RUNTIME TRANSFORM GIZMOS – QUICK START GUIDE

Welcome to **Runtime Transform Gizmos**!

This document is meant to serve as a quick start guide for the **Runtime Transform Gizmos** pack and is not meant to be a complete reference. It covers the most important UI aspects and the basics of the scripting API. However, for a more complete coverage of the scripting API, please see the online docs and video tutorials:

- API Docs
- Video Tutorials

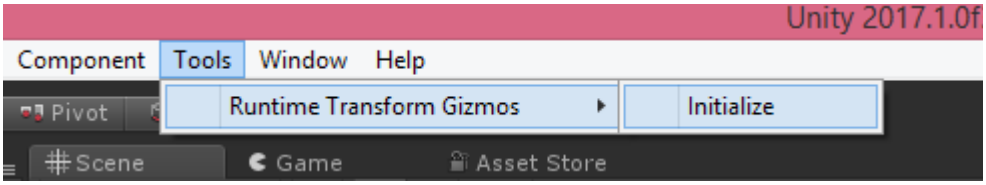
The tutorial scenes and scripts can be found in the **Assets/Runtime Transform Gizmos/Scripting Tutorials** folder.

Support E-Mail: octamodius@yahoo.com

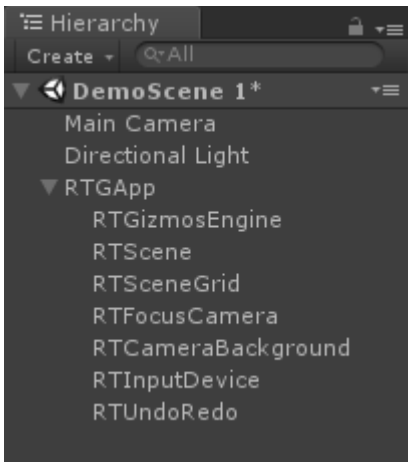
Plugin Initialization

After you have imported the package inside your project, the following steps are needed to initialize the plugin:

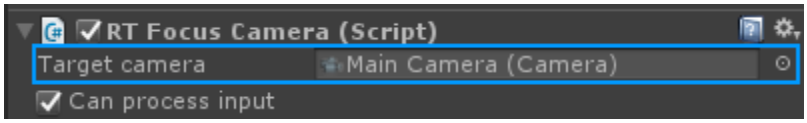
- in the top menu, go to **Tools->Runtime Transform Gizmos** and click on **Initialize:**



- the hierarchy view should look like in the following image:



- in case your scene **does not** contain a camera tagged as **Main Camera,** you also need to select the **RTFocusCamera** object in the hierarchy view and drag and drop a camera object in the **Target camera** field:



Demo Scene

You can get a feel for how the gizmos work by playing around with the demo scene. The demo scene can be found in the following folder: **Assets/Runtime Transform Gizmos/Demos/.** The name of the scene is **Demo.**

Hotkeys

By default, the hotkeys are setup in the following way:

Camera:

- **LMB** + **WASDQE** - move camera forward, left, backwards, right, down and up respectively;
- **LMB** + **MOUSE MOVE** - rotate to look around;
- **LMB** + **LALT** + **MOUSE MOVE** - orbit around focus point. If the camera hasn't been focused, the default focus point is the world origin;
- **MMB** + **MOUSE MOVE** - pan;
- **MOUSE SCROLL WHEEL** - zoom in/zoom out;

Scene Grid:

- **[** - move grid down;
- **]** - move grid up;
- **LALT** + **LEFT CLICK** - snaps the grid Y position to the intersection between the mouse cursor and an object in the scene;
- **LALT** + **LEFT DBL CLICK** - same as above, but this time the position will snap to either the top or bottom of the object, depending which one is closer to the cursor pick point. Useful when you need to quickly place the grid on top or below objects. Works with arbitrary object and grid rotations.

Move Gizmo:

- **LSHIFT** - enable 2D mode;
- **LCTRL** - enable snapping;
- **V** - enable vertex snapping (requires scripting to setup. See **Vertex Snapping**.

