WEAPON THROW GAME MECHANIC

Written by: **Murad Elboudy (PATHIRAL)**

INTRODUCTION:

Hello and thank you for buying *WeaponThrow*. A mobile-optimized highly performant triple-A game mechanic. Suitable for both consoles/pc and mobiles.

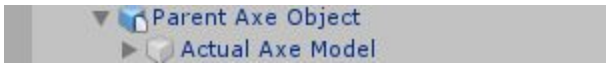
This awesome mechanic which has been seen in some latest games.
You obviously have access to the source code which is 100% documented, the core script of this mechanic as well as all the scripts of the demos. Everything has been commented to walk you through the code. It's written in a clean and simple way. So even if you'd want to edit/change something or add a cool new feature. It would be a walk in the park.

Special care has been put into performance so that it could be easily ported to mobiles. Imagine you can come up with an IO game or a hyper-casual which utilises this mechanic. I'm pretty sure you would impress.

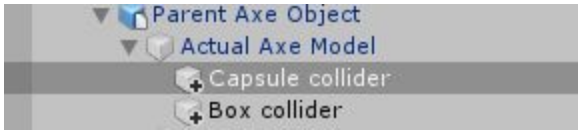
The two added demos contain basically everything you would need to start using this mechanic. Please take special care of how everything is setup and follow with care. In this PDF guide we'll walkthrough how to setup everything.

SETUP:

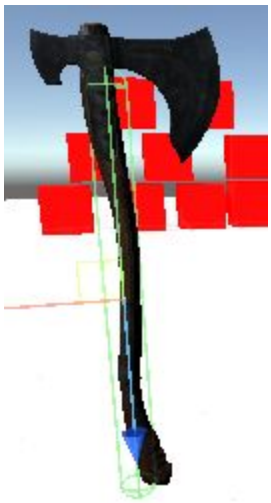
- Start out by making an empty gameobject. We'll call it **Parent Axe Object.** This will act as the parent gameobject for your wepaon/3d model.
- Inside this parent gameobject, insert your actual weapon model, as seen below:



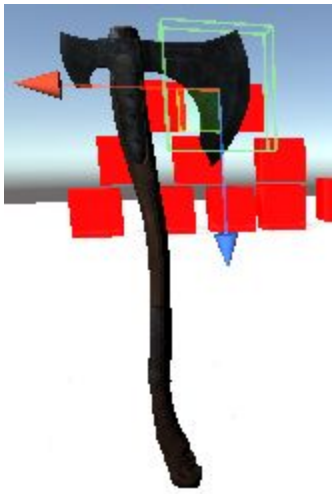
- Inside the Actual Axe Model object insert empty gameobjects, these gameobjects will be the colliders. Put as many as you need to cover the entire the weapon. In this case - the case of the axe - we will only need 2:



The capsule collider for the hand (holdable part of the axe)

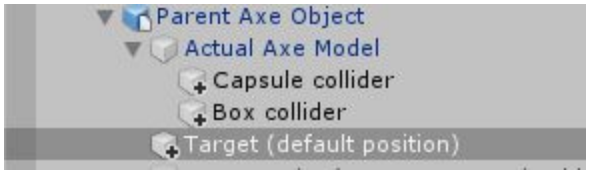


The Box collider for the actual steel axe part of the axe



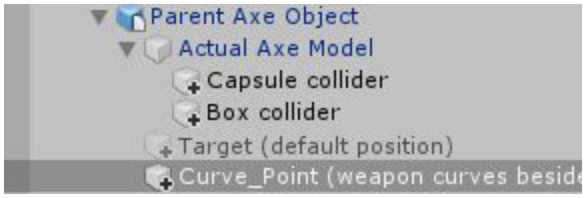
If your model already comes with colliders or you have a better way you can definitely just go with anything you want. But you need colliders for collisions to work

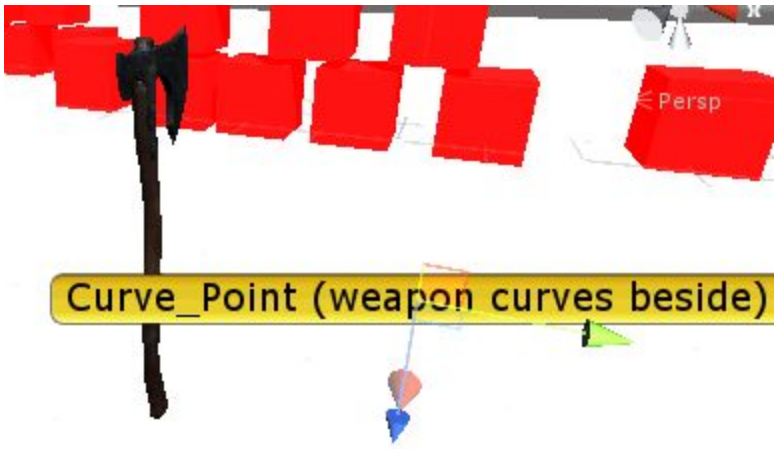
- Now your our axe is ready for collisions but we're not ready yet.
- Back in our parent object make two other child gameobjects, one will be the position that our axe will come back to, the other will be the point position which the axe will curve beside.
- We'll call the first one target. The target is the end position of our axe.



In most use cases, make the target object the exact same as your weapon model.

- The second gameobject is the curve point. The curve point is the position which the weapon will curve/slide beside until it reaches the target which is the end point. It is best to make it the same as the target then offset it a little bit on the x-axis either to the left or right according to your needs. Basically it is best to make it beside the weapon in either direction. As seen below:





As you can see we put the curve_point beside the weapon

- Now we're ready to add the script component. On the Actual Axe Model, add component and add Weapon Throw. It will automatically add a rigidbody component.
- You can now get the script and trigger several methods.

SCRIPT METHODS:
You can get the WeaponThrow script as component and then trigger several methods.

1. Throw() - throws the weapon
2. ReturnWeapon() - returns the weapon back
3. isThrown - returns a bool true/false, whether the weapon has been thrown or not
4. reachedEnd - YOU NEED TO CHECK FOR THIS VARIABLE IN THE UPDATE METHOD. Returns true when weapon reaches end. You need to turn it back to false in the update.