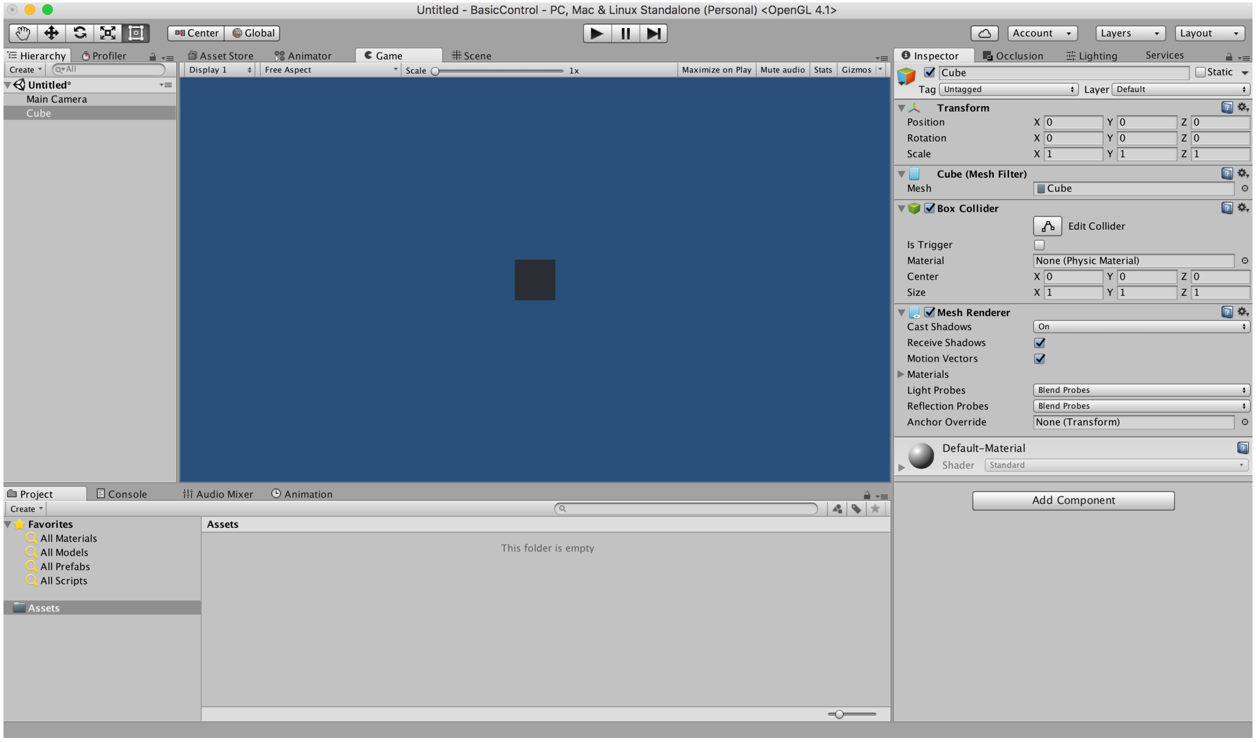
This will load any patch that is contained in the Streaming Assets folder; We'll be saving our Pd patches here. See if you can amend the "randomOsc.pd" patch and hear the changes when you run the game by pressing the Play button.

BASIC CONTROL

1) Create a new Unity project and copy all the necessary integration folders from the basic project, as described above.