Rotation Gizmo:

- **LCTRL** - enable snapping;

Scale Gizmo:

- **LCTRL** - enable snapping;

- **LSHIFT** - change multi-axis scale mode;

Universal Gizmo:

- **LSHIFT** - enable 2D mode;
- **LCTRL** - enable snapping;
- **V** - enable vertex snapping;

Undo/Redo:

- **LCTRL** + **Z** - Undo;
- **LCTRL** + **Y** – Redo;

Note: When using the plugin inside the Editor in PlayMode, Undo and Redo only work if you also hold down the L**SHIFT** key. So to undo you have to press **LCTR: + LSHIFT + Z.** To redo, you need to press **LCTR: + LSHIFT + Y.**

Available Gizmos

The pack implements the following gizmos:

- move gizmo – with vertex snapping;
- rotation gizmo;
- scale gizmo;
- universal gizmo – supports only uniform scale;
- scene gizmo;

Note: All gizmos support snapping when the **CTRL** key is pressed.

Transform Spaces

The gizmos support 2 different transform spaces: **Local** and **Global**. These are the same as the ones that the Unity Editor uses. More info on transform space can be found here.

Transform Pivots

The transform pivot allows you to specify a point of origin for the transform. The following transform pivots are available:

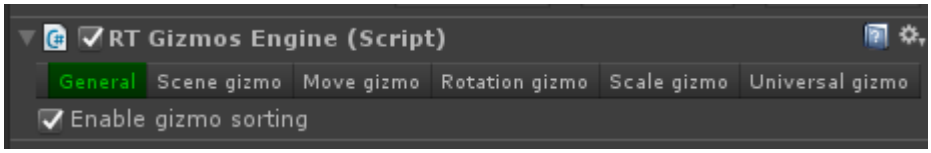
- **ObjectGroupCenter** - when multiple target objects are used, the center of the object collection is used as a pivot. When a single object is used as a target, the center of the object's bounds is used;
- **ObjectMeshPivot** - this is the pivot as was defined inside the modeling software and is the same as the position of the game object (i.e. **gameObject.transform.position**);
- **ObjectCenterPivot** - this is the center of an object's bounding volume. The gizmo will sit at the center of one of the object's volume center;

- **CustomWorldPivot** - a world position specified by you via a function call. When this pivot is active, the gizmo will sit at this position in the 3D world;
- **CustomObjectLocalPivot** - a pivot which is defined in the local space of an object. When this pivot is active, the gizmo position will be set to the postion of the local pivot transformed in world space.

Complete information on transform pivots can be found here.

Gizmo Settings/Look & Feel

In order to configure the way in which the gizmos behave and how they look, you have to click on the **RTGizmosEngine** object in the hierarchy view. Once you do this, the Inspector will show the following toolbar:



There is one tab for each gizmo available.

The gizmos settings (except for the scene gizmos) are divided in the following categories:

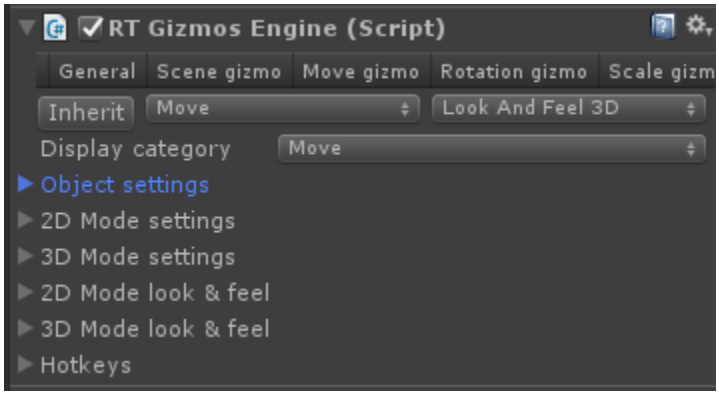
- **Object settings** – configure the way in which the gizmo transforms objects;
- **Settings** – configure gizmo behavior;
- **Look & feel** – configure gizmo appearance;
- **Hotkeys** – configure gizmo hotkeys;

