Zone Controller Users Guide

Thank you for purchasing Zone Controller!

I sincerely hope that you find it to be a useful tool in your Unity development arsenal. Since Zone Controller is a generic tool for controlling components in Unity, you will likely find many amazing and creative uses for it that go far beyond what I could have ever imagined.

So, What is Zone Controller?

If you are reading this document, you have likely already purchased Zone Controller, and watched some of the YouTube videos, which demonstrate it. However, I want take a step back and get to the core of Zone Controller and explain why it was made and what is really does.

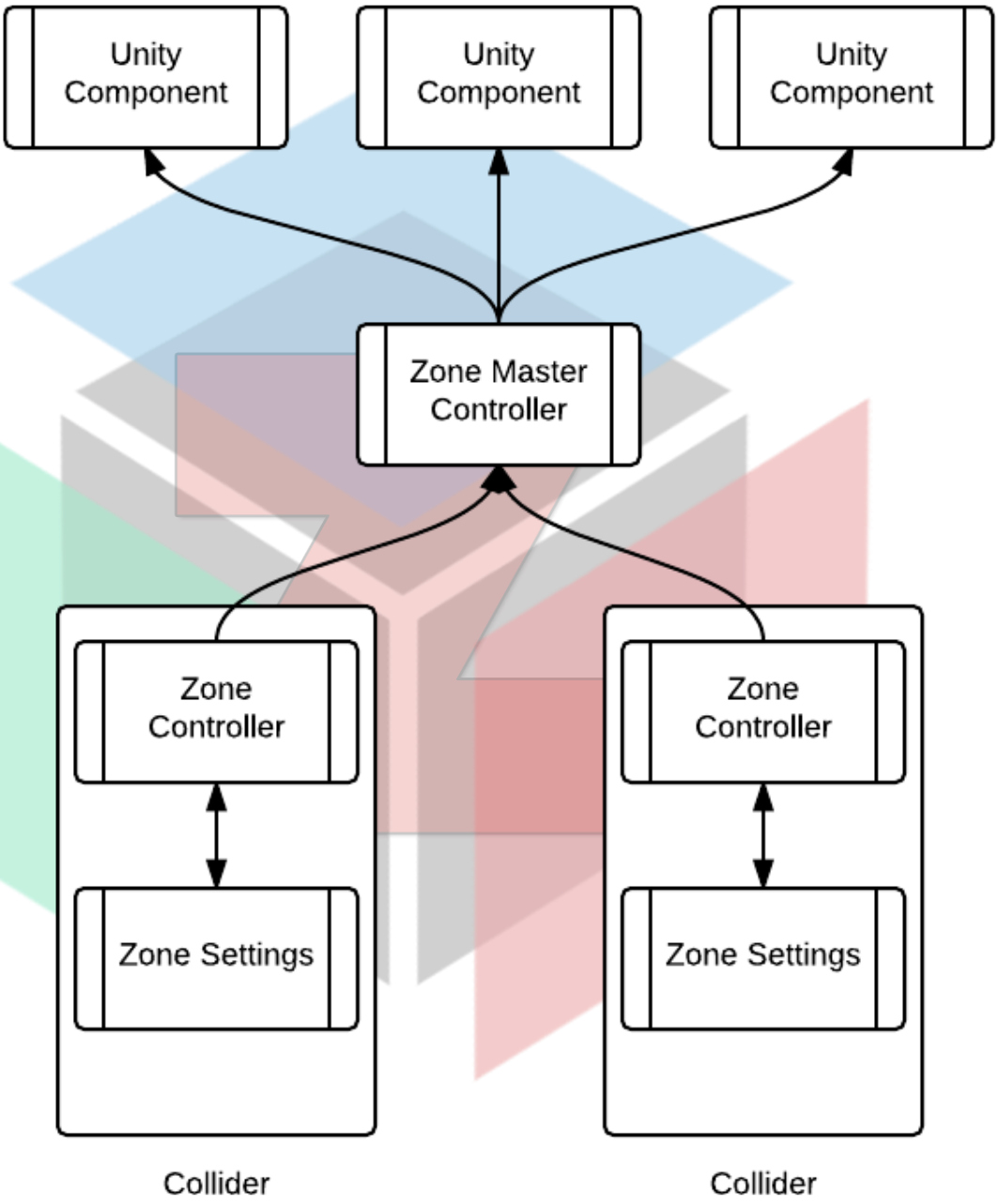
We have all had situations in Unity where we create a transparent cube around an area of our map and slap a trigger collider on it to cause an event like changing the sound track, or opening a door to happen. While these situations make for a quick and dirty solution, I had a vision for a more generic approach for this design pattern. What if I could create a trigger collider zone that could control **anything**? And after some initial prototyping to determine the viability of the solution, Zone Controller was born.

