

# Exploder Documentation

*Version 1.6.1*



---

## Table of contents

[1 Introduction](#)

[2 How does it work?](#)

[3 Quick start](#)

[4 How to use it in my FPS game?](#)

[5 Exploding 2D sprite \(Plane mesh\)](#)

[6 Cracking object \(pre-calculated explosion\)](#)

[7 Using multiple Exploders at the same time](#)

[8 Custom texture for inner side of exploded fragments](#)

[9 Deactivation options](#)

[10 ExploderObject settings](#)

[11 2D Physics \(Unity4.3+\)](#)

[12 UFPS integration](#)

[13 Playmaker support](#)

[14 FAQ](#)

[15 Support](#)

---

## 1 Introduction

**Exploder** is a Unity3d plugin that allows you to explode any GameObject that contains a mesh in **real-time**.

The destruction calculation is processed in **real time**. There are no pre-calculations or predefined GameObjects. Just put your beloved GameObject in the scene, Tag it with "Exploder" and watch the explosion!



Destroying vases from the Demo

---

## 2 How does it work?

The library contains a powerful **mesh cutter** that finds a mesh in your GameObject. After that it cuts the mesh recursively to small pieces, assigns a rigid body and velocity to each of them and makes an explosion.

For best possible performance the fragments are pre-allocated in a **pool** to minimize creating and instantiating GameObjects.

Cutting algorithm is very fast however you can specify **time budget** which is the maximum time to spend on calculation in a single frame. This allows you to create a powerful explosion in few frames with **low or no FPS drop**.

---

### 3 Quick start

The usage is **very simple**:

1. Add Exploder prefab to the scene hierarchy
2. Adjust parameters in the inspector (or from the code)
  - target fragments
  - etc...
3. Add script to your target object and call **ExplodeObject**:

```
using Exploder.Utils;
using UnityEngine;

public class QuickStart : MonoBehaviour
{
    void Start()
    {
        ExploderSingleton.ExploderInstance.ExplodeObject(gameObject);
    }
}
```

For the script reference please see the attached demo examples in [DemoClickExplode.cs](#).

Here's Quick Start Youtube video:

<http://youtu.be/u4WCLB6guBA>

---

## 4 How to use it in my FPS game?

**Tip:** from version 1.3.2 you can integrate Exploder with UFPS camera project, see more information in [this chapter](#).

This package also contains a first person shooter demo to demonstrate how the Exploder can be used with various weapons.



Shotgun example from the Demo

The basic idea:

1. Add Exploder prefab to the scene.
2. (Setup your weapon and camera) and shoot with your mouse.
3. At the same moment run a raycast in direction from the camera.
4. Get an intersecting GameObject (ex. Vase).
5. Move Exploder prefab to the position of the intersecting GameObject (ex. now the ExploderObject has the same position as the Vase).
6. Call ExplodeRadius() or ExplodeObject() in case you want to shoot only one object.
7. Watch explosion!



RPG example from the Demo

The setup for the RPG or Grenade is very similar. Just shoot a rocket (or throw a grenade) and when it hits the object (or after a timeout) move your ExploderObject to the same position and call ExplodeRadius().

You don't need to maintain several ExploderObject components. Just only one and always move it to desired position or

All **source code is included** so you can see how the demo is constructed.

---

## 5 Exploding 2D mesh based sprite

**Tested with 2D Toolkit.**

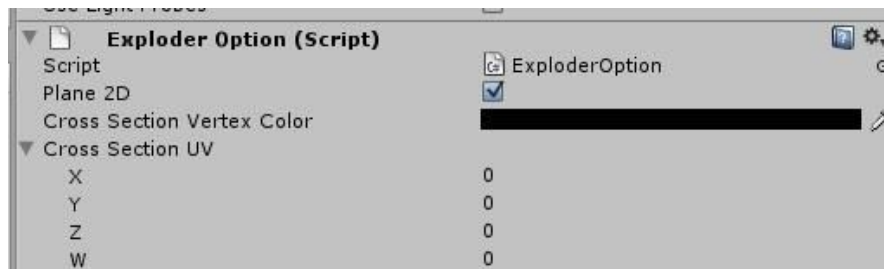
### IMPORTANT NOTE:

Exploder DOES NOT work with Unity built-in sprites, because there is no way to access sprite mesh data required by Exploder.