The move gizmo is a special case because the **Settings** and **Look & Feel** categories are further divided into 2 subcategories each;

- **3D Mode settings**
- **2D Mode settings**
- **3D Mode look & feel**
- **2D Mode look & feel**

3D Mode refers to the mode in which the gizmo operates in its normal state. **2D Mode** refers to the mode in which the gizmo operates when you press the **SHIFT** key. In this mode a 2D shape will appear (by default, this shape is a rectangle) which allows you to move along the camera view axes.

The **Universal Gizmo** is also a bit special because of the fact that it's essentially almost 3 gizmos in one. So the UI has been designed to allow you to choose the category of settings that can be displayed:

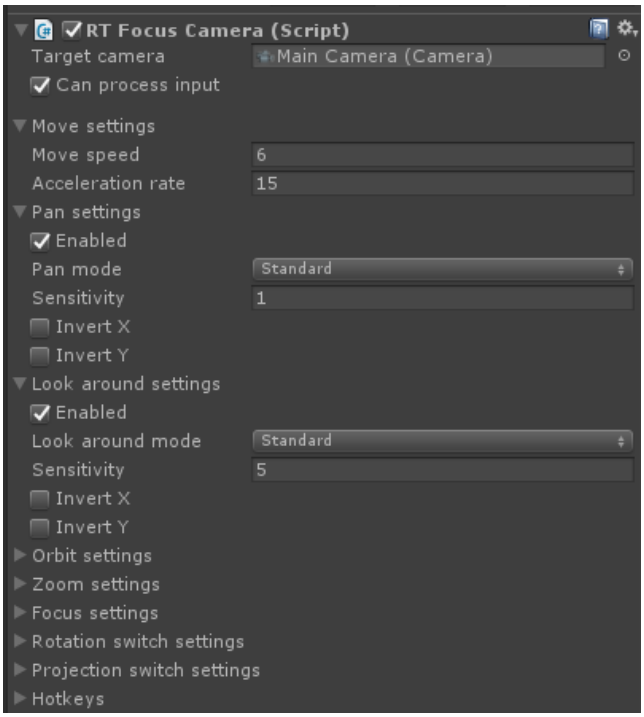


The **Display category** field allows you to select the category of settings to display: **Move, Rotate** or **Scale**. You can also inherit settings from the move, rotate and scale gizmos using the **Inherit** Button.

The Focus Camera

The plugin gives you access to a really nice navigation camera that you can use to navigate the scene. There's quite a lot of functionality packed into the camera script and a lot of settings that can be used to configure its behavior.

The camera can be found in the hierarchy view under then name **RTFocusCamera.** If you click on this object, the Inspector will allow you to customize the way in which the camera behaves:



These are the things that you can do with the camera:

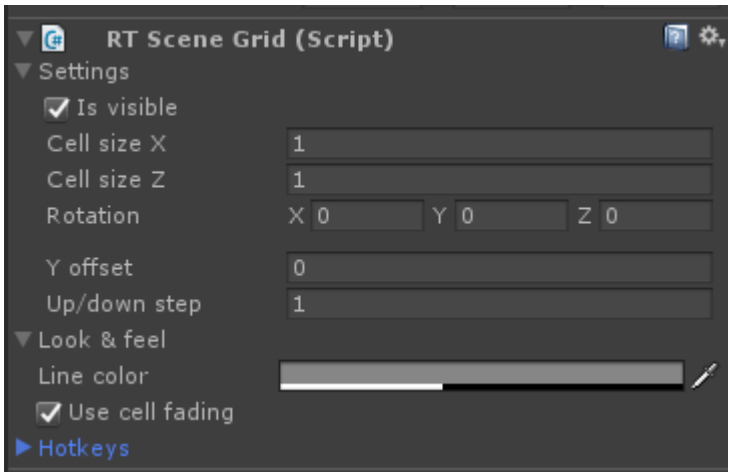
- move;
- pan;
- zoom;
- look around;
- orbit;
- projection switch;
- rotation switch;

Some operations such as rotation switch and projection switch are performed automatically when interacting with the scene gizmo.