At its core, Zone Controller is a very clever remote control system for Unity Components, which is triggered (most of the time) by collisions in Unity. When the defined object (main camera or developer selected collider) enters a Zone, a message is sent up to the “Zone Master Controller” script along with the details of what it needs to be controlling while the collision is still active.

You can see the basic architecture of how Zone Controller works in the diagram below. At the center of the solution is a single instance of our “Zone Master Controller”. Its the nerve center of the solution, and responsible for taking the settings specified in a Zone and smoothly applying them to the various Unity components at Runtime.

The “Zone Controller” is responsible for maintaining the status of if the Zone should be currently active or not. It also contains an instance of the “Zone Settings”. The “Zone Settings” are a collection of individual property settings you want to control when the Zone is activated. You can have as many Zones setup in your scene, as you like.

When you enter a Zone, the “Zone Master Controller” will also ensure that the transitions from one value to the next are a smooth and gradual shift as to avoid hard and abrupt changes.



Common Uses for Zone Controller

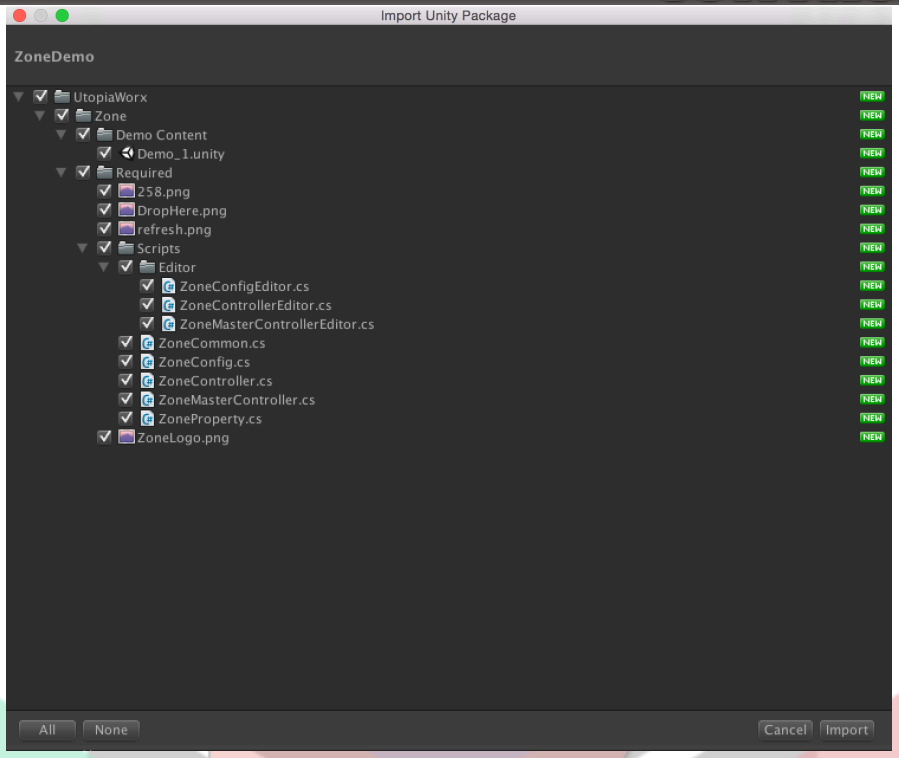
Zone Controller can be used in many ways in your development workflow, however here I'm going to list out a few common scenarios in which Zone Controller Pto has proven to be very effective.

- **Post Processing Effects** – control the parameters of your post processing effects at runtime or enable certain special effects when you enter a specified zone.
- **Control Core Unity settings** – Zone Controller can control core Unity settings like Timescale to create slow-motion effects, or Global Gravity to create low-gravity regions of your map.
- **AI** – Make your enemies more aggressive when you are in a danger area
- **Day/Night Cycle** – Force the control of your day/night manager to change when you enter a certain zone
- **Controlling Sound** – Zone Controller is very effective at controlling conditional and situational sound scenes and can change music, apply sound effects and much more.
- **Custom Scripts –** Zone Controller can control any script you have that inherits from the "MonoBehaviour" class.

