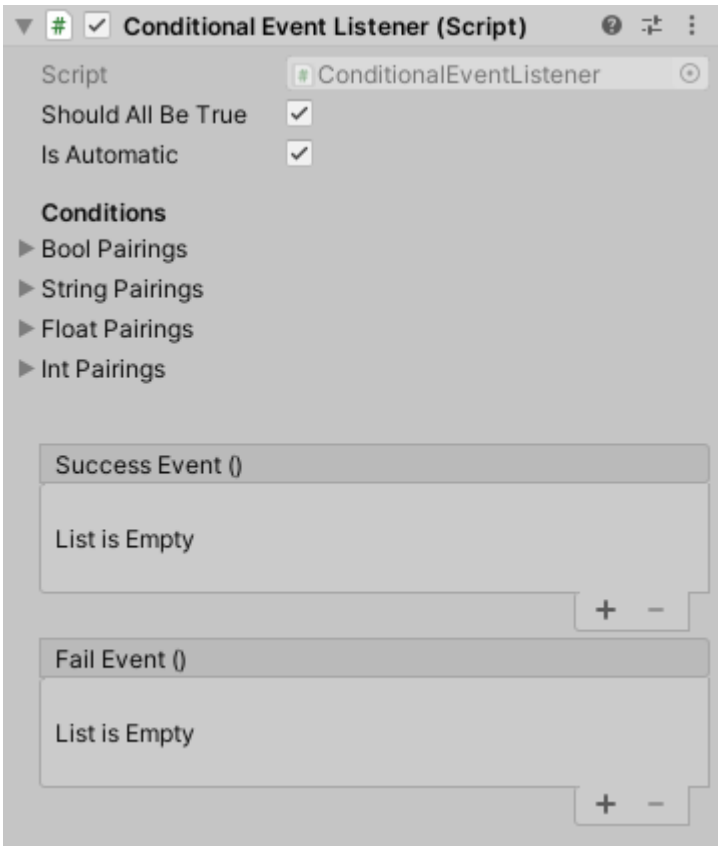
Coldiron Tools Documentation

A number of tools to allow designers to have more freedom and control over the game. The primary goal is to allow for simple integration of the Scriptable Object architecture in Unity, which allows designers to avoid dependencies and contrasts the Singleton architecture pattern. Scriptable Object architecture uses SOs as simple variables which can be assigned and evaluated in the editor without needed to reference the source of the change. In addition to the SO architecture pattern, there are a number of tools to extend the functionality of Unity's UI and a handful of other helpful utilities.

Events

Conditional Event Listener

Allows designers to call Unity Events based on the evaluated scriptable variables.



Properties

Property:	Function:

Should All Be True	Determines if every paired condition should be true, or if the condition should succeed with one condition
Is Automatic	Determines if this should evaluate conditions every time a paired value changes, or should manually evaluate conditions with **AttemptInvoke()**
Bool Pairings	A list of **Bool Pairings** to be evaluated. If the referenced **Bool Scriptable** matches the **Desired Value** then the pairing evaluates to true.
Strings Pairings	A list of **String Pairings** to be evaluated. If the referenced **String Scriptable** matches the **Desired Value** then the pairing evaluates to true. **Is Case Sensitive** determines whether the **Desired Value** and **Referenced Value** will be set to lower case before being evaluated.
Float Pairings	**A list of Float Pairings** to be evaluated. Uses the **Compared Operator** to evaluate the condition, for example (**Desired Value >= Referenced Value**). If **Compared Operator** is set to **Is Between**, then **Desired Value** is replaced with **Desired Min Value** and **Desired Max Value**, the **Referenced Value** must be between, and not equal to either of these values.
Int Pairings	**A list of Int Pairings** to be evaluated. Uses the **Compared Operator** to evaluate the condition, for example (**Desired Value >= Referenced Value**). If **Compared Operator** is set to **Is Between**, then **Desired Value** is replaced with **Desired Min Value** and **Desired Max Value**, the **Referenced Value** must be between, and not equal to either of these values.