The Scene Grid

The scene grid can be used for snapping. For example, when using the move gizmo, you can vertex snap to the grid cells.

If you select the **RTSceneGrid** object in the hierarchy view, the Inspector will allow you to modify the grid settings:



As you can see, you can control the grid visibility, the grid cell size and the grid rotation. The grid rotation can be useful in case you are working with sprites in which case you might decide to set the rotation to something like <90, 0, 0> (i.e. 90 degree rotation around the X axis).

The **Y offset** field allows you to control the offset of the grid along its local Y axis. When a zero rotation is used, this will change the Y position of the grid (move it up or down). When the grid is rotated, it will move the grid up or down along its local Y axis.

The **Up/down step** field allows you to control the rate of change in position along the grid's Y axis when using hotkeys.

Gizmo Scripting API – Quick Guide

This section serves as a quick start guide for the gizmo scripting API. It is not meant to be a complete coverage of the gizmo API, therefore it is highly recommended that you check out the online API Docs instead.

Creating Gizmos

You can create object transform gizmos using the **RTGizmosEngine** class like so:

```
ObjectTransformGizmo objectMoveGizmo = RTGizmosEngine.Get.CreateObjectMoveGizmo();
ObjectTransformGizmo objectRotationGizmo = RTGizmosEngine.Get.CreateObjectRotationGizmo();
ObjectTransformGizmo objectScaleGizmo = RTGizmosEngine.Get.CreateObjectScaleGizmo();
ObjectTransformGizmo objectUniversalGizmo = RTGizmosEngine.Get.CreateObjectUniversalGizmo();
```

These are all **object transform** gizmos that are meant to transform objects.

You can also create general purpose gizmos that can be used to transform other types of entities and are not directly tied to objects:

```
MoveGizmo moveGizmo = RTGizmosEngine.Get.CreateMoveGizmo();
RotationGizmo rotationGizmo = RTGizmosEngine.Get.CreateRotationGizmo();
ScaleGizmo scaleGizmo = RTGizmosEngine.Get.CreateScaleGizmo();
UniversalGizmo universalGizmo = RTGizmosEngine.Get.CreateUniversalGizmo();
```

When creating general purpose gizmos like above, you will have to listen to gizmo drag events to apply drag values to the entities that you wish to transform. See **Gizmo Events** for a complete list of available events that you can listen to.

Assigning Target Objects

The following code creates a move gizmo and then it calls the **SetTargetObject** function to tell the gizmo about the object that it should be transforming.

```
// Create a move gizmo and then set its target object.
// Note: The code assumes the scene contains a game object called "TargetObject".
ObjectTransformGizmo objectMoveGizmo = RTGizmosEngine.Get.CreateObjectMoveGizmo();
objectMoveGizmo.SetTargetObject(GameObject.Find("TargetObject"));
```

When you would like to transform a collection of objects, you need to use the **SetTargetObjects** function instead (notice the plural in 'objects').

The following example demonstrates how the **SetTargetObjects** function can be used:

```
class ObjectSelection
{
    private List<GameObject> _selectedObjects = new List<GameObject>();
    private ObjectTransformGizmo _objectMoveGizmo;

    private void Start()
    {
        _objectMoveGizmo = RTGizmosEngine.Get.CreateObjectMoveGizmo();
        _objectMoveGizmo.SetTargetObjects(_selectedObjects);
    }
}
```

Please see Assigning Target Objects for a complete coverage of this subject.

And Much More...

There is a lot more that you can do with the scripting API, such as changing transform spaces and transform pivots, acquiring gizmo hover information, gizmo events etc. All of these subjects are covered in the online API Doc.