Anarchy for the USB

Rapid 3D game prototyping from idea to Device in less than an hour using Project Anarchy

Contents

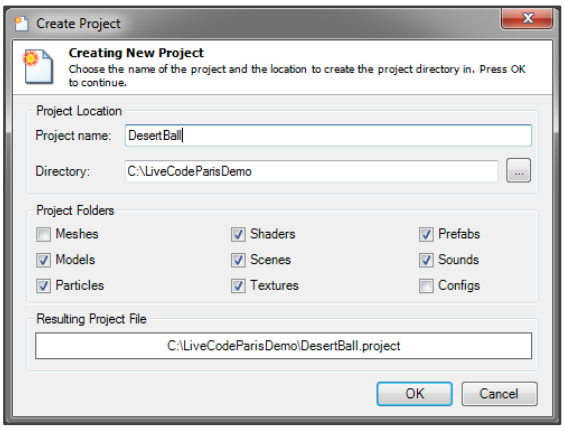
Pre-requisites

- A host PC with the Project Anarchy SDK installed.
- Android SDK and tools
- A target Android device with connecting USB cable
- The necessary device drivers to connect the device via adb.
- The supplied vSceneViewer.apk installed on the Android device
- The asset bundle LiveCodeparisDemo.zip file
- A Wi-Fi connection for both host PC and Android device†.
    - OR
- USB tunnelling software setup and installed. (see appendix A) †
- A game controller connected to host PC

† One or the other of these options must be functional.

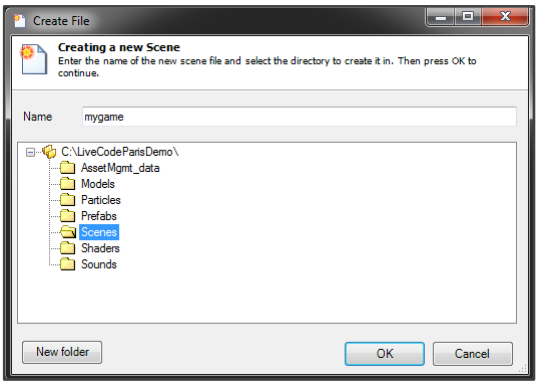
Creating the World

Launch vForge. Create new project in the demo location (**"LiveCodeParisDemo"**).



Unzip the assets file over the new project.

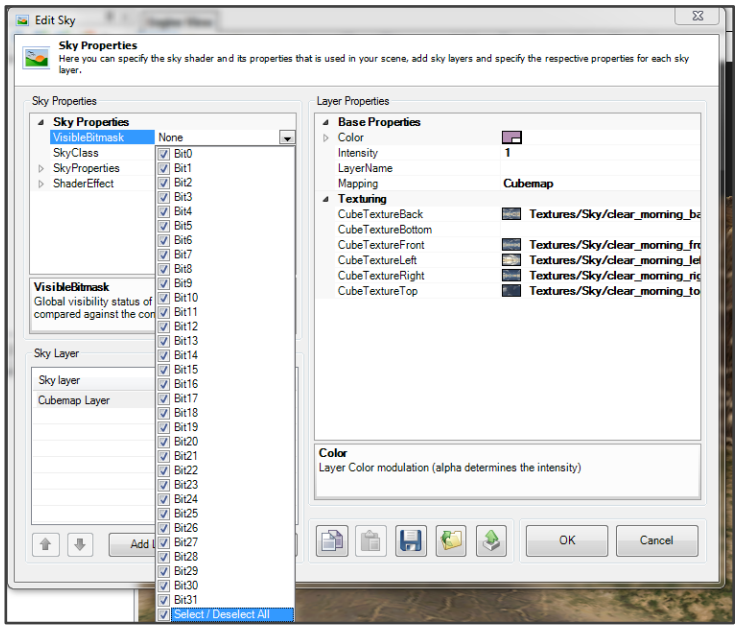
Create new scene called **"mygame"** in the scenes folder



Using the Scene Wizard, select **MobileForward Renderer**, skip the Post Processors and Lighting options and press OK.

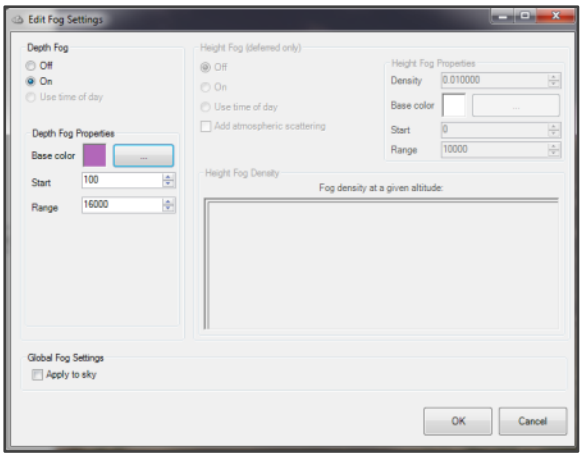
Ensure editor is in ***Default Layout*** and select the ***Shape Creators*** tab. Locate the **Static Mesh Instance** object and drag it into the centre of ***Engine View*** window. In the ***Properties*** tab select **MeshFileName** to open the file browser and select **"track11k.vmesh"**

From the pull down menu open *Engine->Edit Sky*. Ensure mapping is set to **Cubemap** and for each of the CubeTexture's in turn set to their corresponding 'clear_morning_xxxx.bmp'



In **Sky Properties** select the **Visible Bitmask** and ensure all bits are set. Finally open the colour tool in the **Color** option to adjust the overall colour tint [219,156,217].

Again from the pull down menu *Engine->Edit Fog* turn **Depth Fog** *On*, set **Start** to 1000 and **Range** to 16000 and if required adjust **Base color** [178,103,185].