Installing Zone Controller

If this is your first time to use Zone Controller, you will find that the installation is both quick and painless. Once you have purchased the package, and downloaded it to your local copy of Unity, you will want to Import it into your project.

When prompted with the import dialog, you will have the option to install the core system as well as the demo content. While I recommend that you go ahead and install the demo content so you can see examples of Zone configurations, it's completely optional. However, you must install the core system files for zone under the "Required" folder.



Once the import is completed, you will have successfully installed the Zone Controller package. The next thing you will want to do is look at how to configure it in your scene.

Configuring Zone Controller

Now that you have imported Zone Controller into your Scene, you can begin the very simple process of configuring your scene to use the Zone system. The steps are as follows:

1. Click “Game Object->3D Object->Zone-New Zone”
2. Congratulations! You have configured the basic Zone Controller installation.

Seriously though, I made the installation and configuration very simple. When you create your first Zone, the code that generates the Zone itself will setup the environment for you so all you have to do is select the properties you want to control.

By performing the step above you will have added 2 elements to your Scene:

- A Zone Master Controller
- A Zone Controller complete with collider and Zone Configuration

Let’s examine the differences between the two.

The Zone Master Controller is the component, which is responsible for running the show. It's the nerve center of the whole operation and as thus a required component. The Zone Master Controller is what is responsible to maintaining state of the whole system as well as causing the transitions from one Zone to another to work. The UI for the Zone Master Controller is a bit limited in that there are not a whole lot of configuration options to set for it.

As you can see in the image above, this Zone Master Controller holds a reference to all of the Zones in the Scene and gives you quick access to change the transition times for those Zones. If you click on the button to the right of each None name (the one with the "?" in it), Zone will locate that Zone Controller in the Unity Editor for you and pan the Editor Camera to focus it in the Scene view.

Please note that the Editor UI for this as well as all Zone components become locked at runtime, so you can only make changes to the UI when in edit mode.

A **Zone Controller** is an instance of an individual Zone and contains all settings and configurations required to communicate to the Zone Master Controller when the Zone becomes activated. The Zone Controller does very little at runtime other than contain a list of settings, which will be applied by the Zone Master controller.

The Zone Controller contains a few very high level settings, which help establish how each individual Zone will operate at runtime.

Above you can see the settings of the Zone Controller itself. It contains a reference back to the Master Zone Controller and configurations for the Trigger Mode.

Presently there are only 3 Trigger Modes available in Zone:

- **Main Camera** – triggers when the main camera enters the zone
- **Pick Collider** – where you can specify a collider in your scene that will trigger the Zone
- **HotKey** – where you can specify keys on your keyboard to activate and deactivate your zones

The next piece of the Puzzle is the **Zone Config** component. The Zone Config is where you specify all of the nitty-gritty details of what your specific Zone will do. This is where you will register your Game Object, and set the values of what you want your Properties to be. The Zone Config allows you to register as many properties you would like to control and specify how the transition should occur when the Zone is triggered.

When you start to configure your Zone, you will select Game Objects, which contain the components you are interested in controlling. Once you add a new Game Object to the list, Zone will scan that new item and show you a list of fields, properties and methods available to be controlled.

In this picture you see the main configuration settings of the given Zone.

- **Transition Time** – specifies how quickly the Zone will take effect. Lower values will cause the transition to go faster and Higher values will cause the transition to be slower.
- **Bool/Texture Cutoff** – this spedicied when in the transition to change Boolean, Texture, Texture2D and Audio Clip values
- **Transition Curve** - allows you to specify a gradual curve to the transition effects
- **Use Fixed Update** – will allow you to switch between using Update and FixedUpdate on the Zone Master Controller. This will effect the quality of the smoothness of the transition
- **Uses Allowed** – lets you specify how many times this zone can be triggered. This is a great option when you only want Zones to be used a handful of times. (0 will cause infinite uses)
- **Zone Timeout** – this is a number of seconds the Zone will stay active while the user is inside of it. If you have a specific effect like falling into water, you may only want the visual effect to last a few moments. (0 will cause infinite timeout)

What all can I control with Zone?

Zone Controller was written to take a very generic approach to controlling things. Specifically it makes heavy use of .NET Reflection and Generics to discover information about the things you choose. Zone Controller will read and expose he metadata about any Unity Components, which derive from "MonoBehaviour".

