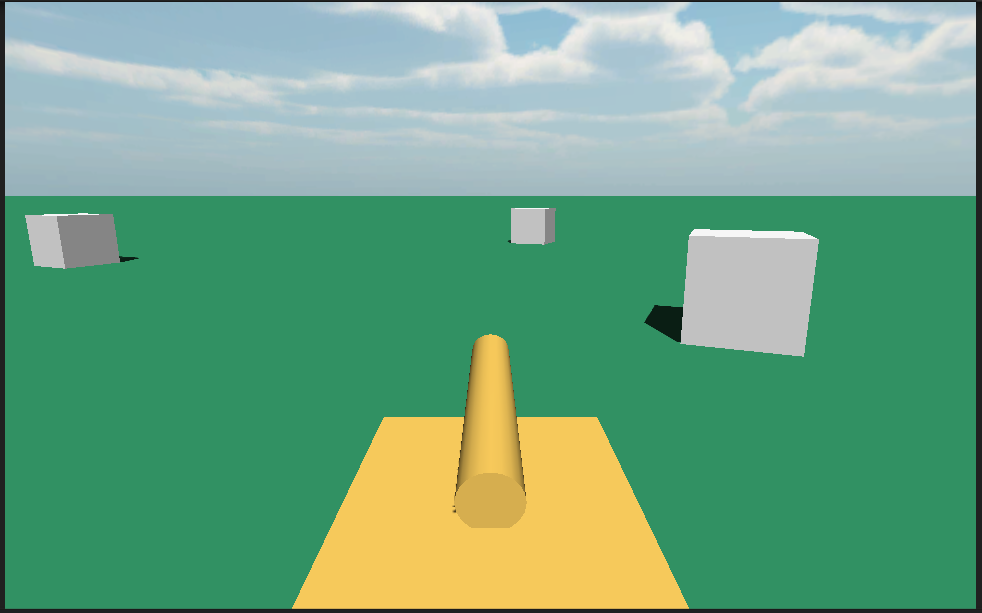
Tank tutorial week 2.

# Overview.

You will now be able to see the tank is moving as well as turning.



This week you are going to work with colliders and triggers to allow the player to bump into objects and be able to shoot objects.

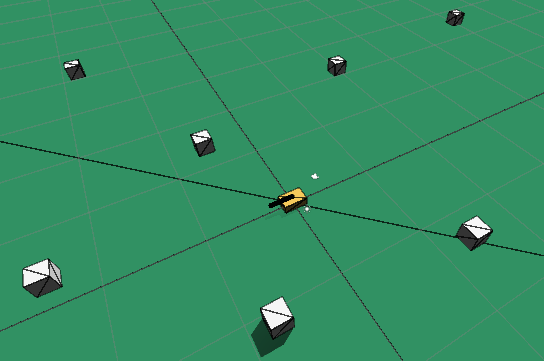
You will follow these steps to complete.

1. Add cubes to the terrain if you haven’t already.
2. Apply colliders to them.
3. Create a simple bullet object.
4. Add a trigger to it.
5. Prepare the collision/trigger system to bullets hit cubes.
6. Create a firing node for the tank.
7. Add a collision system for the tank that works with Unity’s terrain as well.
8. Script a firing solution to launch bullets.
9. Add trigger detect on the bullets to affect the cubes.
10. Add sounds to enhance the effects.
11. Allow the tank to push the cubes around.

### Adding some visual detail.

If you don’t have cubes do this section otherwise skip to the next section. You can see that the tank is moving only turning. You will need to add some objects to create this effect.

1. Add a cube to the scene.
2. Duplicate it many time and move the copies around the scene.



1. Run it again after turning on Maximising On Play in the Game window and run it full screen.
2. You will now be able to see the tank is moving as well as turning. Save you work for next week.

## Making the cubes collidable.

Unity has a physics system that requires 3 components for collisions to work.