(For the 2D physics support, see the [2D Physics chapter](#))

Exploder also works very well with 2D mesh based sprites. The only thing you have to do is to mark the object in **ExploderOption** settings.

**ExploderOption** is another script object you can find in the asset folder together with ExploderObject. You have to **assign this script** to your game object (sprite) you want to explode. There are several options in the script:



As you can see there is a checkbox "*Plane 2D*". By selecting it you specify that this object (owner game object) is a 2d plane or a 2d sprite. Later when you run the explosion Exploder will look for the ExploderOption and if it is marked as 2d plane it will change the explosion calculation to work with 2d meshes.

### In short:

1. Find ExploderOption script and assign it to your 2d game object
2. Check the "*Plane 2D*" option
3. Run the explosion!



---

## 6 Cracking object (pre-calculated explosion)

**Cracking** the game object is another feature that allows you to run explosion calculation = prepare the object for explosion and later immediately execute the explosion.

The **advantage** of using the *crack* is **PERFORMANCE**. You can simply run the expensive calculation a long time before the explosion and then execute the explosion instantly. This can be useful especially on mobile devices, where the performance is critical.

### Example:

```
void Awake()
{
    exploder.CrackObject(obj1);
    exploder.CrackObject(obj2);
    exploder.CrackObject(obj3);
}

// later in the game
exploder.ExplodeCracked(obj1);
exploder.ExplodeCracked(obj2);
exploder.ExplodeCracked(obj3);
```

### Limitations:

The only limitation is the size of the pool. You can crack as many objects as your pool size can take. For example if your pool size is set to 500 (500 available fragments) and your *TargetFragments* is set to 50, you can crack up to  $500/50 = 10$  objects. If your *TargetFragments* is set to 10, you can crack up to  $500/10 = 50$  objects, etc.

### Demo sample:

Demo sample can be found in this file:

You only need to uncomment the macro **ENABLE\_CRACK\_AND\_EXPLODE** on the third line to see cracking in action.

---

## 7 Using multiple Exploders at the same time

Using multiple Exploder objects at the same time to achieve simultaneous explosions is **NOT SUPPORTED** and **NOT RECOMMENDED**.

The reason why is very simple. Exploder calculation can be very CPU expensive, during calculation it takes as much time as specified in FrameBudget parameter. Imagine this scenario with two Exploders:

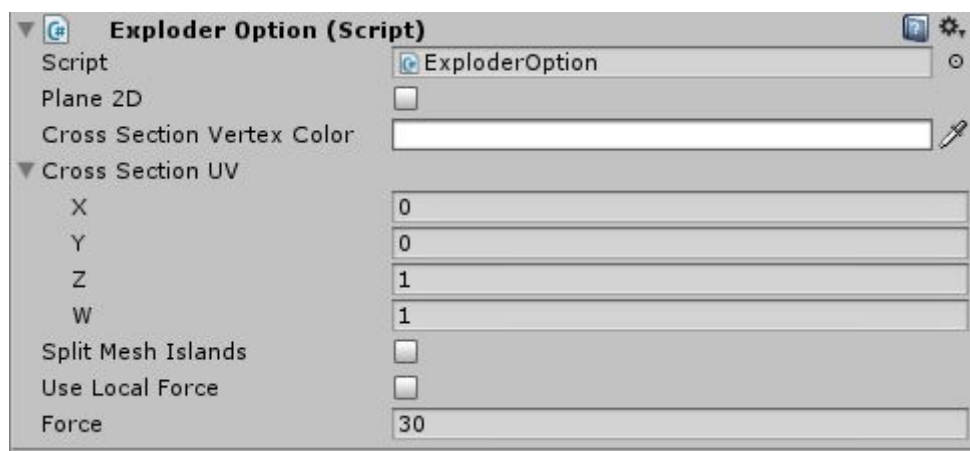
- Exploder\_1 with FrameBudget = 15 [ms]
- Exploder\_2 with FrameBudget = 15 [ms]
- Game is running in average 30 FPS, Exploder\_1 runs explosion and takes approximately half of the framerate, the game experience is still acceptable. Exploder\_2 runs another explosion at the same time and takes approximately another half of the framerate. The game **FPS reaches 0 and player is experiencing lag** that can take several frames.