Zone Controller will inspect each component and look for all Felds, Properties and Methods that match the filters you specify. You can enable both Public and Private as well as Static and Instance based. Presently Zone Controller only supports and exposes the following Data Types:

- String
- Boolean
- Integer
- Float
- Texture (Pro Only)
- Texture2D (Pro Only)
- Material (Pro Only)
- AudioClip (Pro Only)
- Integer Based Enumerations (Pro Only)
- Color
- Vector2, Vector3 and Vector4

Since the core Components provided in Unity itself do not technically derive from "MonoBehaviour" they have been treated special and as of the version, only the following Unity Core Components are supported:

- *Lights*
- *Camera*
- *AudioSource*
- *MeshRenderer*
- *ParticleSystem (Limited Support)*
- *Audio Filters (HPF,LPF,Chorus,Reverb,Distortion,Echo)*
- *RigidBodies (2D & 3D)*

In addition to reflecting on the field/property metadata of a component, Zone Controller will also expose Methods that component may have available, which you can use to queue up Dynamic Method Invocation to the target class upon entry and exit of the Zone. Zone Controller will assist you in creating the variables you need to match the Method Signature for these calls

Outside of these limitations, Zone Controller will work with any 3rd party Unity component you can throw at it. It can be written in C# or Unity Script. It can be a Post Processing Effect, or Complex Weather/AI system. Zone Controller will inspect it, analyze it and tell you which pieces you can control at runtime.

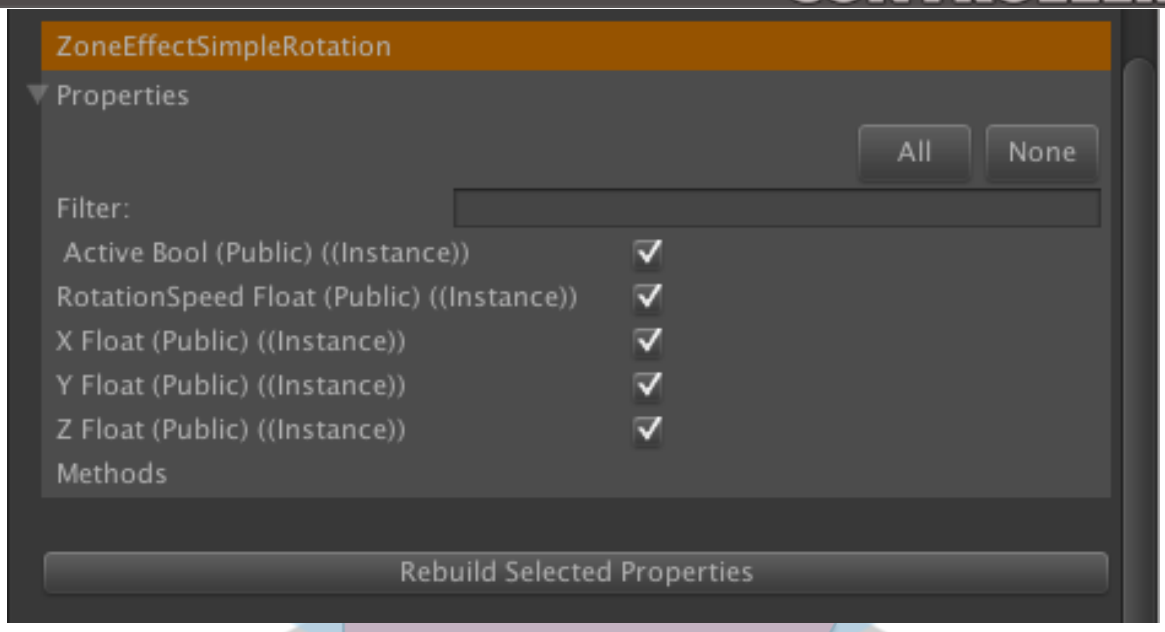
Lets add some properties to our Zone.

The first thing we will want to do is to add a new slot for a Game Object we wish to include in this Zone. Under the section labeled “Object to Control”, you will see a button named “Add Item”. Clicking this will add a new slot in the collection of controlled Game Objects in this Zone. You can add as many Game Objects to a Zone as you like. Once you have added an empty slot you can either drag a Game Object from the hierarchy into the empty slot, or click on the explore button next to the slot to Browse your Scene for the desired one.

No matter how you select the GameObject, Zone Controller will immediately begin the process of introspection on that Game Object, and provide you with a list of Components that the Zone can control. In addition to that, it will also provide you with a list of fields, properties and methods for each of those components.