* A collider of some type on one object
* A collider of some type on the other object
* At least one of them to also have a Rigidbody on it. (when both are static)

**Colliders** can be the following types on 3D objects

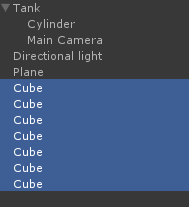
* Box
* Sphere
* Mesh

**Rigidbody** is a term used for a component that uses physics calculations. Just remember if you game object is not bouncing, rolling, falling or colliding you probably need to add a rigidbody to it. Also note that a lot of the time game objects don’t always need complex physics calculations placed on them.

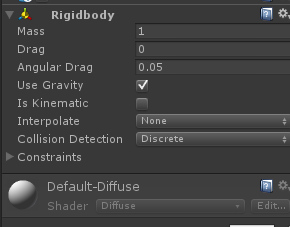
The mesh collider is dead accurate but if you have too many objects in the scene with mesh colliders and the speed of the game drops rapidly. Most of the time use either a cube or sphere, or if a more refined shape for collision is required then use a collection of cubes and spheres unless it’s absolutely necessary for polygon perfect collision. Even a low fidelity collision mesh can be generated for the terrain object in Unity to speed up the game.

You will use Box colliders for now as you just have cube objects. Unless you did something unorthodox the cubes already have Box colliders on them.

1. Select all the cubes at once using CTRL or Shift click – objects of the same type with the same components can be edited simultaneously.



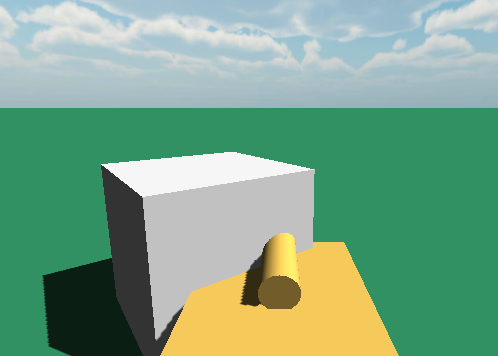
1. Click on the Add Component once more will all cubes selected and type in Rigi then select the Rigidbody component (NOT the Rigidbody 2D component)



## Add collision to the tank.

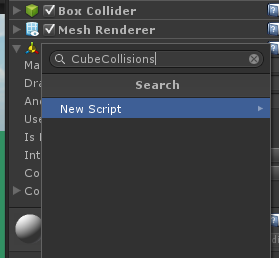
Remember that for a collision to occur you need both objects to have colliders on them and at least one to have a rigidbody. The cubes do already so you just need to add a box collider to the tank, but if you remember the tank was also created from a cube so it already has the Box collider on it.

Try the game now and see what happens?

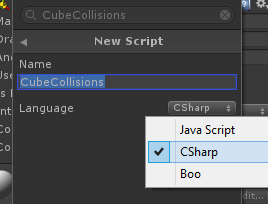


It goes through it. But if we were to test for it in code it should collide. Hint it never does until we make some changes – this is just a troubleshooting technique that follows. Following this procedure will cement in your head the steps to follow to fix problems like this. You will add that functionality to make it collide.

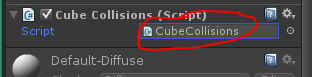
1. Click on the first cube and then click on Add Component.
2. Type the words CubeCollisions (all one word), and click on New Script.



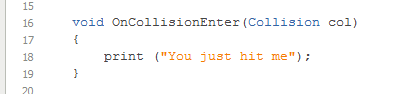
1. Select the dropdown and change it to C#



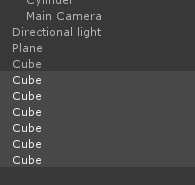
1. When the newly added Component appears double click on the script itself to open Mono Develop.



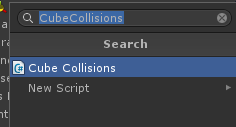
1. Once Mono Develop opens add this code



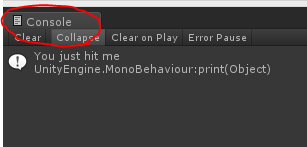
1. Now select all the other cubes



1. In the Inspector click on Add Component and type in CubeCollisions (it show CubeCollision as you type)



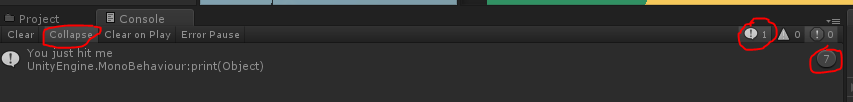
1. Click on Window->Console to see the console.
2. If you have never had this window open before it will be floating / separate from Unity. The lecturer will show you how but you need to click on the tab circled and hold the mouse down, then drag it to dock it alongside the Project tab.