If you really need to run as much explosions as possible in small amount of time, you can always increase the value of FrameBudget. For example if you set FrameBudget = 100, you can easily run more than one explosion in one or two frames, however it can heavily impact your framerate.

---

## 8 Custom texture for inner side of exploded fragments

Specifying custom texture to the inner side of the fragment can be important for your game. For example exploding alien with green blood should show green texture inside the fragments. Exploder allows you to do that in slightly different way. You cannot use completely new texture for performance reasons because it would require creating lots of new submeshes on fragments. However you can specify UV portion of the original texture inside ExploderOption script:

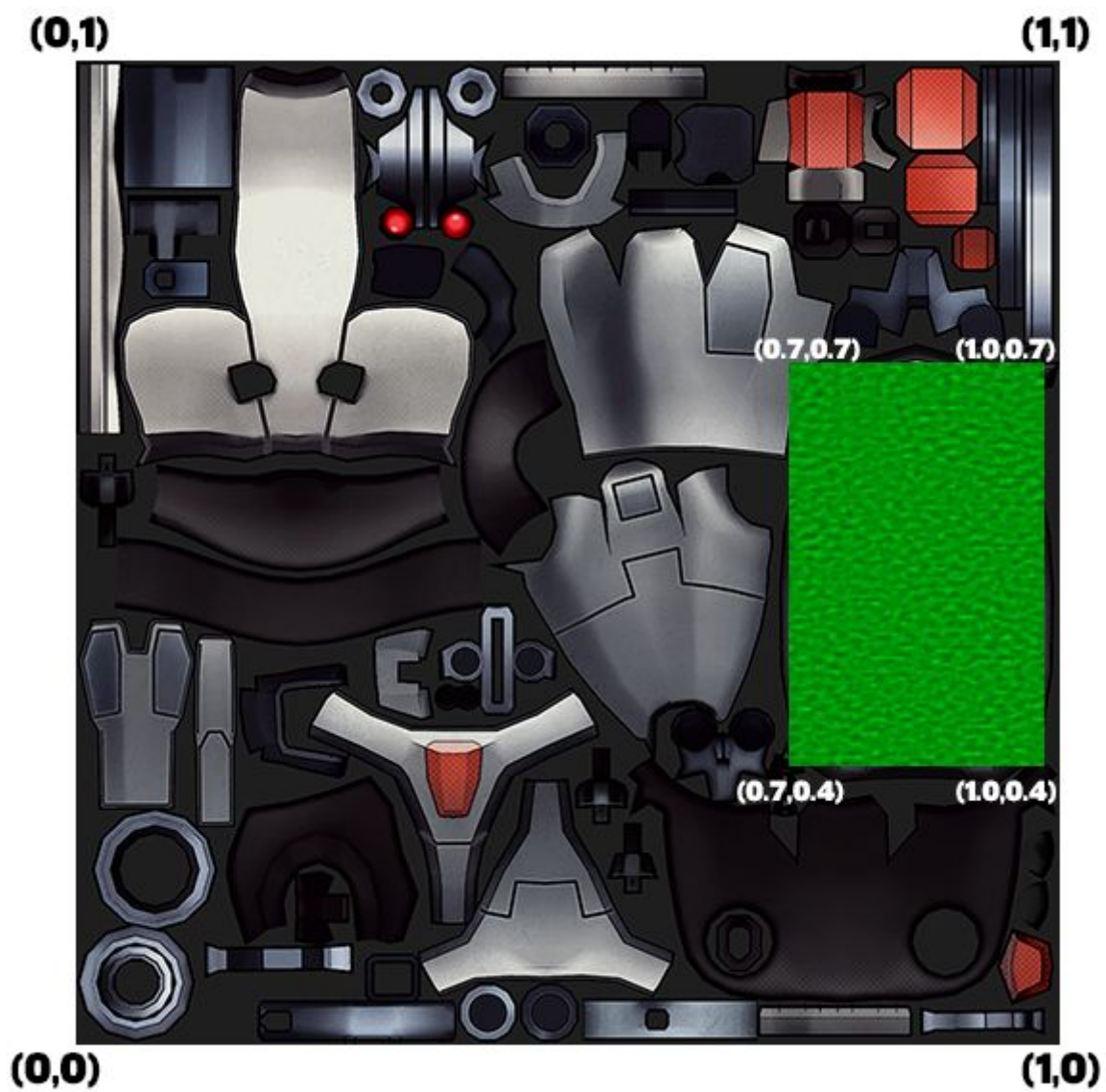


**Cross Section UV** is the place where you can specify the UV mapping for the inner side of fragments.

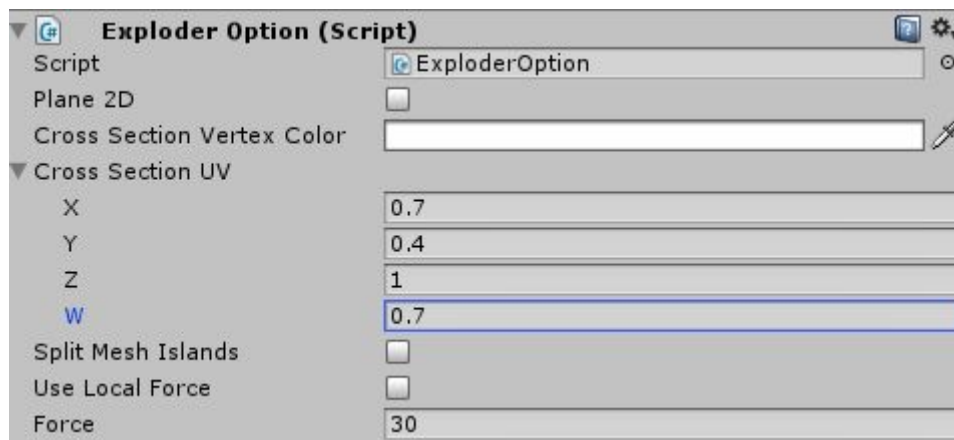
- X is a starting **U** coordinate (between 0..1)
- Y is a **V** coordinate (between 0..1)
- Z is a ending **U** coordinate (between 0..1)
- W is a ending **V** coordinate (between 0..1)

### Example with green blood alien:

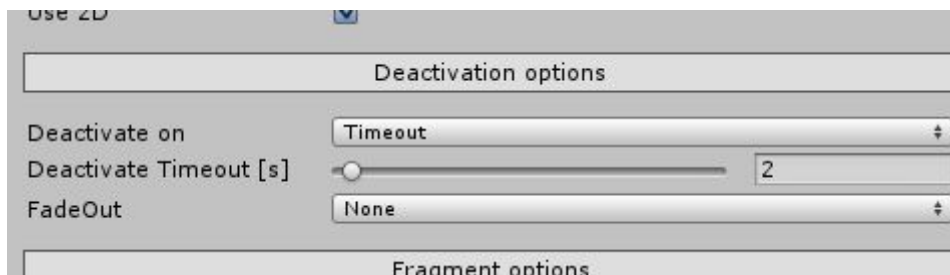
This texture is an example how the character texture can look like. As you can see the UV coordinates are in range between 0 and 1. The green rectangle represents the *alien blood*.



To use the green texture you need to specify the coordinates in ExploderOptions like this:



## 9 Deactivation options



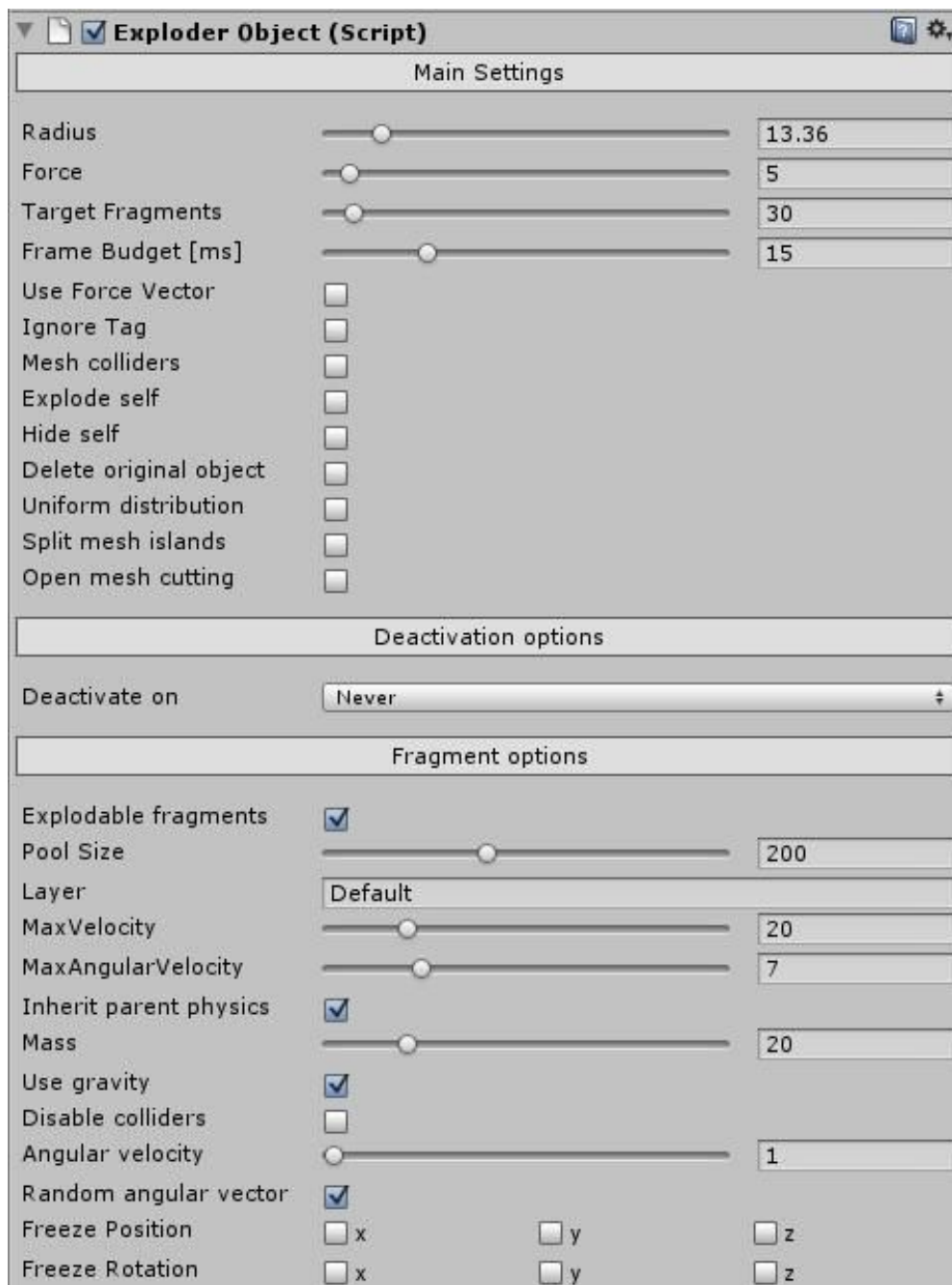
Deactivation options allows you to specify when the fragment can be deactivated.

- **Never** means fragment will be active all the time
- **OutsideOfCamera** means fragment will be deactivated once the main camera doesn't see it
- **Timeout** - means fragment will be deactivated after specified time [s] after explosion

In case of **Timeout** deactivation, you can also choose whether you want to fade-out the fragments:

- **None** - no fade-out option
  - **FadeOut Alpha** - fragments will smoothly change its material alpha to zero and they will become fully transparent  
**NOTE: this option will work only with transparent materials.**
  - **ScaleDown** - fragments will be smoothly scaled down to (0, 0, 0)
- 

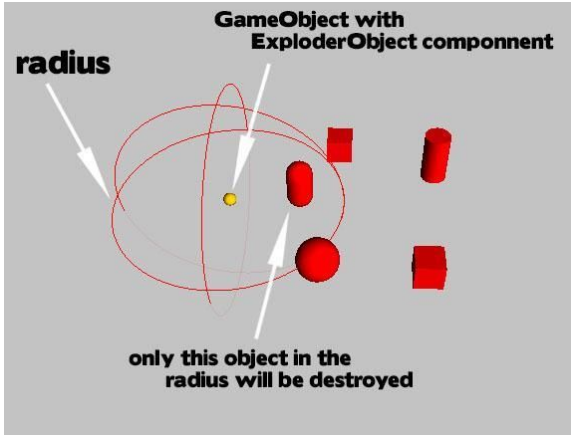
## 10 ExploderObject settings



## Main Settings

### Ignore Tag

Flag for not tagging Explodable objects. If you set this to TRUE you will have to assign "Explodable" script object to your GameObject instead of Tagging it. This is very useful if you already have tagged GameObject and you don't want to re-tag it to "Exploder".