From the top menu, go to 3D Object>Cube. This will insert a new Cube game object in our scene



Untitled - BasicControl - PC, Mac & Linux Standalone (Personal) <OpenGL 4.1>

Hierarchy: Untitled*, Main Camera, Cube

Inspector: Cube; Tag Untagged; Layer Default

Transform: Position X 0 Y 0 Z 0; Rotation X 0 Y 0 Z 0; Scale X 1 Y 1 Z 1

Cube (Mesh Filter): Mesh Cube

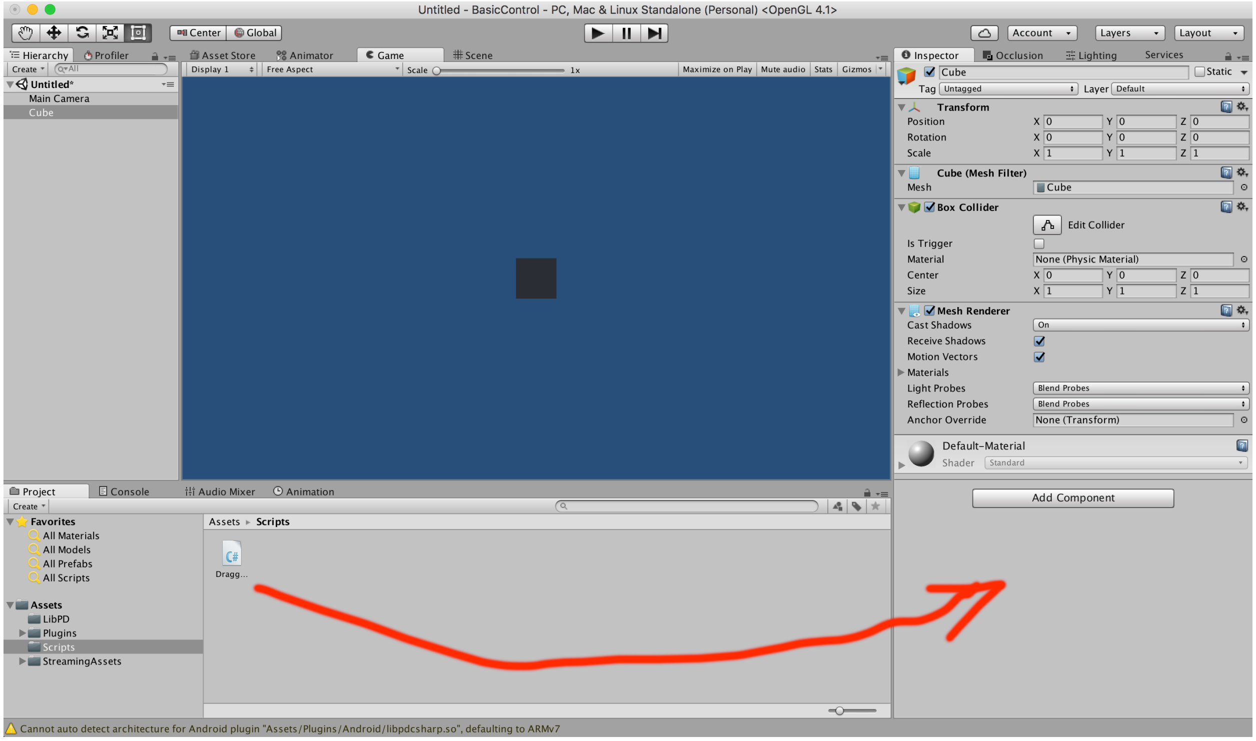
Box Collider: Is Trigger □; Material None (Physic Material); Center X 0 Y 0 Z 0; Size X 1 Y 1 Z 1

Mesh Renderer: Cast Shadows On; Receive Shadows ☑; Motion Vectors ☑; Light Probes Blend Probes; Reflection Probes Blend Probes; Anchor Override None (Transform)