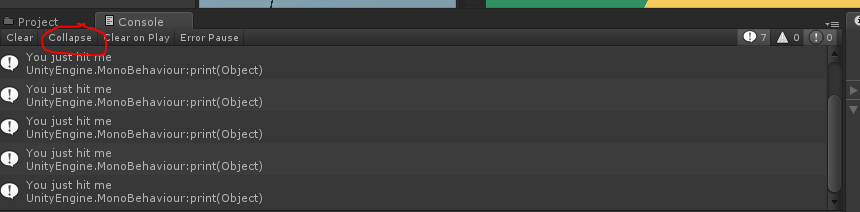
1. It should look like this when you have docked it successfully.



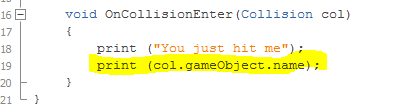
1. This window lets you see the print command you just typed in WHEN it happens – a good way to check how a program is responding from the inside.
2. Run the game now and you will see this in the Console.



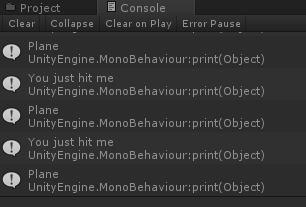
1. The Collapse is collapsing all similar print statements into one group.
2. The Exclamation mark circled with 1 next to it is set so it shows print statements in the console. Turning this off ignores print statements from appearing in the console.
3. The circled 7 at the end (you’re will be the number of cubes you have) says the print statement “you just hit me” was printed 7 times.
4. All the cubes collided with the floor.
5. They reported this to the console.
6. If you turn off Collapse it will look like this.



1. It expands all the comments.
2. The print statement only happening once per object proves that collision happened with the cube and something else and that Unity ignored any subsequent collisions. You need to remember this.
3. But how can we prove what hit the cube?
4. Change the code to this.



1. Run it and then look at the console.



1. So it’s confirmed. The plane hit the cubes.
2. Press the Clear button on the console.
3. Run the tank into the cubes now and check the console. Any change?
4. Why? Because the collision system already registered a collision between cubes and the plane and to optimise the performance of the game no more collision are detected.

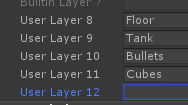
## Masking collisions with layers.

Colliders can you layers so that some objects can ignore others. In this case you want the cubes to ignore collisions with the plane (the floor) because it’s not relevant.

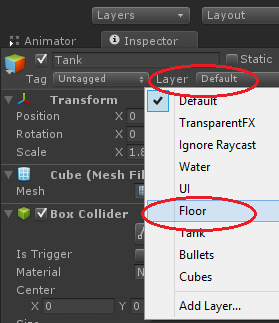
1. Add layers by clicking on the Layers button at the top right.



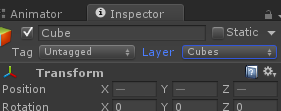
1. Click on Edit Layers
2. Add these layers by typing into the blank spaces.
3. You can only start from User Layer 8.