Events

Property:	**Function:**
Success Event	A Unity Event that is invoked when the conditions are evaluated as true.
Fail Event	A Unity Event that is invoked when the conditions are evaluated as false.

Public Methods

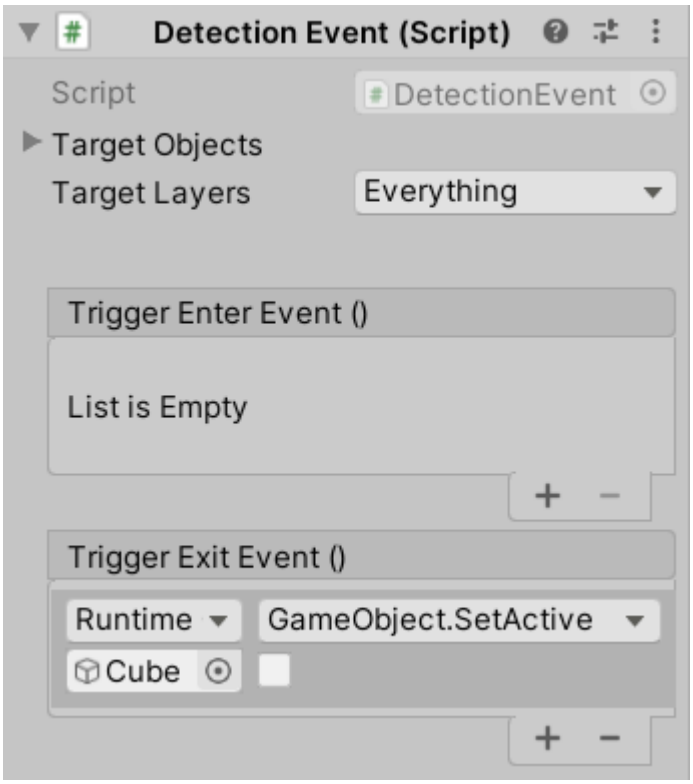
Attempt Invoke	Begins evaluating conditions. Will invoke the **Success Event**, or **Fail Event** depending on the result. Called automatically when any **Referenced Value** is modified unless **Is Automatic** is set to false.

Details

Conditional Event Listeners are flexible tools that can be used for many common actions, such as locking doors or turning lights on and off.

Detection Event

Allows designers to quickly set up collision events.



Properties

Property:	**Function:**
Target Objects	A list of **Game Objects** to check for collision. If the colliding object is contained in that list, it will invoke the corresponding event.
Target Layers	The layers that get checked for collision.

Events

Property:	**Function:**
Trigger Enter Event	A Unity Event that is invoked whenever a **Game Object** (with a **Collider**) enters this object's **Collider**, if that **Game Object** is either in the list of **Target Objects** or on the **Target Layers**.
Trigger Exit Event	A Unity Event that is invoked whenever a **Game Object** (with a **Collider**) exits this object's **Collider**, if that **Game Object** is either in the list of **Target Objects** or on the **Target Layers**.

Public Methods

Add Target Object	Adds a **Game Object** to the **Target Objects** list.