<b>Radius</b>	<p>Radius of explosion is a radius range in which the Exploder will be looking for destroyable objects. Notice the Red wire-frame gizmo in scene view.</p> 
<b>Target Fragments</b>	<p>Number of fragments that will be created by cutting the exploding objects. More fragments means more calculation and more PhysX overhead and also better looking explosion.</p>
<b>Frame Budget</b>	<p>Time budget in [ms] for processing explosion calculation in one frame. If the calculation takes more time it is stopped and resumed in next frame. Recommended settings: 15 - 30 (30 frames-per-second takes approximately 33ms in one frame). However this settings depends on your game and average frames-per-second.</p>
<b>Force Vector (and Use Force Vector)</b>	<p>Note: this option is valid only if UseForceVector is true. ForceVector represents a 3d Vector direction in which the explosion fragments will be moving. For example with Vector(0, 0, 1) exploding fragments will fly in "UP" direction. In the Demo Shotgun weapon is using ForceVector in direction of the camera, so the fragments are always moving from the player.</p>
<b>Mesh Colliders</b>	<p>Using mesh colliders for fragments. Use this option wisely because mesh colliders are very slow and you should avoid them in most cases. Mesh colliders can be very useful if you want to run explosion with small or zero force.</p>
<b>Force</b>	<p>Force is how much the physical force is added to exploding fragments. More force means higher velocity.</p>
<b>Explode Self</b>	<p>Flag for destroying mesh in owner GameObject if there is any mesh component in it.</p>
<b>Disable radius</b>	<p>This option is valid only if ExplodeSelf is set to true. Disable radius scan for object, only the parent object will explode and nothing else.</p>
<b>Hide Self</b>	<p>Flag for hiding owner game object after explosion. This can be useful if you want to only hide and not to destroy owner object.</p>



<b>Destroy Original Object</b>	Flag for destroying original game object after explosion. For example when you destroy a “chair” model, it will cut the chair to small pieces and create new fragment game objects. If this flag is true it will call <code>Object.Destroy(chair)</code> and destroy the original chair <code>GameObject</code> from the game. Otherwise the original object (chair) will be only deactivated.
<b>Split mesh islands</b>	Option for separating not-connecting parts of the same mesh. If this option is enabled all exploding fragments are searched for not connecting parts of the same mesh and these parts are separated into new fragments.
<b>Uniform distribution</b>	By enabling this Exploder will create number of fragments per object equally regardless of object distance from the center. By default objects closer to the center (exploder center) will be shattered into more fragments than objects far from the center. Uniform distribution will guarantee that all objects will be shattered into same number of fragments.
<b>Open Mesh Cutting</b>	This option will allow you to use exploder with non-closed meshes, for example airplane with windows (mesh with holes). However it will not always triangulate the fragments properly. Use this only if your object has holes inside and/or it doesn't want to explode.
<b>Multi-Threading</b>	Exploder support multithreading, you select up to 3 additional threads for calculations. Threads are initialized on start but sleeping and are used only when needed.

### Fragment options

<b>Explode Fragments</b>	Flag for destroying already destroyed fragments. If this is true you can destroy the object and then all its fragment pieces. You can keep destroying fragments until they are small enough.
<b>Fragment Pool Size</b>	Maximum number of all available fragments. This number should be higher than Target Fragments.
<b>Max Velocity</b>	Maximum velocity for fragments, higher velocity will be clamped.
<b>Max Angular Velocity</b>	Maximum angular velocity for fragments, higher velocity will be clamped.
<b>Random Angular Vector</b>	This will randomize fragment rotation after explosion. If this item is false you will have specify axis of rotation manually.
<b>Angular Velocity</b>	Angular velocity for fragments, if “Inherit parent physics” is enabled, final angular velocity will calculated as a sum of parent and this value.