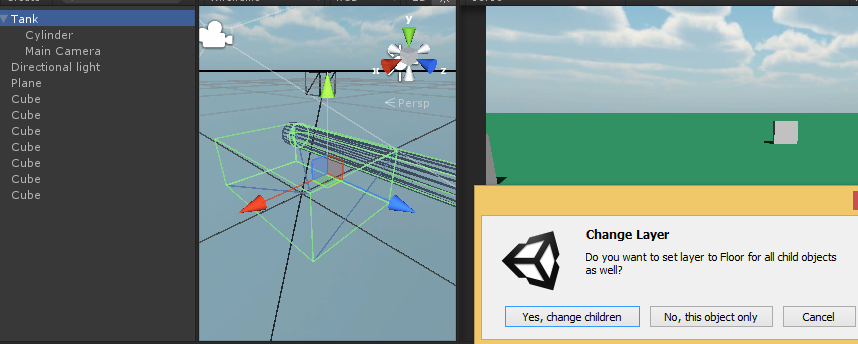
1. If you scroll down you see there the last one is 31. As the numbering system starts from 0 there are actually 32 layers. In nerd terms this is using a 32bit mask. Each layer represents a bit. Computers love numbers like 32 and so using this 32 bit mask, collisions are handled rapidly and efficiently in memory – that’s all you need to know. Look away now :D These aren’t the droids you’re looking for. You can about your business. Move along, move along…
2. Click on the Plane then in the Inspector change it layer which is currently Default to Plane.



1. Select all the cubes at once and change their layer to Cube.

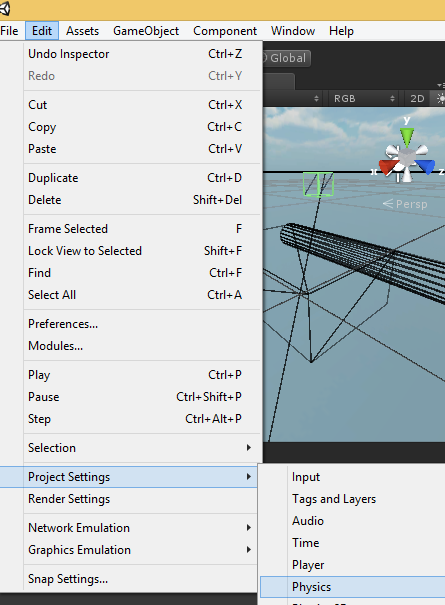


1. Next select the Tank at its root and change its layer to Tank.
2. This message will pop up – Select the YES option. This will make all the child objects under this also set their layer to Tank



Now that all objects have their layer set you will be able to set up whether they can collide with each other or not.

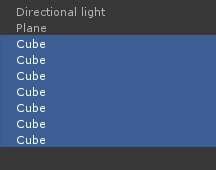
1. Click on Edit->Project Settings->Physics.



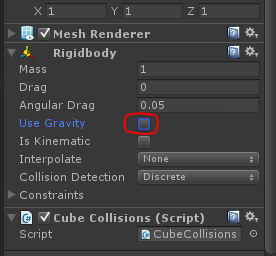
1. You will see this grid.
2. Untick Cubes/Floor and Bullet/Bullet
3. Run the game.
4. Wait a minute???? Where’d all the cubes go?

The cubes fell through the floor because they no longer collided with the floor. So they remain on the ground you will need to turn off gravity for the cubes.

1. Select all cubes.



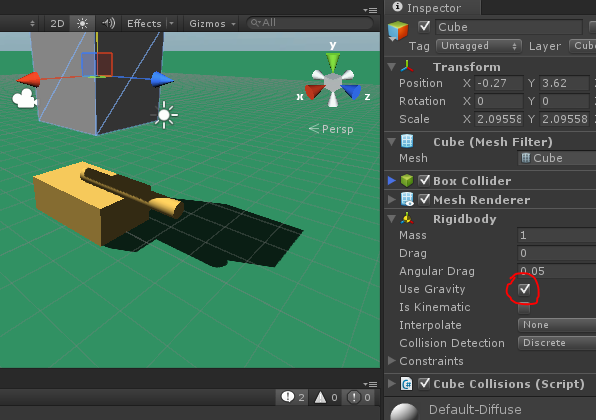
1. In the Rigidbody component untick Use Gravity



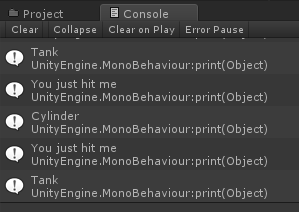
1. Run it again – the cubes are still there.
2. Run the tank into the cubes. Nothing comes up in the console.

Why? Because the tank is moving i.e not static and so it’s collider position in the world has changed. To prove it try this.