Default-Material: Shader Standard

Assets: This folder is empty

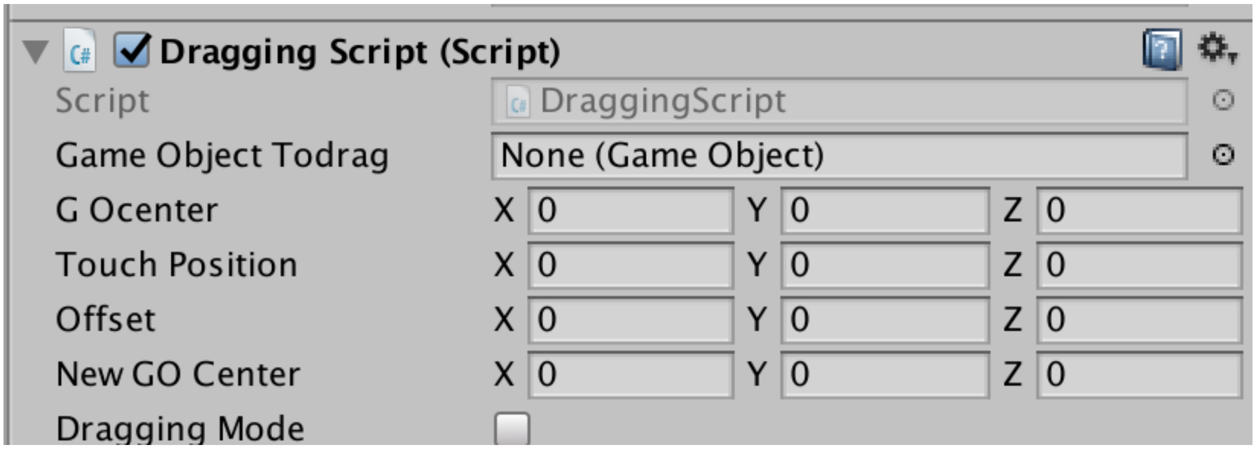
We will use this Cube to control the frequency of an oscillator in a Pd patch. Within your 'Assets' folder create a folder called 'Scripts' and paste in there the 'DraggingScript.cs' from Blackboard. This script should now be appearing in your Unity Assets>Scripts window too. Drag and drop this script on the Cube you created. A new component will be automatically created, containing this script.



Untitled - BasicControl - PC, Mac & Linux Standalone (Personal) <OpenGL 4.1>

Assets ▸ Scripts: Dragg...

Cannot auto detect architecture for Android plugin "Assets/Plugins/Android/libpdcsharp.so", defaulting to ARMv7



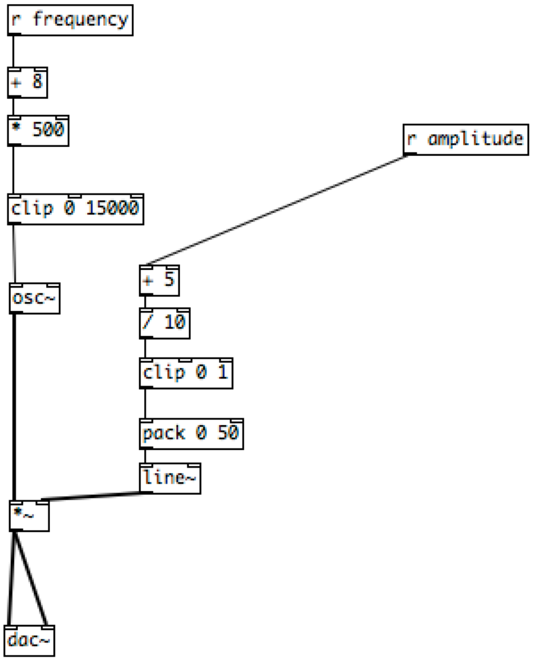
☑ Dragging Script (Script)

Script	DraggingScript		
Game Object Todrag	None (Game Object)		
G Ocenter	X 0	Y 0	Z 0
Touch Position	X 0	Y 0	Z 0
Offset	X 0	Y 0	Z 0
New GO Center	X 0	Y 0	Z 0
Dragging Mode	□		

Don't worry about the contents of the script yet, but now when you Run the game by pressing Play on top, **you should be able to click and drag this cube around using your mouse**. We'll use x to control the frequency and y the amplitude of an osc~ in Pd.