As you can see in the example above, I have specified that I want to control a Directional Light, and Zone Controller has discovered the properties that can be added to the Zone, and I have specified that I want to control the Intensity of this light when my Zone is triggered.

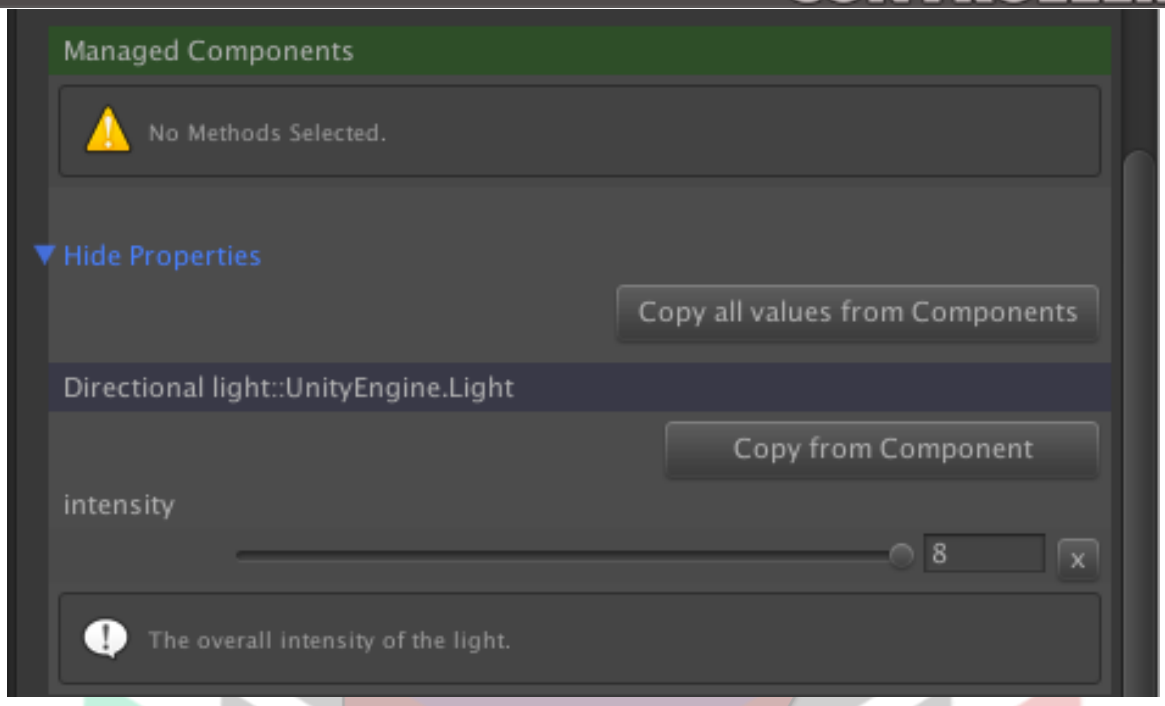
You can also see that I have set my filters to include Public and Private fields as well as Instance and Static fields.



Once you have selected all of the components you wish to control, you will then click the “Rebuild Selected Properties” button. This will load them into the list of properties that are actually controlled by this Zone.

Note that there is an additional filter in each component view card, which will allow you to narrow down the list to be based on the phrase you type.

Once you have selected the properties that you want to control, you can now set the values that you want them to transition to when the Zone is entered.