1. Drag one of the cubes so it’s above the tank in the Scene view.
2. Turn on Use Gravity for the cube.



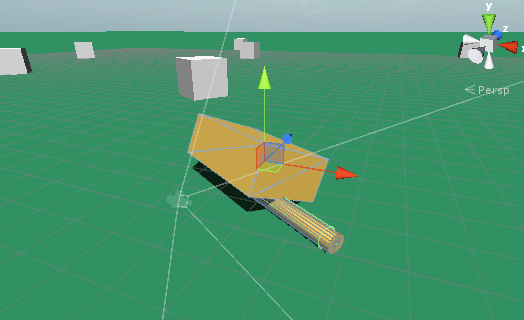
1. Play it. The tank gets hit several times as you can see in the console, until the cube rolls off and falls below the floor. Also the cylinder (gun turret) might get hit too.



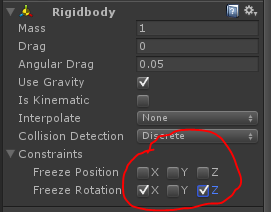
So the collisions work if everything is stationary and so you can reasonable assume as long as nothing moves everything can collide. But that also moving objects need rigidbody components added to them to allow for constant updates to check for collisions.

1. Add a rigidbody to the tank – turn on gravity
2. Select all the cubes – turn on gravity.
3. Go back into Edit->Project Settings ->Physics and tick the Cube/Floor box.
4. Play the game now and you can move the cubes around.

Drive around a bit. You’ll have some fun with it but eventually this will happen. To stop the tank flipping you need to constraint its X and Z axis.



1. Click on the Tank
2. Under Rigidbody expand the Constraints section. Tick Freeze rotation on X and Z (not position)



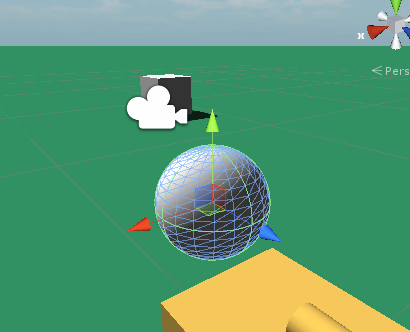
Driving around now it’s like CUBE SOCCER!!!!! If you haven’t tried it already select all the cubes at once and turn off their gravity, then drive the tank around – the cubes tends to float.

## Create a simple bullet object.

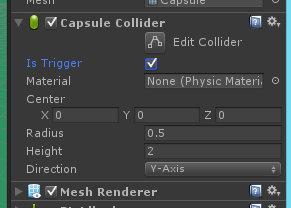
Now that you have mastered collisions, you can add the bullets into the equation. You want the bullets to penetrate the cubes though so this time you need to make colliders trigger-able. This means that an objects can pass into another object it is a trigger. You can check for that in script. Firstly it’s time to make the bullet.

1. Right click in the hierarchy and select 3D Object ->Sphere.
2. Name it Bullet
3. Change scale to 0.3, 0.3, 0.3 and make sure it looks like this.

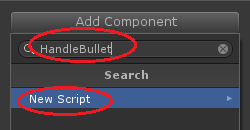




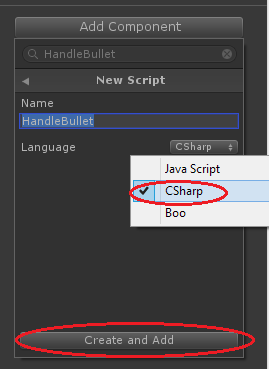
1. Still in the Inspector click on Add Component -> Rigidbody
2. Tick “Is Trigger” on the Capsule collider component.



