

# SmoothTurret

## Tutorials

### Setting up a turret for player control

To make one or more turrets player controlled, you need an object in the scene to represent the player. This object doesn't have to be visible, or be controllable, it just needs to exist to hold the ST\_Player script.

1. Attach the ST\_Player script to the player object.
2. Assign an object in the scene to the Aim Object field to represent the aim point.  
If this is left blank, a blank object will be created.
3. Make sure Turret Control is checked.
4. On the turret, in the ST\_TurretController script, change the control menu to "Player".
5. Then, drag the player object into the player field.

You may assign multiple turrets to a player, but you may not assign more than one player to a single turret.

### Creating your own turret

A blank turret prefab with the proper hierarchy has been provided to make things easier.

1. From the Prefabs > Turrets folder, drag the Blank Turret Template prefab into your scene.
2. From the GameObject menu, select Break Prefab Instance, so you don't accidentally modify the template prefab.

In the inspector, you can see the object hierarchy:

**Blank Turret Template**

**Rotator**

**Elevator**

**Mount**

The root object, Blank Turret Template, has the ST\_Turret and ST\_TurretControl scripts attached.

Next, we'll add 3D objects to represent the turret parts.

Instead of placing mesh components directly to the parts, it is highly recommended to give these their own GameObjects placed as children of the part. This will allow you to freely move, scale, or rotate the mesh object.

1. Place your base mesh as a child of the Blank Turret Template, and move, scale, or rotate it as desired.
2. If needed, move the Rotator object so that it's pivot represents the center of rotation for your turret. It need not be centered with the root object, but don't change its rotation.

3. Place the mesh associated with the rotation part as a child of of the Rotator. Move, scale, or rotate the mesh object as desired. Match the direction you want to be forward to the positive x axis.
4. Move the elevator object so that its pivot is where its elevation movement should be centered. This need not be centered with the Rotator, but don't change its rotation.
5. Place the mesh associated with the elevator part as a child of the Elevator. Move, scale, or rotate the mesh object as desired. Match the direction you want to be forward to the positive x axis.
6. Now drag your weapon object as a child of the Mount object. Keep the weapon's rotation at (0,0,0) and it should be centered on the Mount object.  
If you want to add more than one weapon, make the mount's pivot centered between the weapons.

In the inspector, your object hierarchy should look something like this:

### **Blank Turret Template**

**GameObject** (with mesh)

**Rotator**

**GameObject** (with mesh)

**Elevator**

**GameObject** (with mesh)

**Mount**

**Weapon Object**

If this is a weapon turret, you'll need the ST\_TurretControl script to bridge communication between the weapon and the turret. This is already on the root object of your turret.

1. Change the weapons field size to the number of weapons on the turret.
2. For each weapon, drag and drop it into the new corresponding object fields.

This completes the basic set-up of the GameObjects, but a few decisions must still be made.

See the Target Selection tutorial below for the next steps.

## **Automatic and Manual Target Selection**

There are a number of ways the turret can detect targets.

- Designate the target manually, or have another script provide the target.

- MF\_BasicTargeting can supply the turret a target from a list provided by MF\_BasicScanner and MF\_TargetList.

Note, that if multiple turrets are to use the same target list, they can all point to a single list, and they may all pick different targets. In this way you can have less scanners running, and improve performance.

If the MF\_TargetList script is at the root turret object or the root object of what the turret might be attached to, you can leave the Target List Object field of MF\_BasicTargeting blank.

If the list is anywhere else, you'll have to drag and drop the object holding the script into the Target List Object field.

For now, we'll set up the turret to choose its own targets.

1. In the Prefabs > Other folder, you can drag a target into the scene. Alternately, you can use an object you already have in your scene.
2. Create a new tag called, "Side1" and change the tag of this target object to the new tag.  
Tags may be created using the Unity3D menu: Edit > Project Settings > Tags and Layers.  
You can rename the tags SmoothTurret uses, but more on that below.
3. Duplicate this target a few times, if you want, and move them around.
4. From the Scripts folder, drag MF\_BasicTargeting to the root object of your turret. You won't need to change anything on this script just yet.
5. Next, drag MF\_TargetList to the root object of your turret.
6. And then drag MF\_BasicScanner to the root object of your turret.
7. On MF\_BasicScanner, change the Targetable Factions field size to 1, and notice that a new drop-down menu appeared. Change it to Side 1, so it will target factions tagged Side1.
8. Set the Scan Range to a range that will reach at least one of your target objects.
9. Set Scan Interval to 1 second.
10. Press Play, and the turret should now track the closest target. You can change the target priority to the farthest target (in range of the scanner) by selecting it from the drop-down menu of the Priority variable on MF\_BasicTargeting.

## Customizing Tag and Layer names

At some point you may wish to add or change the tag or layer names that MF\_BasicScanner uses.

1. Open the MF\_BasicTargetData script in the editor.
2. Simply change the following line near the top of the list:

```
enum FactionType { Side0, Side1, Side2, Side3 };
```

Add or replace Side0, Side1, etc. with whatever names you want to use.

3. Save the script with your changes.