<b>Disable colliders</b>	Disable colliders from all fragments, use this if you don't want the fragments collide.
<b>Inherit parent physics</b>	By enabling this fragment will use same physics properties as its parent rigid body. It will inherit Mass, Velocity, Angular Velocity and Use Gravity. If there is no valid parent rigid body default settings will be used instead.
<b>Mass</b>	Mass of the fragment. if the parent object object has rigidbody and Inherit Parent Physics is true the mass property for fragments will be calculated based on this equation ( $\text{fragmentMass} = \text{parentMass} / \text{TargetFragments}$ )
<b>Layer</b>	Name of layer for fragments. Use this if you want to change default layer for fragments.
<b>Freeze position</b>	Freeze position of fragment in specified axis.
<b>Freeze rotation</b>	Freeze rotation of fragment in specified axis.
<b>Material</b>	Optional material for fragments, if not selected default material (first) from the original object is used

### SFX options

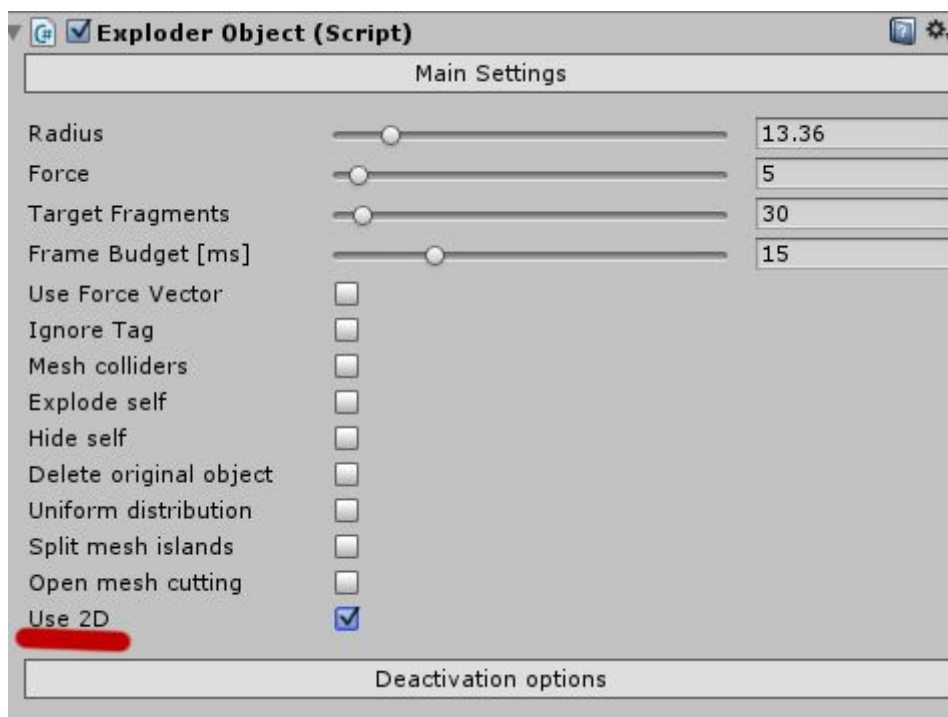
<b>Explosion sound</b>	Optional <a href="#">AudioClip</a> that will be played on the beginning of explosion.
<b>Fragment hit sound</b>	Optional <a href="#">AudioClip</a> that will be played when one of the fragment piece hit another object (collider).
<b>Hit sound timeout</b>	This option is valid only if Fragment hit sound AudioClip is set. Timeout between AudioClips, this is useful to prevent over-halting audio system when the explosion has lots of fragments.
<b>Fragment particles</b>	GameObject with particle emitters, that will be run when the explosion starts. Particles will be emitted from every fragment.
<b>Maximum emitters</b>	This option is valid only if Fragment particles is set. Maximum number of particle emitters, use low number if your game is experiencing lag.

## 11 2D Physics (Unity4.3+)

*Tested with 2D Toolkit.*

Exploder support 2D physics as well. You can enable this feature by setting Use2Dcollision (Use 2D) to True. All explosion fragments will have 2D colliders (PolygonCollider2D).

It is important that you will not use this settings with exploding 3D objects (it might work but it might lead to unexpected behavior).



<http://youtu.be/lZWz5mt2kIY>

### IMPORTANT NOTE:

Exploder DOES NOT work with Unity built-in sprites, because there is no way to access sprite mesh data required by Exploder.