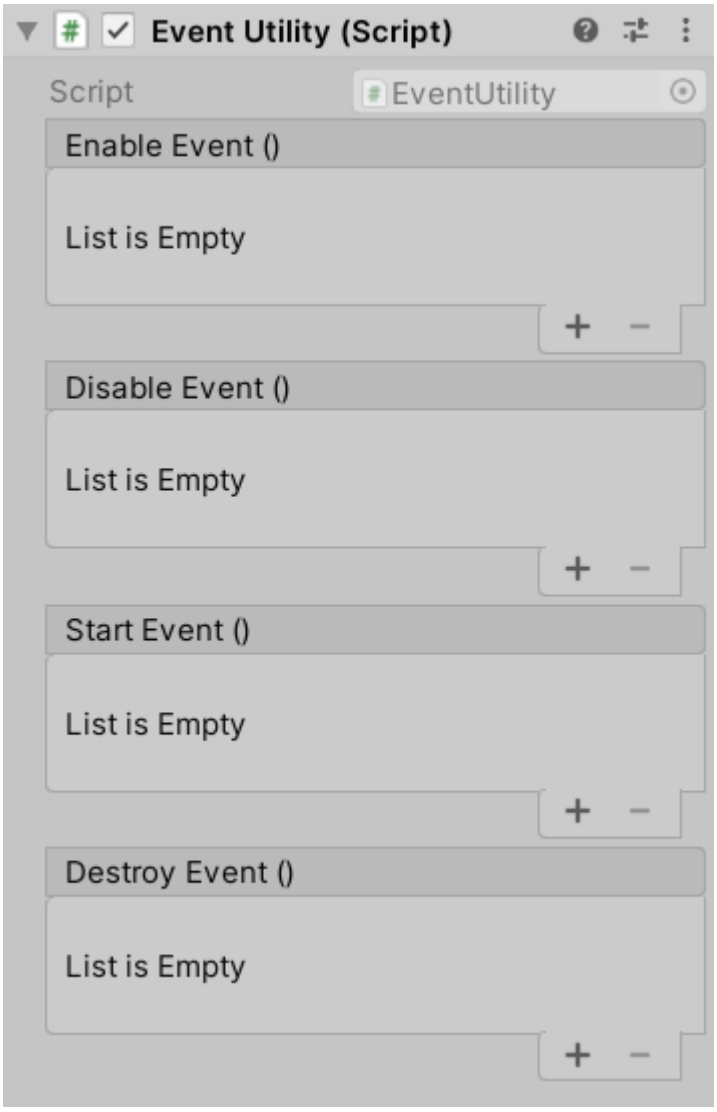
Remove Target Object	Removes a **Game Object** from the **Target Objects** list.

Details

Detection events are designed to allow designers to quickly set up trigger events without having to directly code them. Useful for things like triggering cinematics, or damaging the player.

Event Utility

Allows designers to invoke Unity Events during certain useful times.



Events

Property:	**Function:**

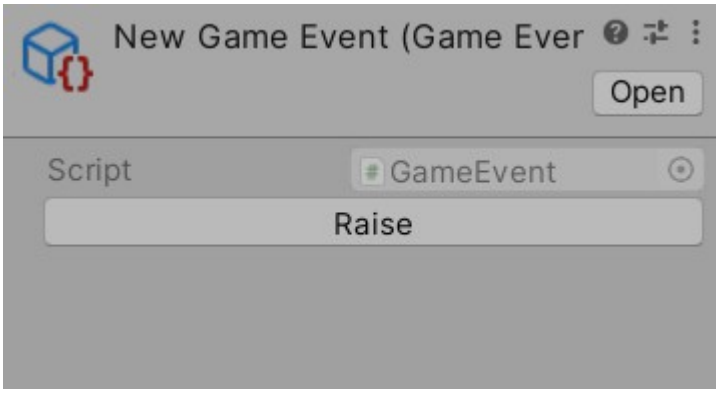
Enable Event	A Unity Event that is invoked whenever the **Game Object** is activated.
Disable Event	A Unity Event that is invoked whenever the **Game Object** is deactivated.
Start Event	A Unity Event that is invoked whenever the game starts.
Destroy Event	A Unity Event that is invoked whenever the **Game Object** is destroyed.

Details

Event Utilities are helpful extensions for when simple actions need to be performed on Start or when the game object is activated that aren't worth manually writing in code.

Game Event

Allows designers and developers to broadcast events without needing to manually assign references in code.



Public Methods

Property:	**Function:**
Raise	Invokes the Unity Event on all registered **Game Event Listeners**.
Register Listener	Registers a **Game Event Listener** to this **Game Event**, called on enable.
Unregister Listener	Unregisters a **Game Event Listener** to this **Game Event**, called on disable.
Register Action	Registers a specific method as a listener to this **Game Event**, executing it on **Raise**.
Unregister Action	Unregisters a specific method as a listener to this **Game Event**.