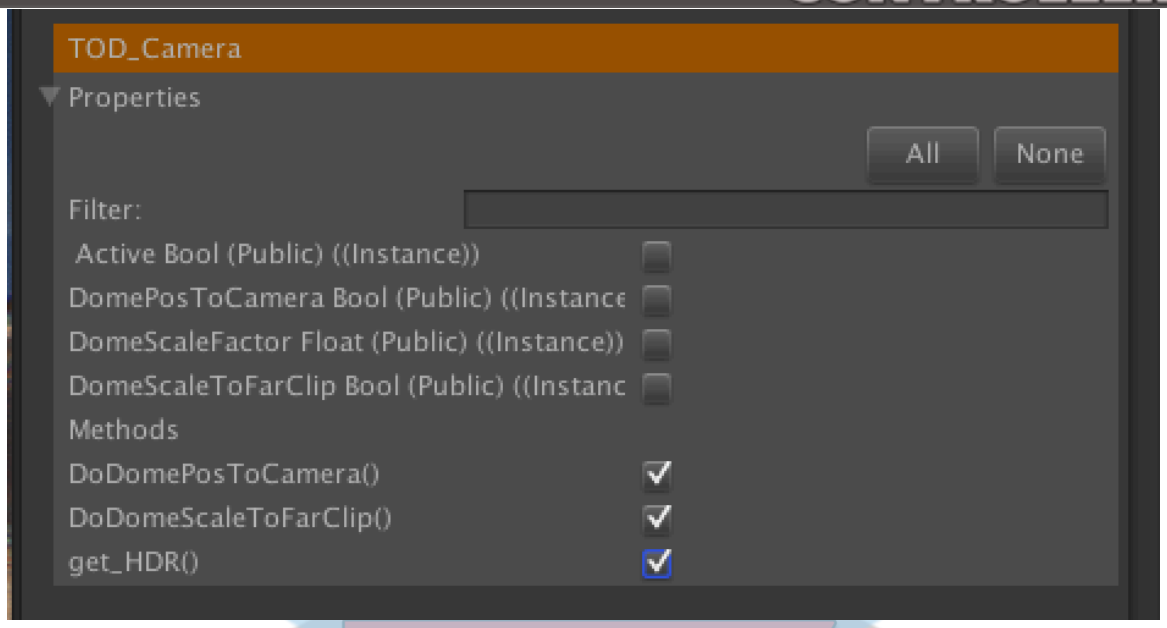
In this Image you can see how Zone Controller allows you to specify what values you wish the selected properties to be set to at runtime.

Pro Tip: You can click on the Button that says, "Copy all values from Components" and Zone Controller will capture all of the existing values from the component and set them in the Zone versions. This is a very useful feature when working with visual effects and you can tweak the settings to be exactly how you like them in the game preview window. Just set them how you want them, click the copy values button to set the Zone versions and then dial them back on the Game Object itself.

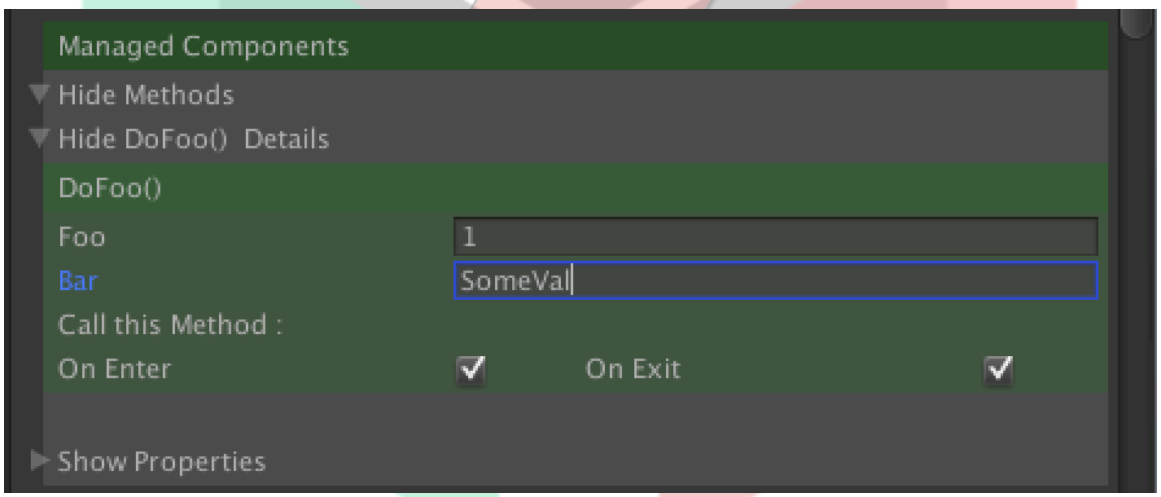
Calling Methods

One of the more exciting features built into Zone Controller is the ability to call Methods on other Components when Zones are entered and exited. This form of Method invocation differs slightly from the traditional "Broadcast Message" way of cross component communication done in many other tools. These invocations are done directly on the instance of the component itself and not broadcast to a Game Object, which then tries to relay the message to all child components.

Similar to how you select the fields and properties you want to register in your Zone, you also have the ability to select Methods.



Once you select the Methods you would like to use, click the same “Rebuild Selected Properties” button as you did before and then take a look at the Managed Components cards again.



You will notice that Zone Controller has exposed every parameter required to call this Method and provides you with a UI to specify the values you would like to pass into the Method. Additionally you can specify when you would like to fire the method by clicking on either “On Enter” of “On Exit”.