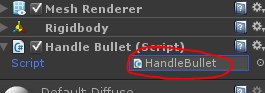
1. Add Component and type in HandleBullet (all one word) and click on New Script



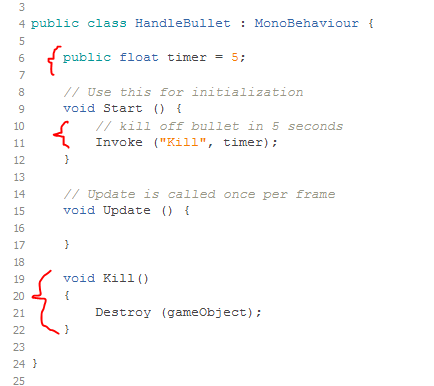
1. Select C# from the drop down and then click on Create and Add



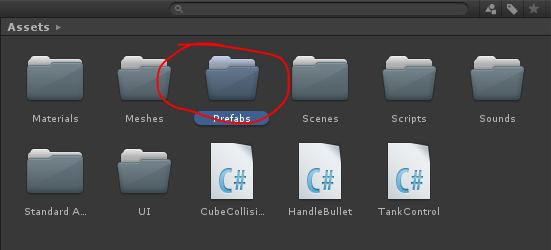
1. Double left click on the newly created component where indicated below.



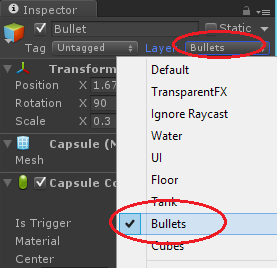
1. Add this code. This will start a timer then destroy the bullet after 5 seconds if it didn’t hit anything.



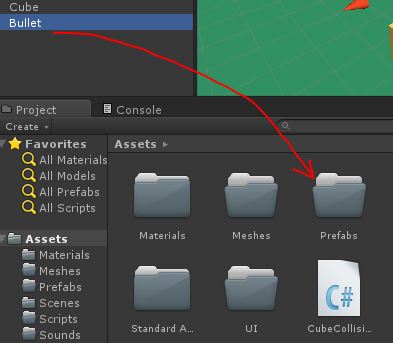
1. Create the Assets folder create a folder called Prefabs.



1. Change the Layer to Bullet



1. From the hierarchy drag the bullet to the prefabs folder.



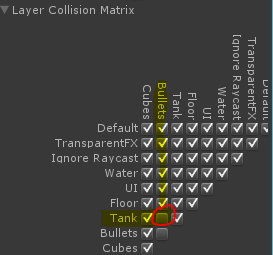
1. Delete the bullet from the hierarchy as it’s saved in the prefabs folder now.

It will now be a **prefab**ricated game object with all the stuff to make it work when fired.

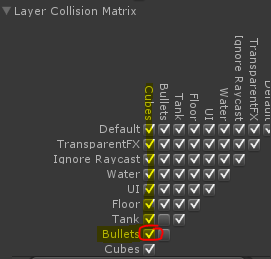
## Prepare the collision/trigger system to bullets hit cubes.

The bullet when activated will fly out of the turret’s nozzle and it will most probably collide with the turret itself. Also you don’t want it to hit the player tank. You can change the collision masking results to do this.

1. From the menus at the top of the screen, click on Edit->Project Settings->Physics
2. In the Layer Collision Matrix find Tank/Bullets and uncheck it.



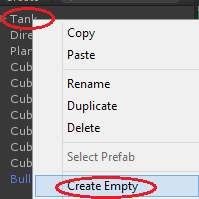
1. Make sure Cubes and Bullets collide.



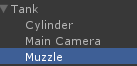
## Create a firing node for the tank.

Quite often you will create an object like the tank and you want the bullet to eject from a particular point. If you create the bullet from the position of the tank it wont look right. So you need to make a node or a point in space where you want the bullet to come from linked with the tank.

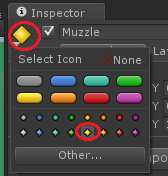
1. Right click on the Tank in the Hierarchy and create an Empty object



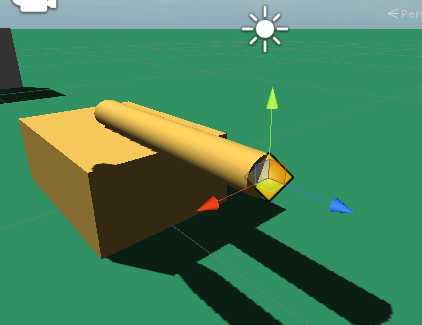
1. Call it Muzzle.