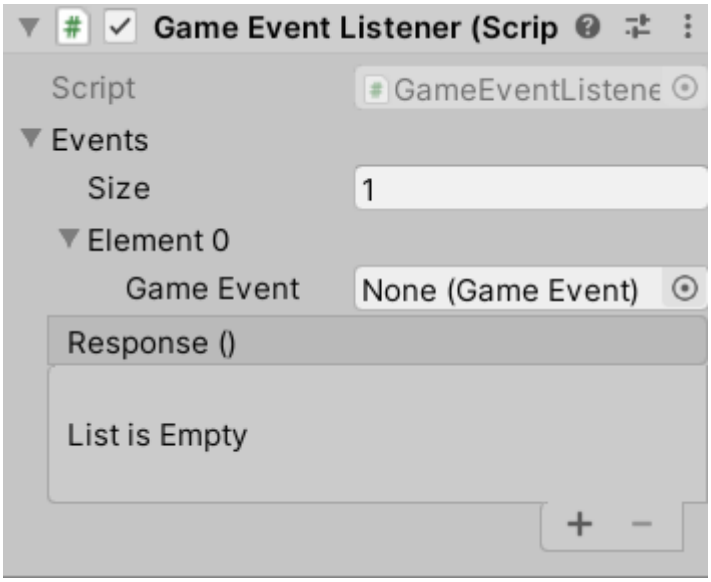
Details

Game Events are extremely helpful and generic tools that allow more freedom to designers and developers, and are one of the core elements of the Scriptable Object architecture. Game events can be created by right clicking in the project window and selecting Create>Game Event. The **Raise** method

can be called in the editor by clicking the button in the Inspector called **Raise**. Game events can also be raised in code, or in Unity Events, offering them an incredible amount of utility.

Game Event Listener

Allows for Game Events to be easily used with Unity Events.



Properties

Property:	Function:
Events	A list of **Paired Event Listeners**. Which consist of one **Game Event** reference and a Unity Event **Response**. If the **Game Event** is raised then the corresponding **Response** is invoked.

Public Methods

Property:	Function:
On Event Raised	Called by any **Game Events** that the listener is registered to when that **Game Event** is raised. Invokes the Unity Event **Response**.
Raise Event By Index	**Invokes the Response** of the **Paired Event Listener** at the specified index without calling the corresponding **Game Event**.

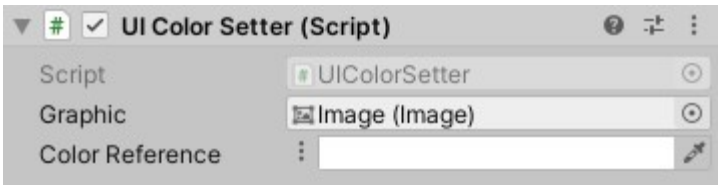
Details

Game Event Listeners are largely responsible for the actual utility of Game Events, and are helpful elements of the Scriptable Object architecture for designers.

UI

UI Color Setter

Allows designers to have more control over UI colors.



Properties

Property:	**Function:**
Graphic	The component who's **Color** should be modified.
Color Reference	A **Scriptable Color Reference**, can be manually assigned or reference a **Color Scriptable** object. The **Color** of the **Graphic** will be automatically set to that of the **Color Reference**.

Public Methods

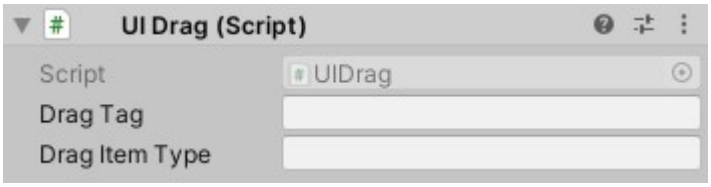
Property:	**Function:**
Set Color	Can take either a **Color Scriptable** or a **Color** and sets the **Graphic's Color** to the value.
Set Red	Sets the Red value of the **Graphic's Color**.
Set Blue	Sets the Blue value of the **Graphic's Color**.
Set Green	Sets the Green value of the **Graphic's Color**.
Set Alpha	Sets the Alpha value of the **Graphic's Color**.