Go to the ***Shape Creators*** tab and from the **Lighting** section drag a **Lights (static) – Directional Light** into the scene. Set the following ***Properties***:

Pitch	**0**
Multiplier	**0.8**
Light Color	**255,164,101**

Go to the ***Shape Creators*** tab and from the **Lighting** section drag a **Time-of-day sunlight** into the scene. Select **Back Light** in the pop-up dialogue. Set the following ***Properties***:

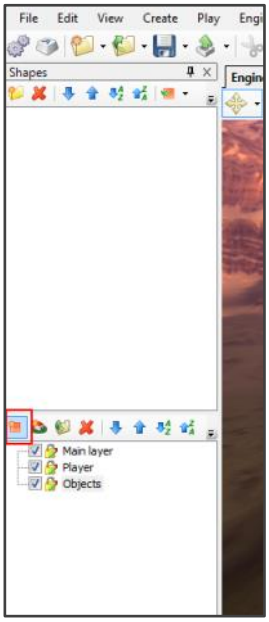
Sunlight

Pitch	**30**
Light Color	**183,172,109**

Back Light

Yaw	**-180**
Pitch	**-20**
Light Color	**173,150,102**

From the pull down menu select *Lighting->Calculate Lighting*. Ignore the warning message, this is normal first time. The scene should now be fully lit and shadow maps calculated and added.



Adding Objects.

Add 2 new layers to the scene; **Player** and **Objects**

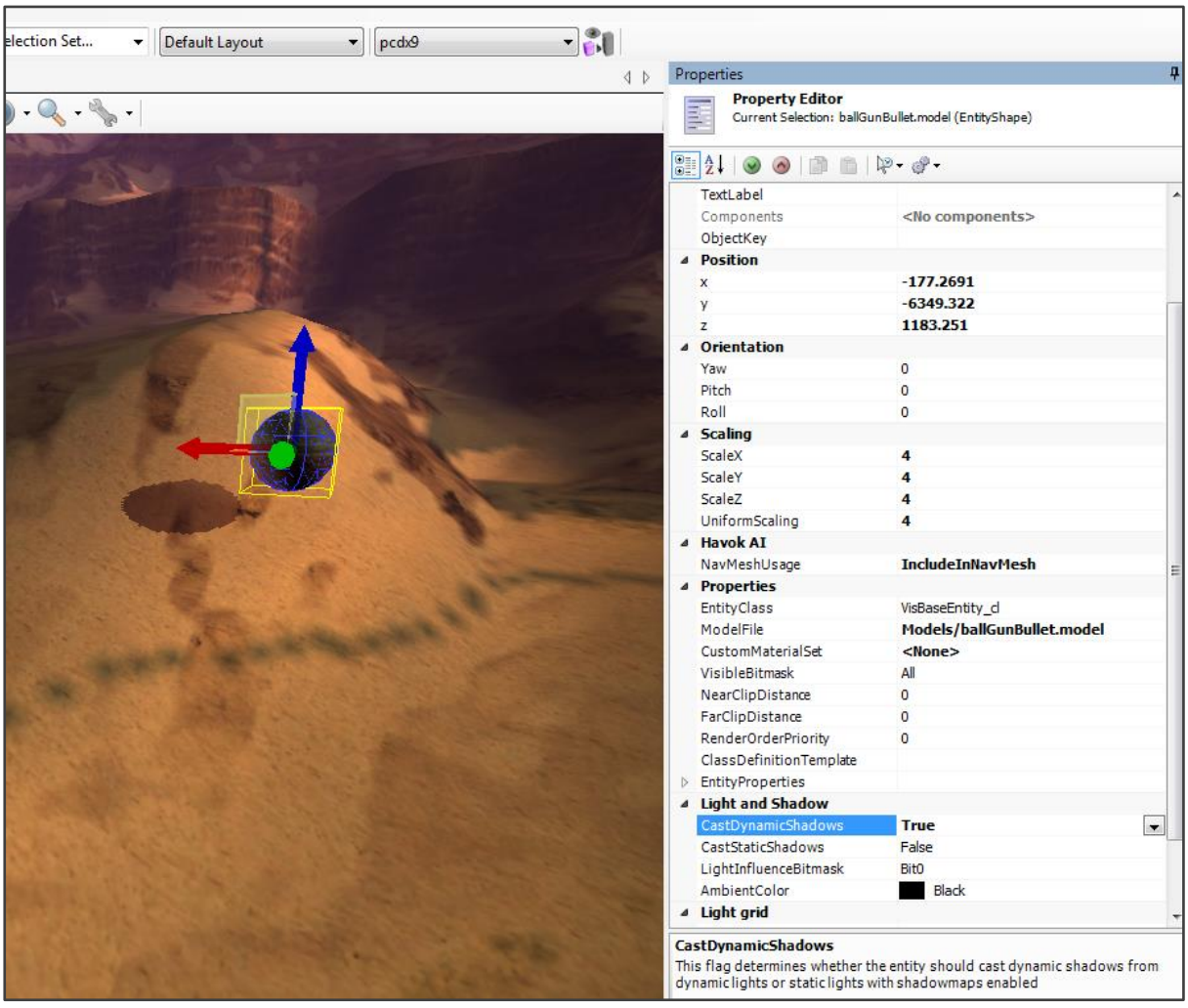
Add the Player

Select the **Player** layer

Open the ***Asset Browser*** tab and check the **Models** box. Locate the **ballGunBullet.model** and drag into the scene. (note: identify a suitable location with sloping ground). In the ***Properties*** tab set the following:

UniformScaling	**4**
CastDynamicShadows	**True**

Use the move tool to raise object a few meters above the ground.