1. Add a marker for it from the Colourful cube and pick the yellow diamond. This will help you place in 3D.



1. Us the W key or move tool to place it at the end of the turret. Make sure the BLUE arrow faces the direction you want the bullet to fire. Rotate the muzzle node if required.

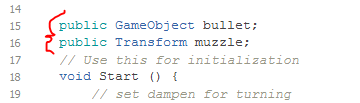


## Script a firing solution to launch bullets.

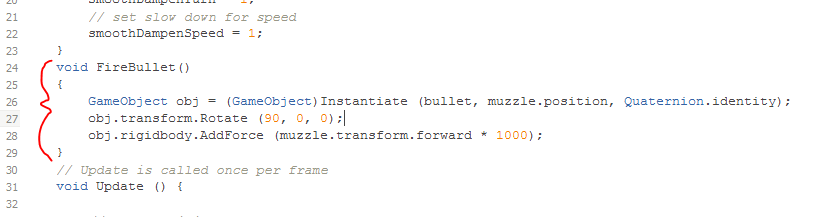
To fire a bullet is a fairly complex concept to understand but easy to implement. The game waits for the player to press the fire button, at which point a bullet is created at the muzzle point and force is added to shunt it along its way. You probably haven’t turned off rigidbody, unless you were mindful enough to think that the bullet is going to eject and then follow a ballistic curve out of the muzzle.

All projectiles in real life work this way even space ships so you will first program the firing solution and then see the ballistic curve it creates.

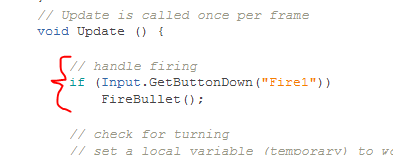
1. Select the Tank and double left click on the TankControl script. By doing that you will be taken to MonoDevelop
2. Add this code at the top. This will track the bullet and muzzle.



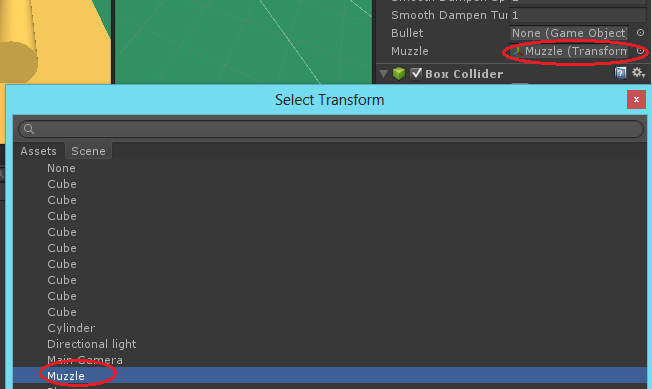
1. Add this code above Update – this will fire the bullet.



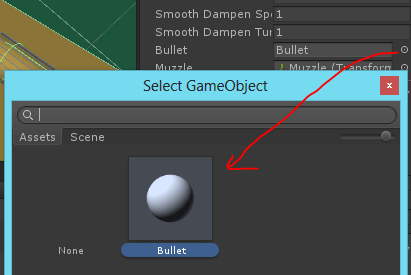
1. Add this code in Update – this will wait for the fire button to be pressed.



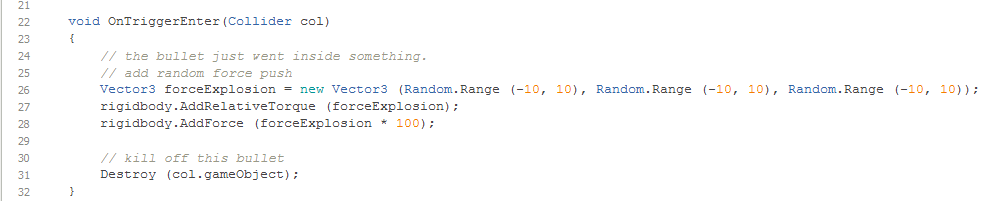
1. Save and flick back to Unity
2. In the Muzzle slot click on the circle at the end and select Muzzle.