Details

This tool is helpful for designers or developers to have more fine tuned control over their colors. One excellent use for it is to maintain color consistency across your UI. By setting up a primary UI color **Color Scriptable** and assigning it to all of your UI elements, you can easily make sure that you entire UI uses the same color pallet, and it can all be changed at one time to allow for rapid iteration.

UI Drag

Allows a simple implementation of draggable UI elements.



Properties

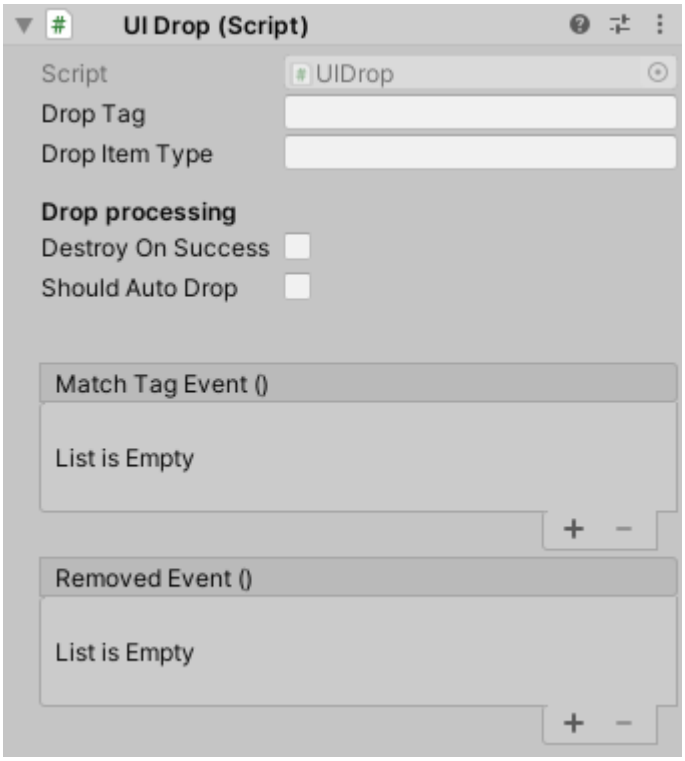
Property:	Function:
Drag Tag	A string. Is referenced with the **UI Drop** to determine whether a Unity Event should be fired.
Drag Item Type	A string. The type must match the **UI Drop** in order for it to successfully drop. This allows you to have multiple types of draggable items without them all being able to be added to into any **UI Drop**.

Details

This was created because there is no native Unity component to allow drag and drop. This is a simple framework and can be extended if needed.

UI Drag

Allows a simple implementation of draggable UI elements.



Properties

Property:	Function:
Drop Tag	A string. Is referenced with the **UI Drag** to determine whether **Match Tag Event** should be invoked.
Drop Item Type	A string. The type must match the **UI Drag** in order for it to successfully drop. This allows you to have multiple types of draggable items without them all being able to be added to into any **UI Drop**.
Destroy On Success	Determines whether the **UI Drag** object should be destroyed if the tags match. Useful if you don't want the object to actually appear over this object.
Should Auto Drop	Determines whether the player needs to release the **UI Drag** object or if it should be pulled out of the players hand.

Events

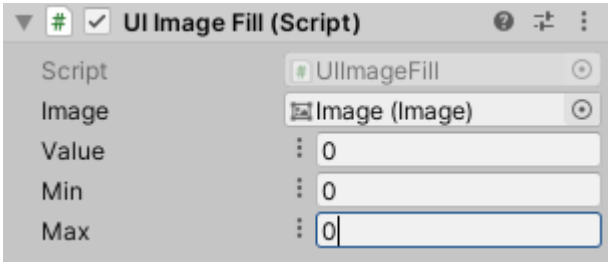
Property:	Function:
Match Tag Event	Invoked when the **Drop Tag** and **Drag Tag** are the same.
Removed Event	Invoked when the **UI Drag** object is lifted off this object.