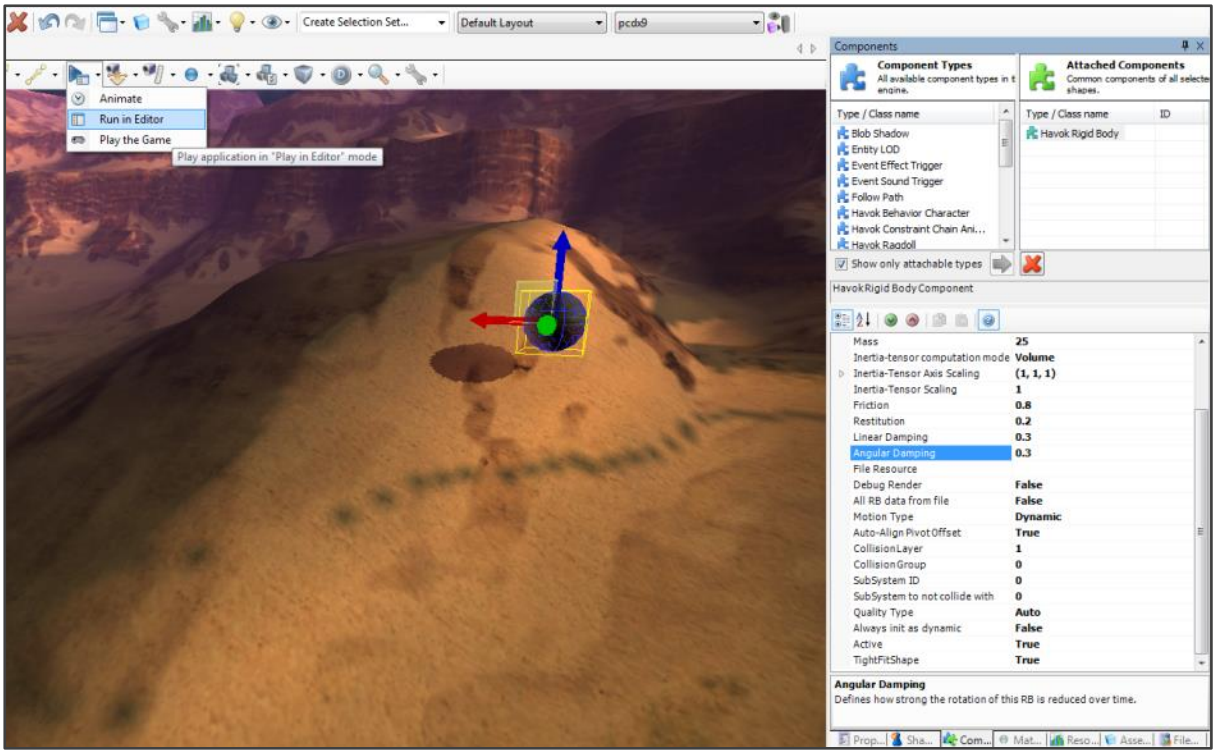


Add the Objects

Select the **Objects** layer

Open the ***Components*** tab, locate the **Havok Rigid Body** component and drag into the **Attached Components** list. Set the rigid body ***Properties***:

Mass	**25**
Friction	**0.8**
Restitution	**0.2**
LinearDamping	**0.3**
AngularDamping	**0.3**
TightFitShape	**True**



Select **Run in Editor** and watch the result. Stop **Run in Editor**

Return to the ***Asset Browser*** tab and locate and drag the **Barrel.model** into the scene, placing it at a suitable location downhill from the ball. Ground the barrel by raising it above the floor and using *RMB* on the move gizmo select *Drop to Floor -> Bottom center*. Select the ***Components*** tab and add a **Havok Rigid Body** setting the following rigid body ***Properties***:

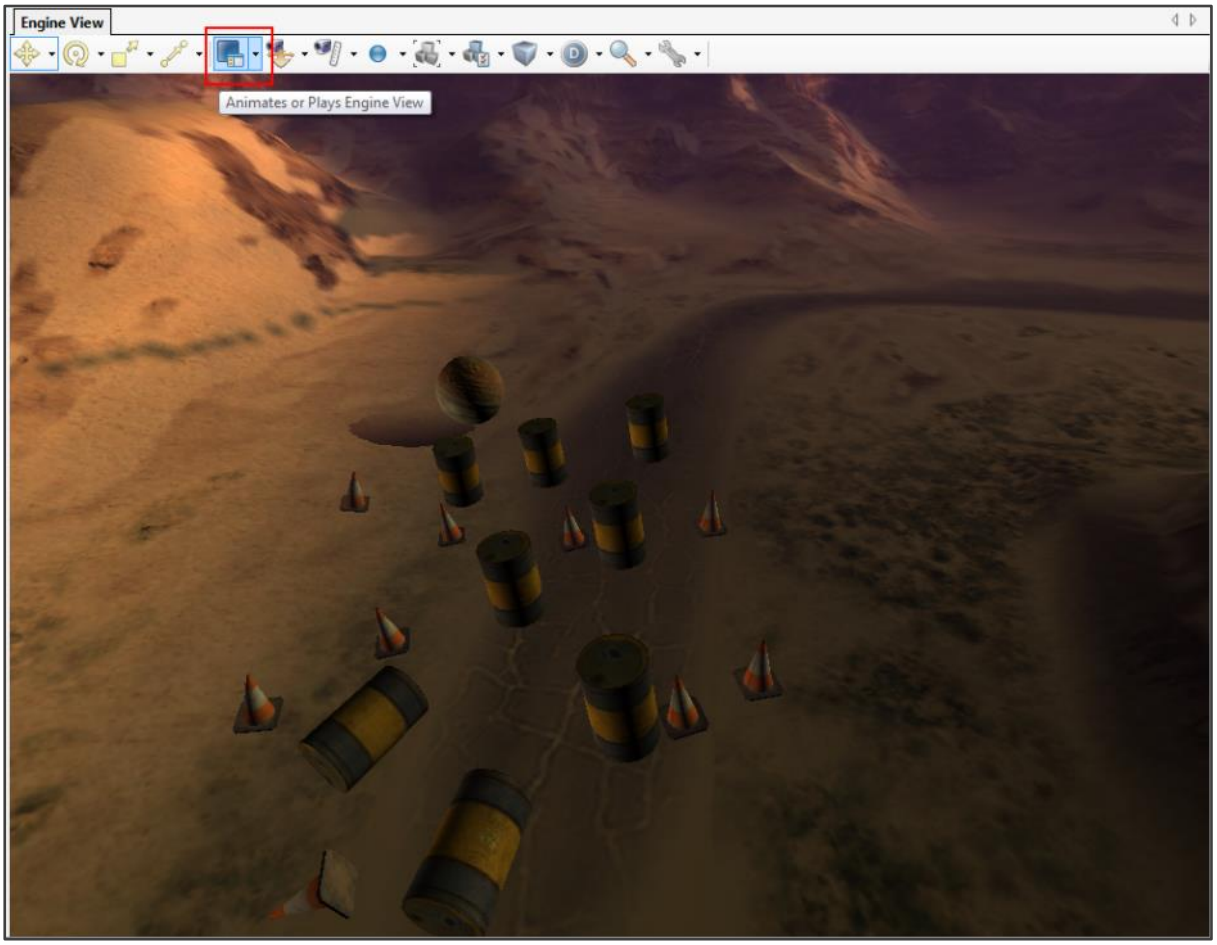
Mass	**4**
TightFitShape	**True**

Clone several copies of the barrel (hint: with an object selected, hold *Shift* and use the move gizmo to quickly clone and position). Ground barrels by selecting all and using *RMB* on the move gizmo select *Drop to Floor -> Bottom center*.