Now, we have essentially told the Zone Controller that we want to control the various properties and methods from all of the collected components contained within all of the selected GameObjects on it. We set the values of those

properties and defined our runtime values to pass into any Methods we want to use. Now when the game runs, and our collider enters the defined Zone, a message will be sent to the Zone Master Controller to notify it that a zone has been activated and to start applying the settings from that Zone and if any methods were specified, they will fire off one time on enter and exit.

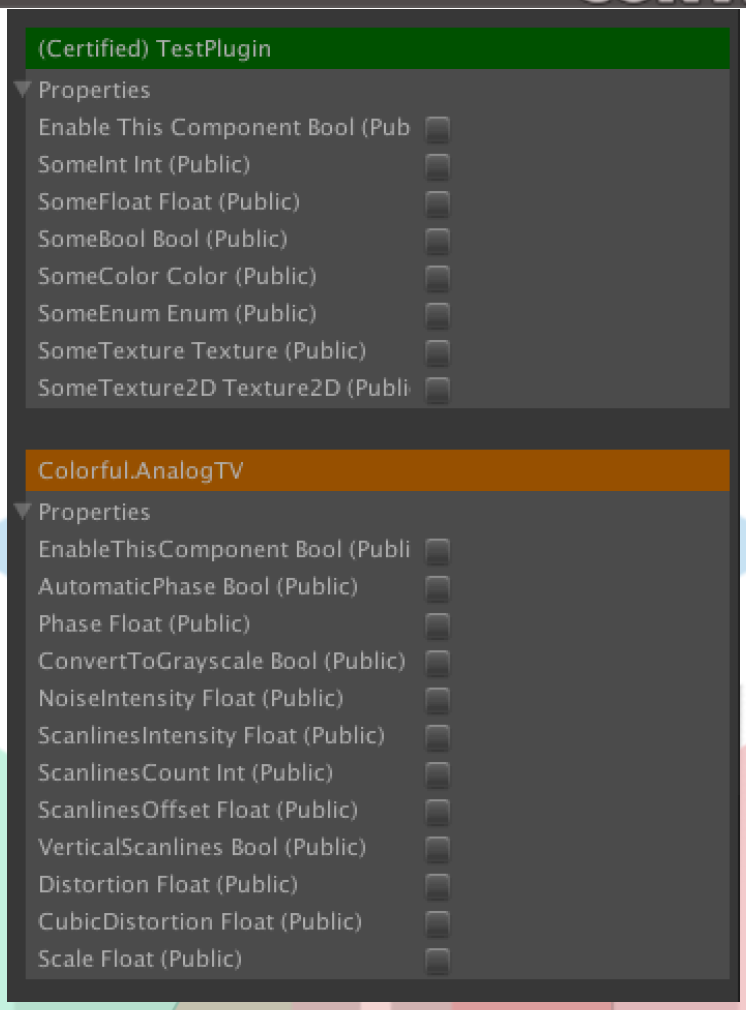
Once the Zone is entered, and the Zone Master Controller is in charge, it will slowly and gradually transition all of the values you selected in order to create a seamless transition into that Zone. You will also notice that as you leave that Zone, the previous values assigned to those properties has been remembered and the Zone Master Controller will slowly and gradually fade you back.

Certified Plugins

While Zone Controller is written in such a way that it will work with virtually any Unity component, you must realize that Zone Controller does not know the internal details of how those plugins work. When you select a GameObject and Zone Controller searches its internal classes for fields, properties and methods, it is not aware of any meanings to these properties. There are situations where you are trying to change a property that is also trying to be changed by the plugin itself.

To combat this, I have made an API that other Unity component developers can use to ensure that their code will work in perfect union with Zone Controller. You will notice the difference in Certified Plugins and Non Certified ones because the Certified Plugins will appear Green and have the prefix of “(Certified)”

Certified plugins have been integrated and tested by the developers who made them.



Integrating Zone Controller

If you are a component developer and are interested in learning how you can integrate your component into Zone Controller, I have made a simple but yet powerful API you can use to expose your settings to Zone Controller.

Everything you need to know about how to integrate your component is in the Demo Content folder under the “Test_Plugin” directory. There you will find an example C# class with a full integration into Zone Controller. It’s also full of comments and information about how Zone Controller will work with your plugin.