Details

This was created because there is no native Unity component to allow drag and drop. This is a simple framework and can be extended if needed.

UI Image Fill

Allows for quick control of Fill type images.



Properties

Property:	Function:
Image	The **Image** component that will be filled.
Value	A **Scriptable Float Reference** that controls how much of the **Image** is

	filled.
Min	A **Scriptable Float Reference** that establishes the lowest point that the **Value** should be.
Max	A **Scriptable Float Reference** that establishes the highest point that the **Value** should be.

Details

This tools allows designers to easily set up UI bars and things of that nature, pairing them to whatever value you need. Health or mana bars are a common use.

UI Initialize Slider

Allows sliders to automatically be set to a referenced value.

Properties

Property:	**Function:**
Slider	The **Slider** component that will be initialized.
Value	A **Scriptable Float Reference** that controls how far the slider should fill.

Details

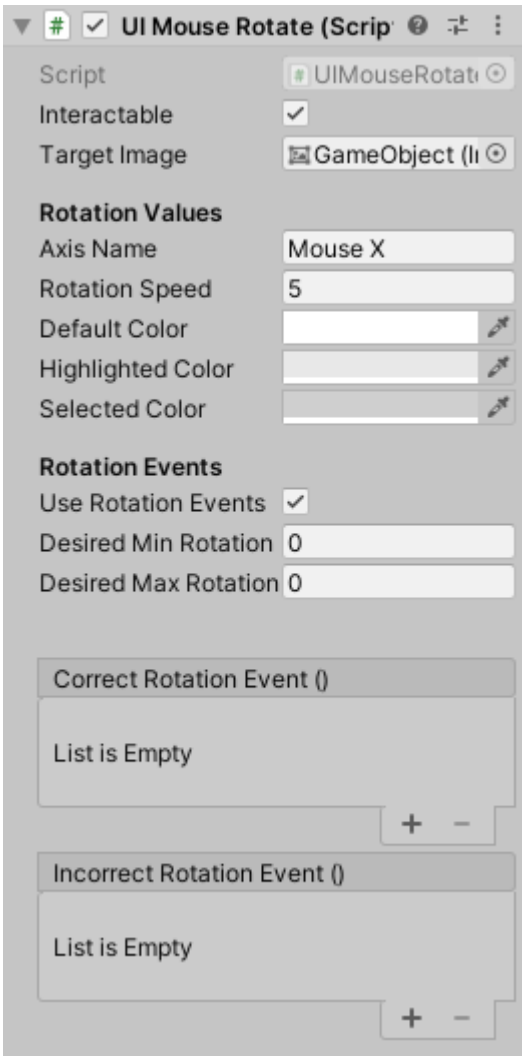
It can often be helpful to control **Float Scriptables** with **Sliders**, however you can end up with a discrepancy between the current value of the **Float Scriptable** and the displayed value of the **Slider**. This script is designed to prevent that by automatically setting the Value of the **Slider** to the value of the **Float Scriptable** on Enable (called whenever the game object is activated).

UI Mouse Follow

Sets the **Transform** of the **Game Object** to the mouse position.

UI Mouse Rotate

Allows the player to control a UI Image’s rotation by clicking and dragging with the mouse.