Repeat this process using the **TrafficCone.model** with the following ***Properties***:

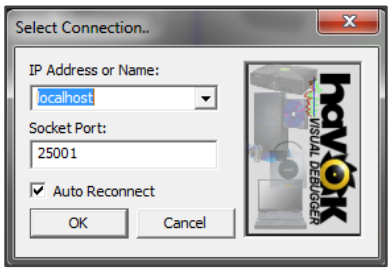
Mass	**2**
TightFitShape	**True**

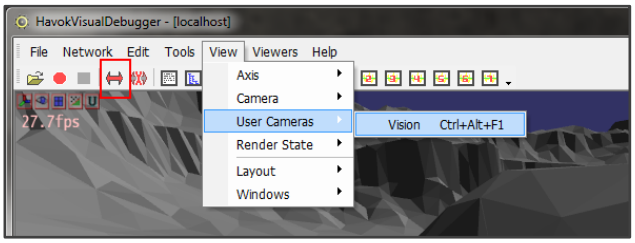
Select **Run in Editor** and watch the result. Stop **Run in Editor**



The Visual DeBugger (VDB)

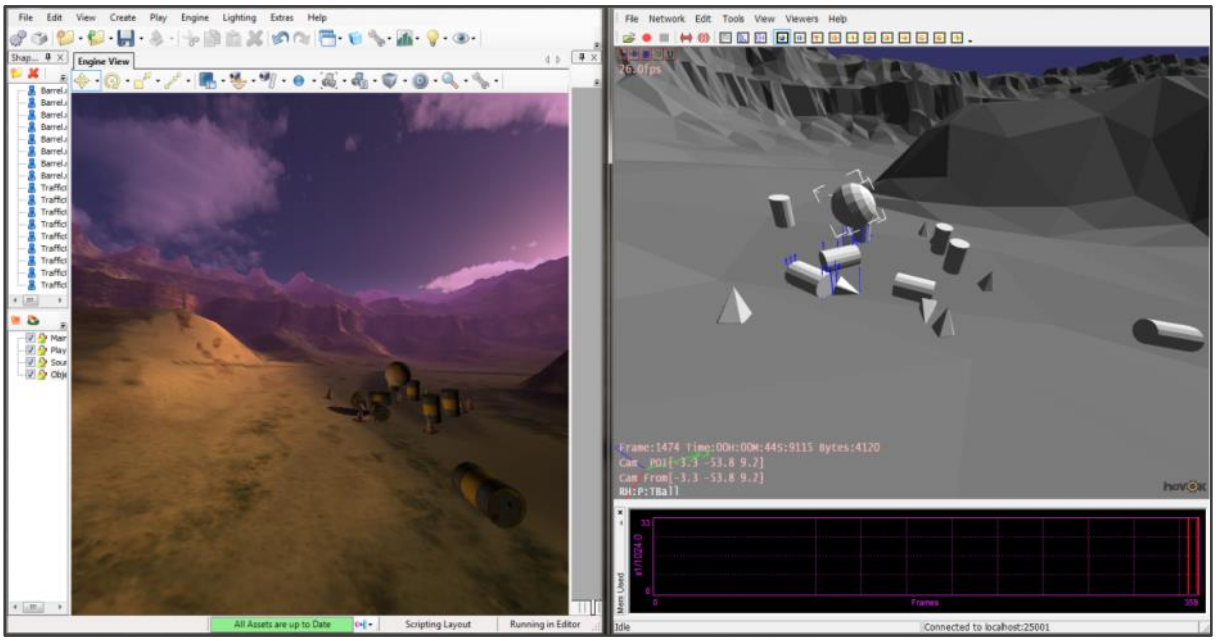
Optional demonstration: launch the **VDB** and arrange the windows such that both **VDB** and **vForge** are visible together. Connect the **VDB** to **vForge** via localhost and start





the simulation in vForge with ***Run in Editor***.

Manipulate the VDB camera view to best show the simulation. (Tip: menu item *View->User Cameras->Vision* will cause the **VDB** to use the **vForge** Camera). The Havok Physics properties can be examined and manipulated in the **VDB**. Pressing and holding *Space* whilst hovering the mouse pointer over an object allows that object to be 'grabbed' and dragged about the scene. Any manipulation of object in the **VDB** will immediately be reflected within **vForge**.



Stop the ***Run in Editor*** simulation. This time start a recording within the **VDB** (the red circle icon) and specify a suitable filename to save to. ***Run in Editor*** again and allow the simulation to run for a while. Stop the recording (the black square icon) and the ***Run in Editor*** simulation.

Re-load the recording and playback the simulation in the **VDB**, zoom in on objects, scrub the playback with the time slider and pause on frames for detailed examination.

Add Player Control

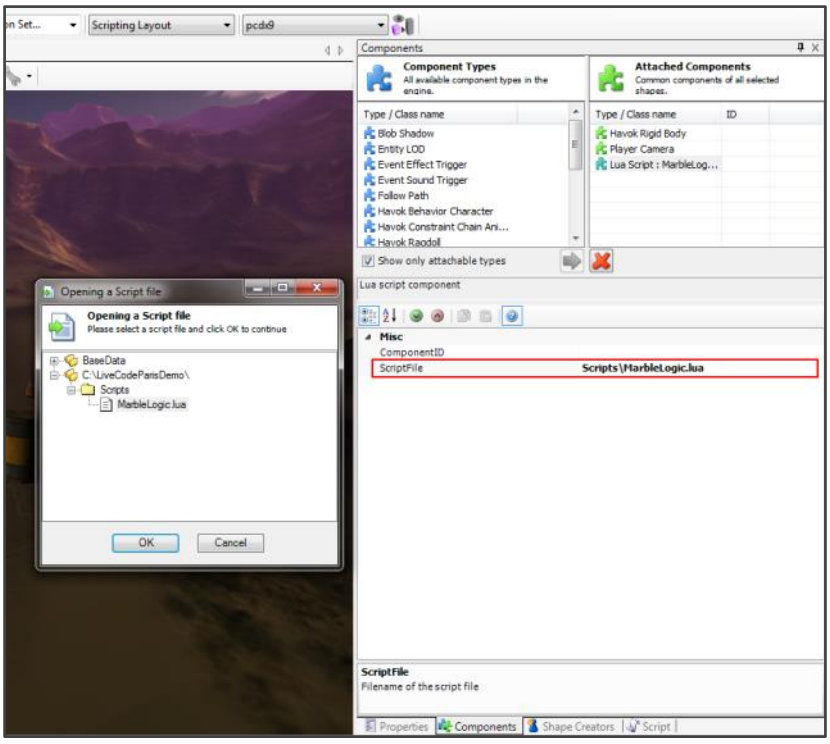
In the **Player** layer select the **ballGunBullet.model**.