---

## 12 UFPS integration

Exploder has been successfully tested with **Ultimate FPS camera 1.7**, all you need to do is to add 2 lines of code inside UFPS script.



1. Add Exploder prefab to the scene
2. Open file vp\_Bullet.cs in UFPS folder.
3. Add these two lines into method **DoHit()**

```
var exploder = Exploder.Utils.ExploderSingleton.ExploderInstance;  
exploder.ExplodeObject(m_Hit.collider.gameObject);
```

4. Done! You can now fine tune Exploder settings and see explosions in the game.

**Tip:**  
If you find it difficult to explode some of the UFPS crates, enable "OpenMeshCutting" in Exploder settings.

---

## 13 Playmaker support

There are 5 Exploder custom actions ready to work with Playmaker, you can find them in action browser in section "Effects".

- Explode Object
- Explode Radius
- Crack Object
- Crack Radius
- Explode Cracked

Steps to enable Playmaker:

1. Add **Exploder prefab** to your scene first.
2. Open action files located in Assets/Exploder/Playmaker/\* and uncomment the second line (//#define PLAYMAKER)
3. Select the target object you want to explode and add one of the Exploder action to it

---

## 14 FAQ

**Q:** I cannot explode my objects. Sometimes it creates only one big fragment.

**A:** Make sure your objects are in exploder radius, properly tagged and the TargetFragments value is high enough. If it still does not work you can try to enable “Open mesh cutting” which is an option for exploding non-closed meshes. This option makes Exploder more tolerant to complex meshes with holes. If it still does not work, please contact me by e-mail with screenshot of your object hierarchy and your model in scene.

**Q:** Explosion seems to be huge even if I set small Force value.

**A:** It is because all fragments have by default box colliders and they are tight together overlapping each other, which might cause high explosion regardless of Force value. You can easily fix it by enabling **Mesh colliders** and/or limiting velocity with **MaxVelocity** parameter.

**Q:** FadeOut Alpha does not seem to work. Fragments disappear in one frame (blink).

**A:** FadeOut Alpha works only with transparent materials that allow to change alpha value.

**Q:** Can I specify custom texture for fragments?

**A:** You can specify UV portion from parent texture, see more details in this [chapter](#).

**Q:** Does Exploder support reverse explosion?

**A:** Unfortunately not, Exploder can only explode objects in usual way.

**Q:** Can Exploder destroy Unity sprite?

**A:** Unity built-in sprites are not supported by Exploder because there are no mesh data. You can use Exploder 2D product to enable this functionality.

**Q:** I don't want to change the tag to "Exploder", it already has one.

**A:** You don't have to, just make sure to enable "Ignore Tag" and add "Explodable" component to your game object.

**Q:** Can Exploder destroy only a portion of the object, like a head or arm of a character?

**A:** Unfortunately it can't, Exploder works only with a whole mesh, in case of a character it will explode head, arm, body, legs and everything else at once.

**Q:** Does Exploder support submeshes?

**A:** It does, but in limited way. It can explode mesh with submeshes but resulting fragments will have only one mesh. It means that all fragments will have only one material.

**Q:** How to access fragment pool and get active fragments?

**A:** Please refer to file ***HowToGetActiveFragments.cs***.

**Q:** Can I use Exploder to create fragment prefabs?

**A:** No, Exploder is only for real-time use. However you can use Crack() feature to pre-calculate explosion and execute it later.

**Q:** Any tips for increasing performance?

**A:** First of all **TargetFragments**, try to keep it low, less fragments mean less overhead and less calculations. Also make sure that **FrameBudget** is lower than half of your expected frame rate, for example game with 30 FPS should have **FrameBudget** 15 and less, otherwise you can notice lagging. Try to **avoid Mesh colliders**, they can be very expensive. Also don't use **SplitMeshIslands** so much, it can save you few more frames. Don't forget that performance is always about tradeoff, good luck with your game!

---

## 15 Support

If you have any questions or need a help with setting up the Exploder game object you can write to unity forum:

<http://forum.unity3d.com/threads/190198-Exploder-RELEASED>

or you can send me an e-mail to:

[gamesreindeer@gmail.com](mailto:gamesreindeer@gmail.com)