Properties

Property:	Function:
Interactable	Determines if the **Game Object** can be interacted with.
Target Image	The **Image** who’s **Transform** is rotated.
Axis Name	The input axis name that controls motion. Defaults to Mouse X.
Rotation Speed	The multiplier for how fast the object should rotate. Defaults to 5.
Default Color	Determines the **Image’s** color at rest.
Highlighted Color	Determines the **Image’s** color at when hovered over.
Selected Color	Determines the **Image’s** color while the mouse is held down.
Use Rotation Events	Determines if Unity Events should invoke when a certain rotation value is

	met.
Desired Min Rotation	The lowest rotation value (Z rotation) that will trigger the **Correct Rotation Event**. In the Inspector, rotation values may not match their actual value in code. The rotation must be between 0-360.
Desired Max Rotation	The highest rotation value (Z rotation) that will trigger the **Correct Rotation Event**. In the Inspector, rotation values may not match their actual value in code. The rotation must be between 0-360.

Events

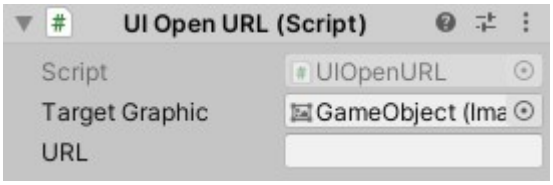
Property:	**Function:**
Correct Rotation Event	Invoked when the z rotation enters the desired range.
Correct Rotation Event	Invoked when the z rotation exits the desired range.

Details

Designed to allow designers to make UI Objects rotate with player input. Optionally allows you to trigger events based on that rotation. Could be used for something like a combination lock. Supports transparent images, allowing you to only interacting with opaque elements (requires the sprite to be set to Read/Write in import settings, will return a warning if settings are wrong).

UI Open URL

Allows URL links in Unity's UI.



Properties

Property:	**Function:**
Target Graphic	The **Image** or **Text** component that can be clicked on.
URL	The URL that should be navigated to on click.

Public Method

Property:	**Function:**
Open	Takes in a **String** as the URL to open.

Details

Designed for convenient setup of hyperlinks. Will not work with WebGL due to increased web security.

UI Text Updater

Allows designers to easily set up changing text messages in UI.

Properties

Property:	Function:
Text	The **Text** component that gets updated.
Prefix	A **Scriptable String** that is placed at the beginning of the text message.
Content	A **Scriptable String** that is placed at the center of the text message.
Float Content	A **Scriptable Float** that is placed between the **Content** and the **Suffix.**
Int Content	A **Scriptable Int** that is placed between the **Content** and the **Suffix.** If the value is less than 10, it will prefix the Int with a 0, ie. 05, 06, 10.
Suffix	A **Scriptable String** that is placed at the end of the text message.
Show Float	Determines whether the **Float Content** should be shown to the screen.
Show Int	Determines whether the **Int Content** should be shown to the screen.

Public Methods

Property:	Function:
Change Content	Sets the value of **Content** to the new string.
Change Int Content	Sets the value of **Int Content** to the new int.

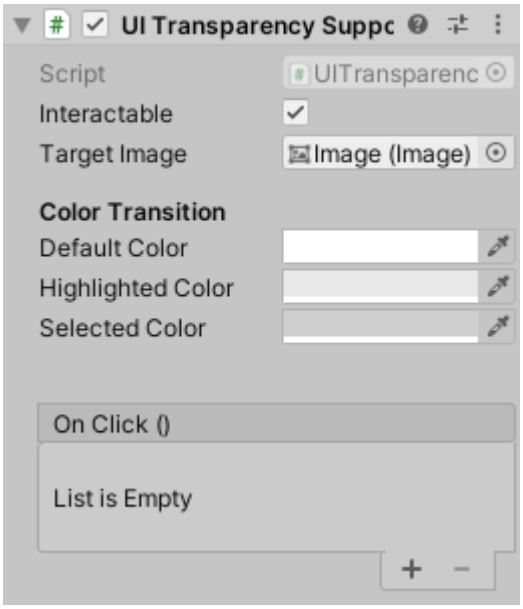
Change Float Content	Sets the value of **Float Content** to the new float.

Details

This script is designed to allow for easier control over text element in UI. Some examples include keeping track of scores, or display player names.

UI Transparency Support Button

Allows designers to set up buttons on a UI image that will ignore the mouse over transparent pixels.