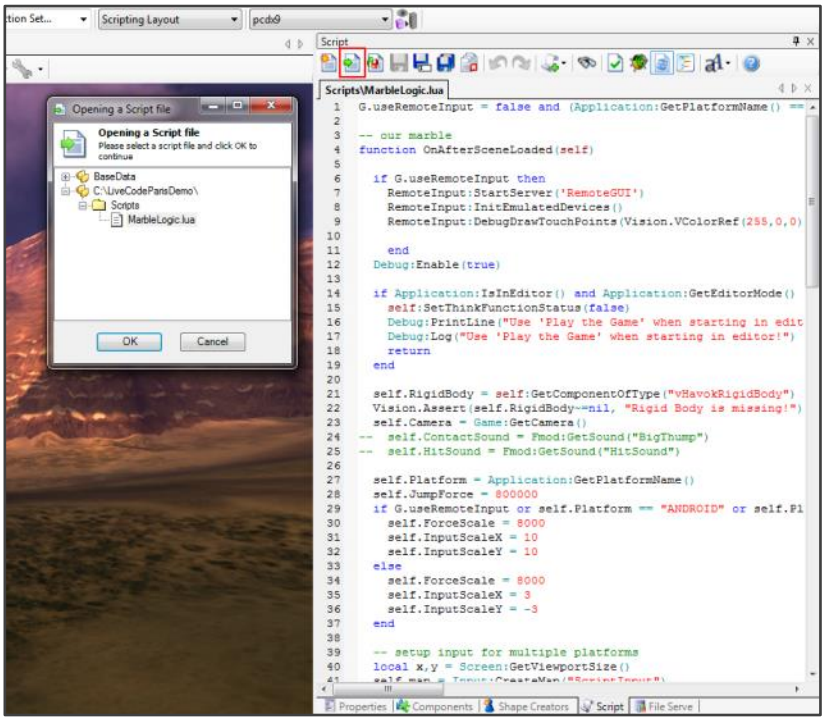
Open the ***Components*** tab and add a **Player Camera** component. Set the following ***Properties***

FollowFixed	**True**
CameraDistance	**1000**
MiniamlDistance	**1000**
MaximalDistance	**2000**

Add a **Lua Script** component and set **ScriptFile** to **"Sciptes\MarbleLogic.lua"** with the file browser.

In the drop-down change ***Default Layout*** to **Scripting Layout** and select the ***Script*** tab. The Lua script should be visible and editable. If not then click on the **Load existing Script** icon and select the **"MarbleLogic.lua"** file.





Select **Play the Game** mode and run. You will be able to control the ball with a suitable game controller. The camera should follow smoothly. Press *ESC* to exit **Play the Game** mode

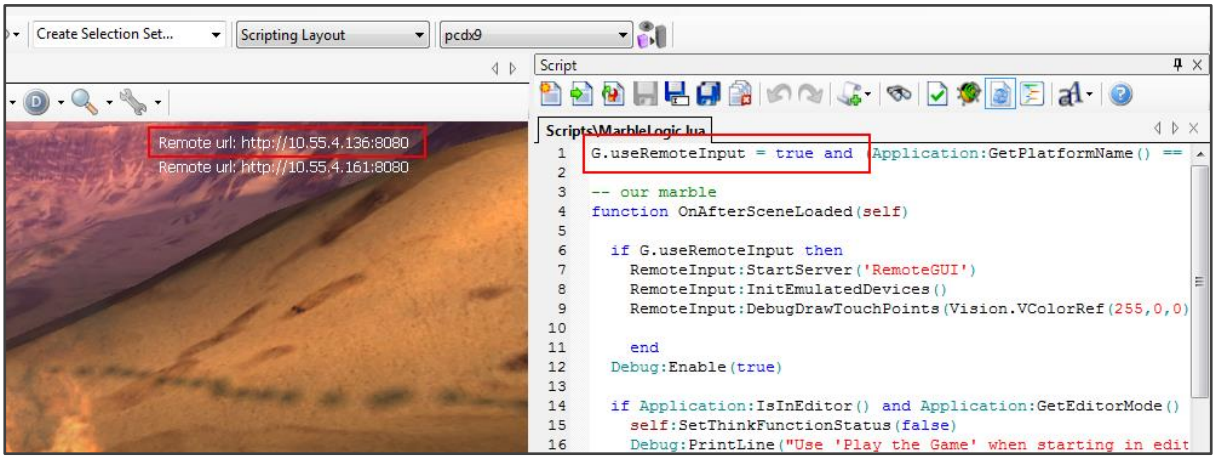
Using Remote Input

Note: This only works via a Wi-Fi connection on the same network as the host PC.

In the ***Script*** tab edit the Lua script line 1:

```
G.useRemoteInput = true and ...
```

Save the script and **Play the Game**. An IP address will be shown in the top right corner of the screen. On the device open an internet browser and navigate to the supplied IP address.



If all is well, touch and accelerometer inputs from the device will be sent to vForge and will control the Ball.