CREATE YOUR PD PATCH

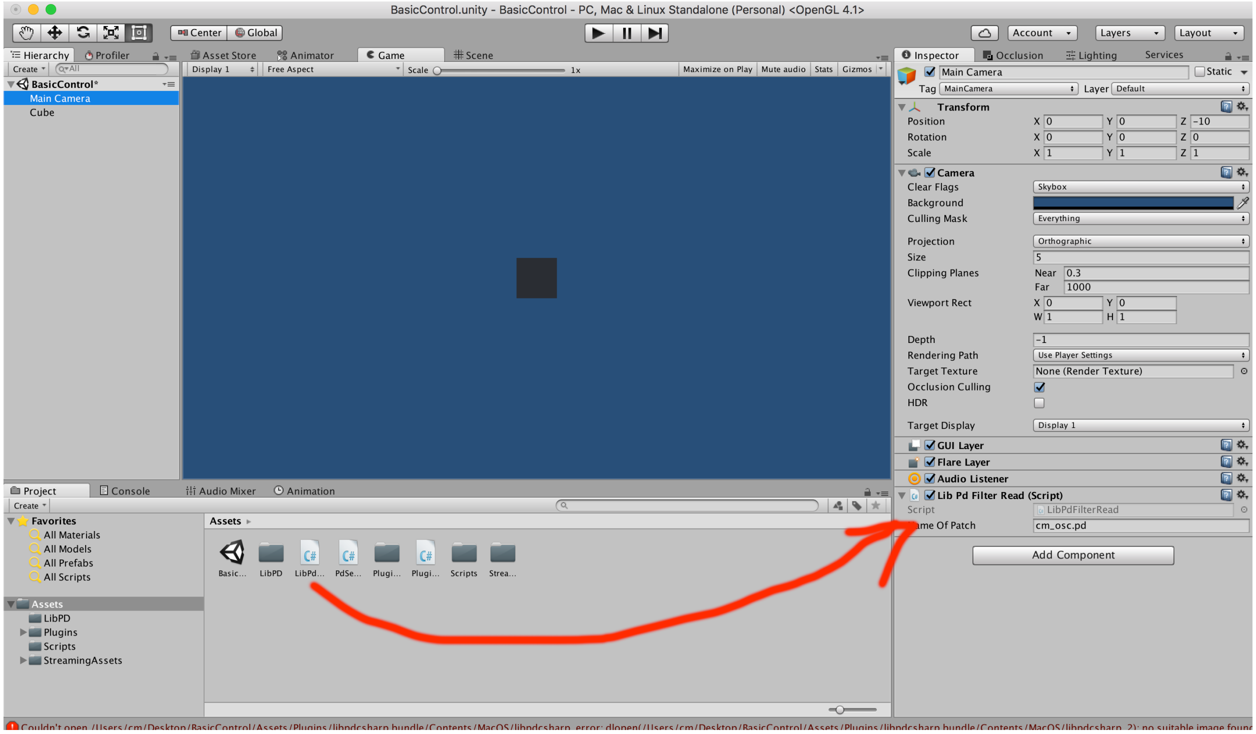
1) Create a Pd patch using an oscillator which receives "frequency" and "amplitude". We also need to do the appropriate scalings in order for the values from Unity to make sense in Pd:



Then, save the patch in the “Streaming Assets” folder and give it a name.

LibPdFilterread

1) Click, drag and drop the script LibPdFilterRead onto the main camera. Next to “Name of the Patch” type in the name of the patch you have created and saved in the “Streaming Assets” folder. (in my case it’s cm_osc)