Version Comparison

Zone Controller is offered in two versions Zone Controller Pro and Zone Controller Free. Below is a chart that displays the comparison data between the two, which can help you decide what version is best for you:

Feature	Free	Pro	Category
Runtime Engine	✔	✔	Core
AudioClip	✖	✔	Data Type
Boolean	✔	✔	Data Type
Color	✔	✔	Data Type
Enumerated Fields	✖	✔	Data Type
Float	✔	✔	Data Type
Int32	✔	✔	Data Type
Materials	✖	✔	Data Type
String	✔	✔	Data Type
Texture	✖	✔	Data Type
Texture 2D	✖	✔	Data Type
Vector2	✔	✔	Data Type
Vector3	✔	✔	Data Type
Vector4	✖	✔	Data Type
Beta Access	✖	✔	Extra
Bonus Scripts	✖	✔	Extra
Documentation	✔	✔	Extra
Source Code	✖	✔	Extra
Certified Components	✔	✔	Integration
Reflected Components	✔	✔	Integration
Unity Core Components	✔	✔	Integration
2D collider support	✖	✔	Runtime
Fine-tune Rounding	✖	✔	Runtime
Method Invocation	✖	✔	Runtime
Copy Values from Components	✔	✔	UI
Curved based transitions	✖	✔	UI
Filter Visibility	✔	✔	UI
Fixed Update or Update	✖	✔	UI
Hot Key based Zones	✖	✔	UI
Multi Game Objects Per Zone	✖	✔	UI
Zone firing limits	✖	✔	UI
Zone Timeouts	✖	✔	UI
Unlimited Zones Per Scene	✔	✔	Editor
Unlimited Components Per Zone	✔	✔	Editor
Unlimited Properties Per Zone	✖	✔	Editor

Support

If you find yourself in need of support, please feel free to contact me directly at:
john@smarterphonelabs.com

Or

Visit the Unity Forums page here:
http://forum.unity3d.com/threads/zone-controller-pro.380426/

Thanks again for purchasing Zone Controller,

Enjoy!

Version History:

New features In Beta 2

In this build, I have added support for some new concepts and components as detailed below:

- Added support for some Core Unity types (Lights, Cameras, AudioSources)
- Added the ability to bind a Zone to more than one GameObject
- Made some major performance improvements to the Core Lerp engine
- Added support for String and AudioClip
- Added some example Scripts to demonstrate Zone (Time Warp, Gravity)
- Changed from a Single Transition Time setting on the Master Controller to individual Transition times per Zone
- Major UI improvements
- The ability to search/filter selected properties
- The ability to filter based on Public/Private/Static/Instance
- Fixed an issue where the Master Controller would forget some settings on exit.
- Support for descriptions from Certified plugins
- Worked with more asset publishers to start integrating their plugins.

New Features in Beta 3

In this build I added support for some more methods of controlling the Zones as well as a major update to allow for Shaders and Materials:

- Added Undo support to the UI for most properties
- Added runtime support for 2D colliders
- Added Runtime Method Invocation so you can call methods when you enter and exit a zone
- Added Support for Javascript based components to work at design time
- Added API change to allow registration of Certified Methods in addition to properties
- Added an optional limit to how many times a Zone can trigger
- Added an optional Time limit to allow Zone to be active only a certain timespan
- Added support for Unity Materials class and MeshRenderer objects
- Added a Master Disable for Components
- Added support for Hot Key driven Zones
- Integrated some Menu Items for making new zones from the GameObject Window (GameObject->3D Object->Zone->New Zone)
- Auto Collapse the foldouts for Components, settings, objects, filters and Methods to keep screen real estate usage down.
- Added Animation Curve to control the speed at which the changes are applied at runtime (with default linear curve)
- Locked all inspector windows to not allow property changes in Zone editors at runtime
- Added support for Particle Systems

New Features in Beta 4

Beta 4 is meant to be the final beta issued before releasing on the Asset Store. It will mostly contain bug fixes and performance enhancements to the existing system, however a few small features have been added:

- Added support for Vector2, Vector 3 and Vector4 data types
- Added rounding control to the Zone Master Controller
- Split up the functionality in Free and Pro versions
- Added the ability to change where in the transition process Boolean/texture materials change over

Planed Enhancements

These are a list of enhancements I intend to add to Zone at some point in the near future, though each of them has an unknown ETA at this point:

- Add support for Conditional zone triggering where you can specify certain fields must have certain properties for Zones to trigger.
- Add support for nested Zones where the Master controller will average the properties of many zones in the stack before applying changes.