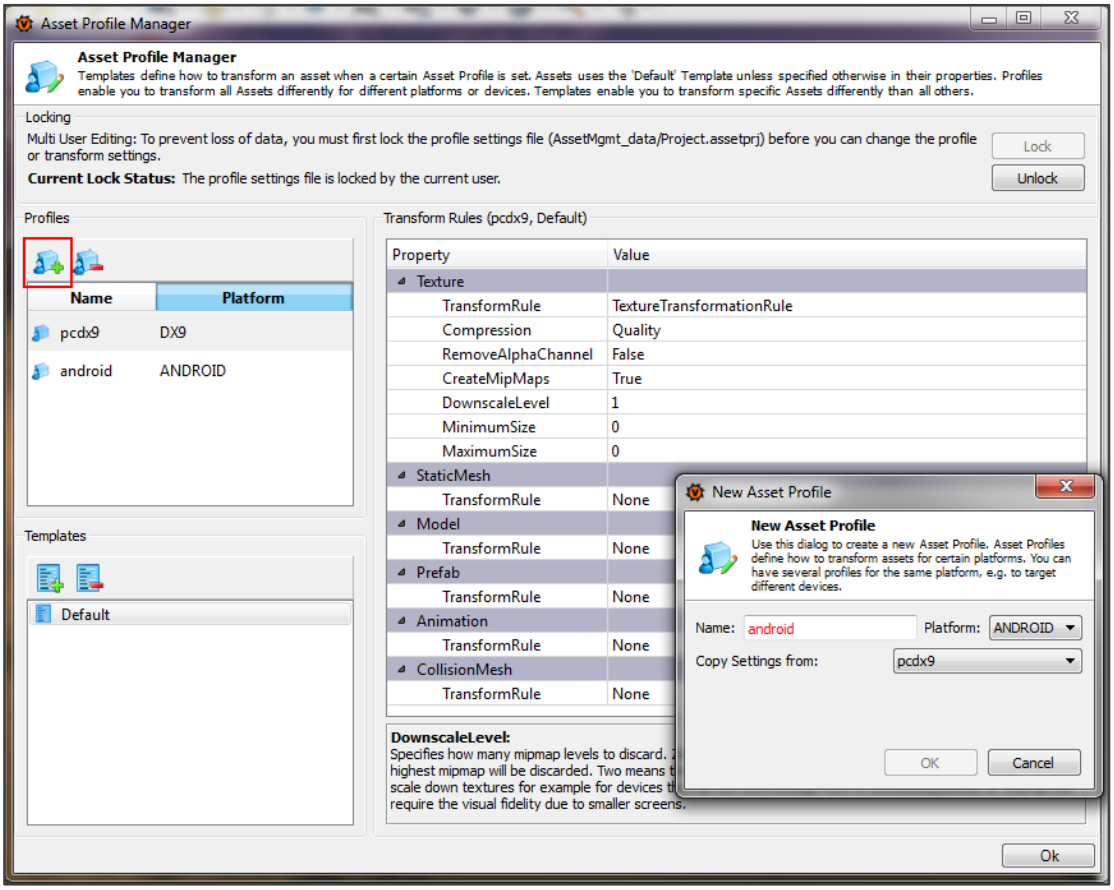
To disable the Remote Input, change the script back to:

```
G.useRemoteInput = false and ...
```

Preparing for Android

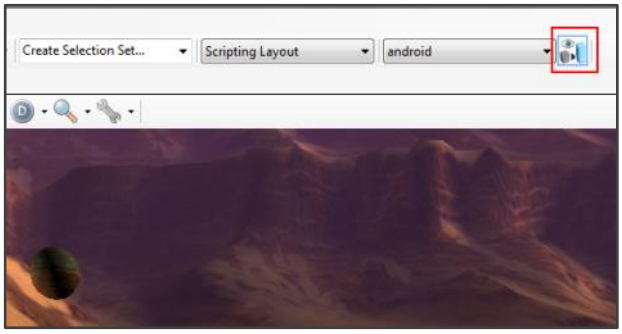
In the pull down menu select *Engine->Asset Profile Manager*. ***Lock*** the Profile to enable editing, and click the ***Add Profile*** icon.

In the ***New Asset Profile*** dialogue, set **Platform** to **ANDROID** and press *OK*. Set **Texture: DownscaleLevel** to 1 and press *OK*.



The progress of the automatic asset transformation can be monitored by the green progress bar at the bottom of the screen.

Select the ***android*** profile from the drop down selector. The icon to the right of the profile selection allows the scene to be viewed as the selected profile. Toggle this icon to view the differences.

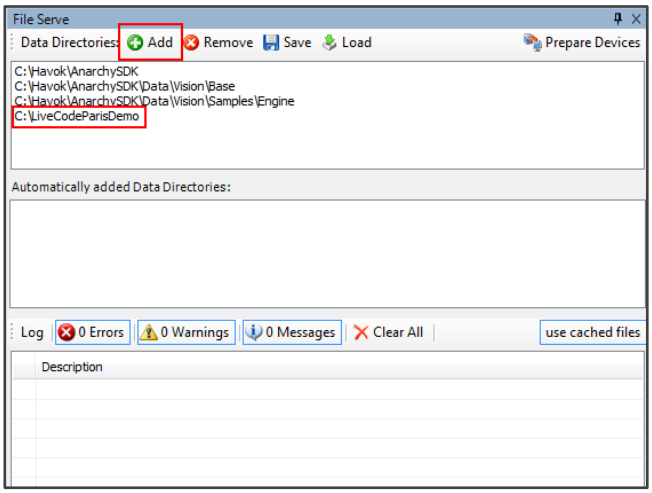


Once the asset transformation is complete, select export scene *File->Export-> Export Scene* and ensure the **Target File Name** is set to **"Scenes\mygame"**. Press ***Export Active Profile***. Check that inside the Scenes folder there exists **"mygame.android.vscene"** and **"mygame.android.vscene.data"**.

Deploy to Device

If the ***File Serve*** tab is not displayed, open it via *View->panels->File Serve* and attach it to pane.

Open the ***File Serve*** tab. In **Data Directories** click **Add** and add a path to your project folder.



Connect your Android Device and check it is correctly connected [adb devices], then click **Prepare Devices**. (If using file tunnelling skip this step or after clicking **Prepare Devices**, re-edit the **"vFileServeHost.txt"** file on the device as per appendix A)

On the device, launch the custom ***vSceneViewer*** app and select ***mygame***. Loading may take a few seconds longer on the first launch as no files are cached. The game should now be running and the Ball controlled by tilting the device.