1. In the Bullet slot click on the circle at the end and select Bullet from the Prefabs folder – probably the only choice you have.



1. Run the game and use the Mouse button or Left CTRL to fire – the bullet pops out it falls too short. This is gravity catching up and making it fall.
2. Go into the Prefabs folder and change the Use Gravity to false and run again.
3. Click on any Cube and double left click on the CubeCollision script.
4. Add this code at the bottom – this will respond when the bullets trigger hits it.

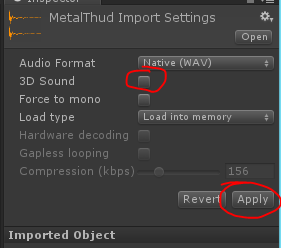


It’s kinda dumb and not realistic but it’s design to show you how to push objects around.

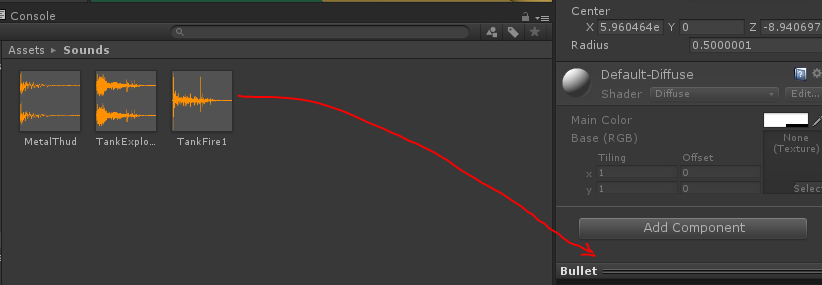
## Add sounds to enhance the effects.

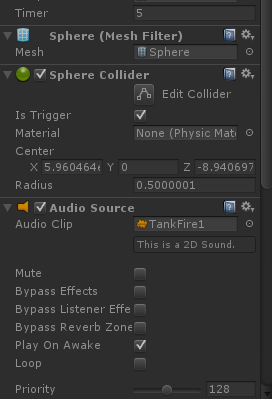
Finally add some sounds.

1. Import MetalThud, TankExplodes and TankFire1 to the Sounds folder but either right clicking in the Sound folder and selecting Import New asset to bring them in one at a time, or find the sounds on the G drive under Session 2 and drag all 3 in at once to the Sounds folder.
2. Select each one and untick 3D sound then click on Apply



1. Open the Prefabs folder.
2. Tap on the Bullet once to reveal the Inspector for it.
3. Tap on the Sounds folder again once to show the sounds.
4. Drag the TankFire1 into the grey space below the ADD COMPONENT button in the Bullets inspector.

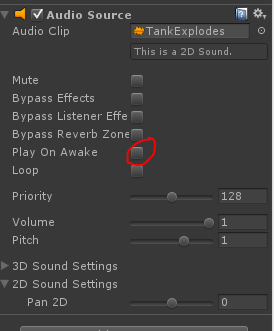




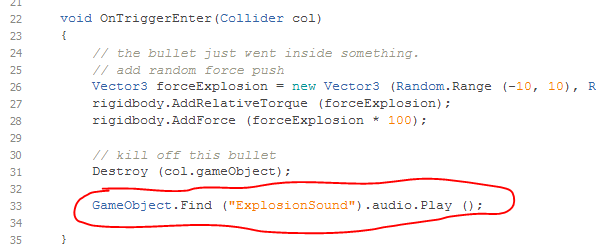
1. Play the game and every time you fire you will hear the firing sounds because as the bullet spawns it also plays the sound you just added.

Finally you will add the explosion sound when hitting a cube.

1. Create an Empty GameObject in the scene.
2. Call is explicitly ExplosionSound.
3. Tap once on the Sounds folder.
4. Drag the TankExplodes sound onto the ExplosionSound
5. Untick Play On Awake in the Audio Source component now sitting on the ExplosionSound



1. Open CubeCollisions and add this code at the bottom of OnTriggerEnter



1. Play the game and you can hear when you hit the cubes.