Main Camera ☐ Static
Tag MainCamera Layer Default

Transform
Position X 0 Y 0 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

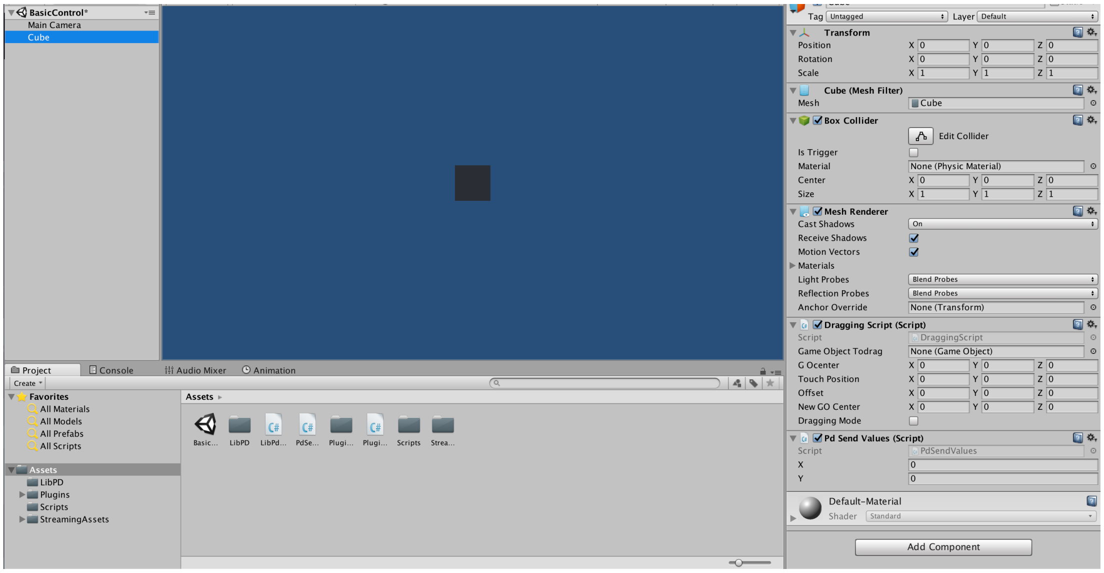
☑ Camera
Clear Flags: Skybox
Background
Culling Mask: Everything
Projection: Orthographic
Size: 5
Clipping Planes: Near 0.3 Far 1000
Viewport Rect: X 0 Y 0 W 1 H 1
Depth: -1
Rendering Path: Use Player Settings
Target Texture: None (Render Texture)
Occlusion Culling: ☑
HDR: ☐
Target Display: Display 1

☑ GUI Layer
☑ Flare Layer
☑ Audio Listener
☑ Lib Pd Filter Read (Script)
Script: LibPdFilterRead
ame Of Patch: cm_osc.pd

```
Couldn't open /Users/cm/Desktop/BasicControl/Assets/Plugins/libpdcsharp.bundle/Contents/MacOS/libpdcsharp, error: dlopen(/Users/cm/Desktop/BasicControl/Assets/Plugins/libpdcsharp.bundle/Contents/MacOS/libpdcsharp, 2): no suitable image found
```

SENDING VALUES TO PD

The last thing we need to do is click drag and drop the script “PdSendValues.cs” (downloaded from Blackboard) onto the Cube. The cube components should now look like that:



Tag: Untagged · Layer: Default

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

Cube (Mesh Filter)

Mesh: Cube

☑ Box Collider

Is Trigger: □
Material: None (Physic Material)

	X	Y	Z
Center	0	0	0
Size	1	1	1

☑ Mesh Renderer

Cast Shadows: On
Receive Shadows: ☑
Motion Vectors: ☑
Materials
Light Probes: Blend Probes
Reflection Probes: Blend Probes
Anchor Override: None (Transform)

☑ Dragging Script (Script)

Script: DraggingScript
Game Object Todrag: None (Game Object)

	X	Y	Z
G Ocenter	0	0	0
Touch Position	0	0	0
Offset	0	0	0
New GO Center	0	0	0

Dragging Mode: □

☑ Pd Send Values (Script)

Script: PdSendValues
X: 0
Y: 0

Default-Material
Shader: Standard

Finally, when you Run your game now (by clicking the Play button), you should be able to control the sine wave through the Unity interface!

Have a look at the “PdSendValues.cs” script, read the commented code and see if you can understand how we manage to send the values to Pd. We’ll examine this in detail in the next lecture.