Iterate and Run

It will probably be noted the Ball moves much slower than within vForge. Go to the ***Script*** tab and edit the Lua file lines 30-31 :

```
self.InputScaleX = 10
self.InputScaleY = 10
```

```
21    self.RigidBody = self:GetComponentOfType("vHavokRigidBody")
22    Vision.Assert(self.RigidBody~=nil, "Rigid Body is missing!")
23    self.Camera = Game:GetCamera()
24  --  self.ContactSound = Fmod:GetSound("BigThump")
25  --  self.HitSound = Fmod:GetSound("HitSound")
26
27    self.Platform = Application:GetPlatformName()
28    self.JumpForce = 800000
29    if G.useRemoteInput or self.Platform == "ANDROID" or self.Platform == "IOS" then
30      self.ForceScale = 8000
31      self.InputScaleX = 10
32      self.InputScaleY = 10
33    else
34      self.ForceScale = 8000
35      self.InputScaleX = 3
36      self.InputScaleY = -3
37    end
38
```

Save the file and re-load onto device by pressing back '↩' then the ***mygame*** icon again. The application will reload from cache, loading only the modified Lua script via the file server and should now be running with a much more responsive Ball.

Pimping the Scene

Background Music

Create a new Layer called **Sound**. From the ***Shape Creators*** tab drag a Sound object into the scene and set its properties:

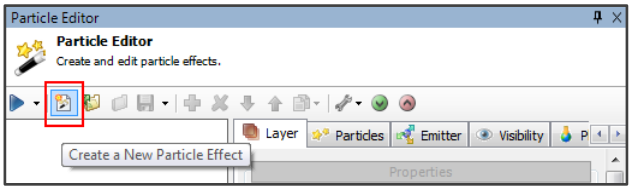
FileName	**Sounds\HeavyIndustrial.ogg**
Is3D	**False**
Background	**True**

Select **Play the Game** and check the music is audible.

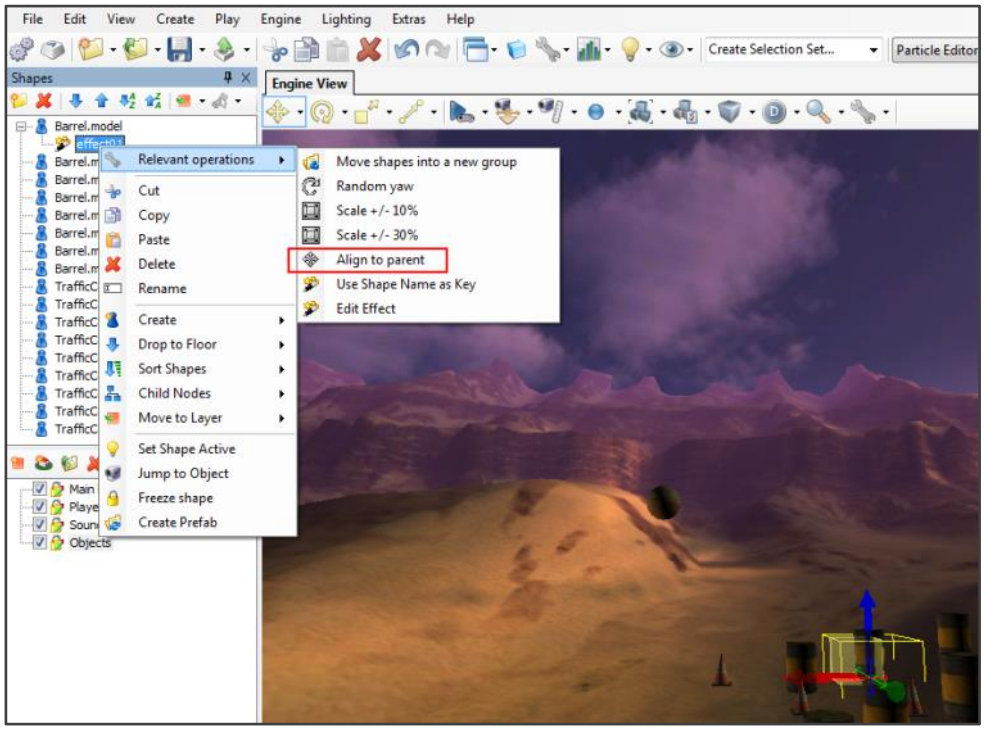
Re-export the scene and reload onto device ('**ↄ' + *mygame***). The music should be playing.

Flaming Barrels

From the *Layout* pulldown select *Particle Editor Layout* and open the ***Particle Editor*** tab. Click the ***Create a New particle Effect*** icon and from the **Template Selection** select **Fire Torch With Smoke**.



Open the **Objects** Layer and drag the newly created particle effect into the scene. Inside the ***Shapes*** window, drag the newly added effect object onto one of the barrel entities such as to make it a child of the Barrel. *RMB* on the effect object and select *Relevant operations->align to parent*.

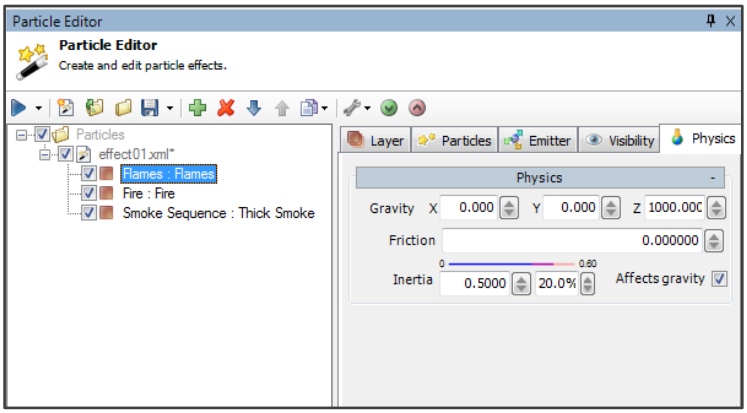


Using the move gizmo's blue arrow, drag the effect to the top of the Barrel. In the ***Properties*** tab set the following:

Pitch	**-90**
UniformScaling	**4**

Then back in the ***Particle Editor*** tab for each of the three subsystems set the **Z** component in the ***Physics*** tab and ensure **Affects gravity** is checked:

Flames : Flames	**Z**	**1000**
Fire : Fire	**Z**	**1000**
Smoke Sequence : Thick Smoke	**Z**	**300**

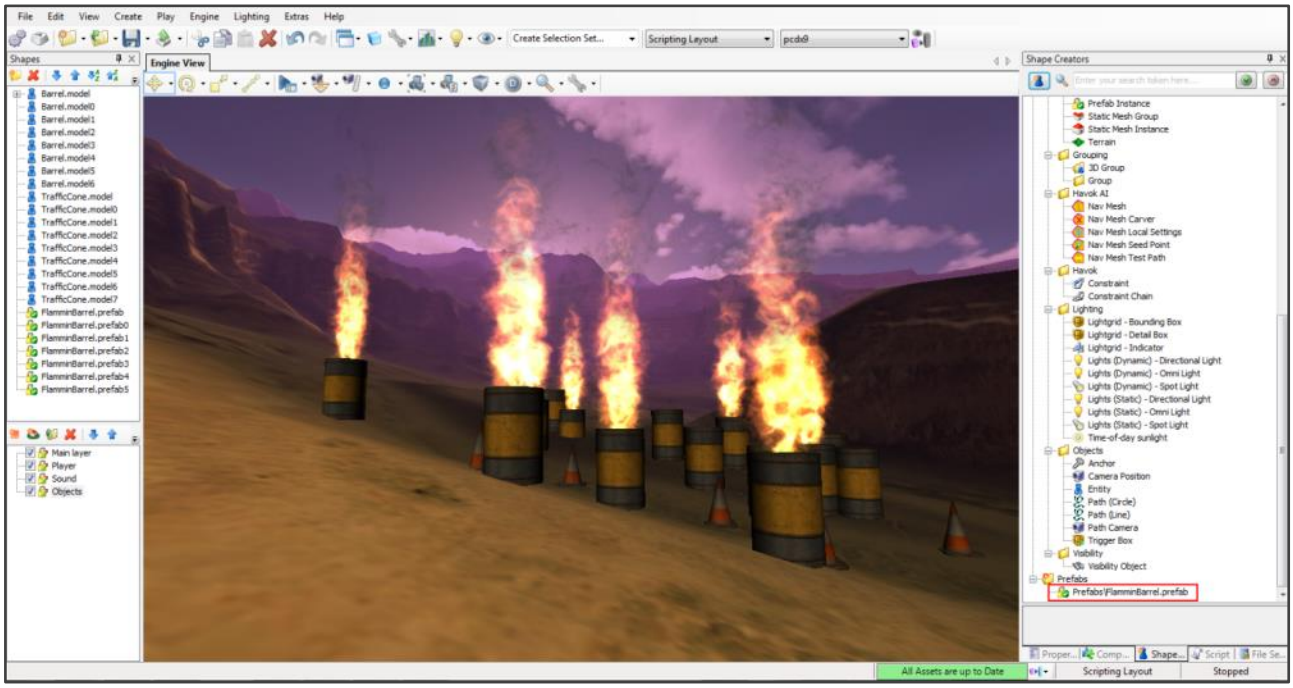


Select ***Play the Game*** and view the results.

Creating Prefabs

In the ***Shapes*** window, *RMB* on the Barrel with the flame particle effect, select **Create Prefab** and in the ***Create File*** dialogue set the **Name** to **"Flaming Barrel"**. Press *OK* then *OK* on the following ***Edit Properties*** dialogue.

Select the ***Shape Creators*** tab and open the ***Prefabs*** folder. The new prefab should be listed. Drag the prefab in to the scene a few times to populate with flaming barrels.



Select **Play the Game** and view the results.

Re-export the scene and reload onto device.

Appendix

A: USB Tunnel

It's possible to create an ethernet tunnel over USB which works with the **vFileServe**.

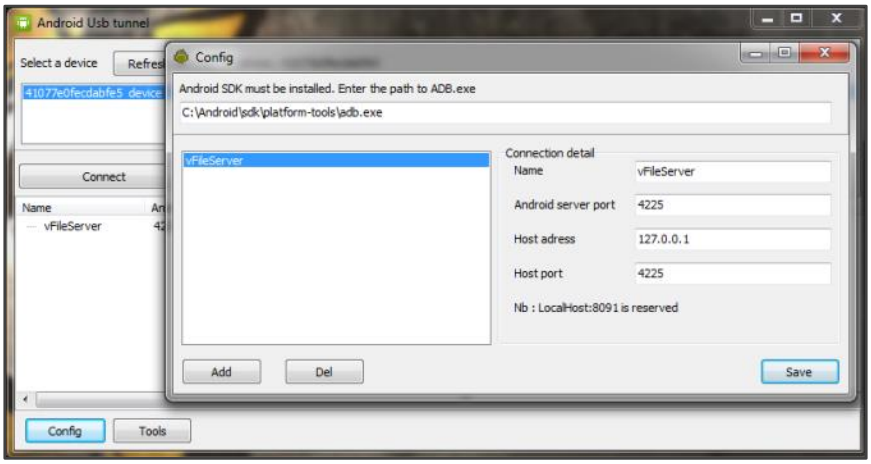
The main instructions can be found here http://www.codeproject.com/Articles/...ort-Forwarding.

Download Tracetool_service.zip and AndroidTool.zip and extract the contents.

Install Tracetool service.apk on the device with

```
adb install Tracetool service.apk
```

Open AndroidTool.exe on the host machine, select ***Config*** and add a connection as shown



Restart the AndroidTool.exe and then select connect. It should connect successfully

On the device, edit **"sdcard/vFileServeHost.txt"** to read: http://127.0.0.1:4225

If not present, then create the file on the host PC and:

```
adb push vFileServeHost.txt sdcard/vFileServeHost.txt
```

Launch the USB tunnel app with ***Show android interface***

From vForge, open the ***File Serve*** tab and ensure the data directories are correctly added (see *Deploy to Device*). If ***Prepare Devices*** is used, then it will be necessary to re-edit the **"vFileServeHost.txt"** file as above.

Launch the **vSceneViewer** app. It should connect to the **vFileServe**.