Properties

Property:	Function:
Interactable	Determines if the **Game Object** can be interacted with.
Target Image	The **Image** the script applies to.
Default Color	Determines the **Image's** color at rest.
Highlighted Color	**Determines the Image's** color when hovered over.
Selected Color	Determines the **Image's** color the mouse is held down.

Events

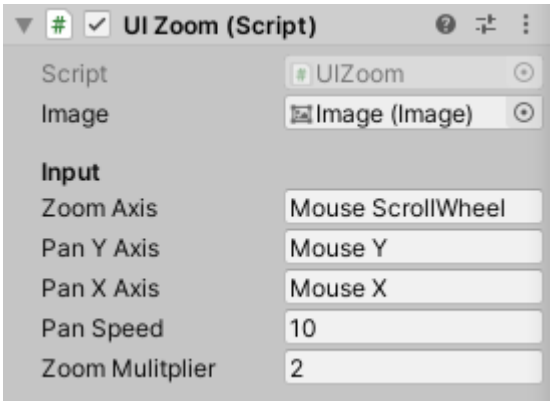
Property:	Function:
On Click	A Unity Event that is invoked when an opaque pixel on the **Target Image** is clicked.

Details

Designed to provide a function not present in standard Unity **Buttons**, allowing it to detect the difference between transparent and opaque portions of the **Target Image**.

UI Zoom

Allows simple zoom functionality for UI images.



Properties

Property:	**Function:**
Image	The **Image** component to apply zoom functionality to.
Zoom Axis	The input axis for zooming in and out. Defaults to Mouse ScrollWheel.
Pan Y Axis	The input axis for moving the zoomed image vertically. Defaults to Mouse Y.
Pan X Axis	The input axis for moving the zoomed image horizontally. Defaults to Mouse X.
Zoom Multiplier	The value the **Rect Transform** scale is multiplied by when zooming.

Public Methods

Property:	**Function:**
Zoom In	Zooms in. Is bound to the positive value of **Zoom Axis**, but can be called externally.
Zoom Out	Zooms out. Is bound to the negative value of **Zoom Axis**, but can be called externally.

Details

A simple way to support zoom in functionality for Images.

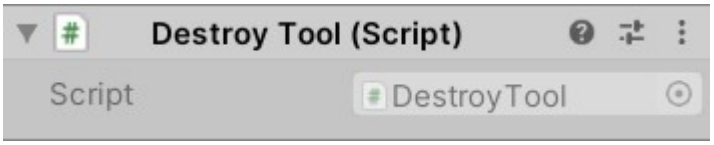
Utilities

Conditional Hide Attribute

Adds an attribute to developers, allowing fields to be hidden from the editor determined by a bool.

Destroy Tool

Allows Game Objects to be destroyed with Unity Events.



Public Methods

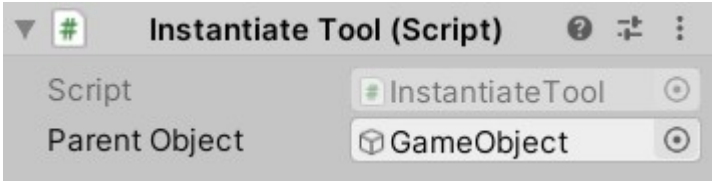
Property:	**Function:**
Destroy Object	Takes a **Game Object** reference. Destroys the referenced game object. Can be called with Unity Events.
Destroy Child	Takes an int reference. Destroys the child at the specified index. Can be called with Unity Events.

Details

A tool for designers to destroy objects at specified times without needing to write unique code using Unity Events.

Instantiate Tool

Allows prefabs to be instantiated via Unity Events.



Properties

Property:	**Function:**
Parent Object	The **Game Object** that will be the parent of the instantiated object.

Public Methods

Property:	Function:
Instantiate As Child	Takes a **Game Object** reference as prefab. Instantiates the prefab as a child of the **Parent Object**.

Details

Gives designers the ability to instantiate objects via Unity Events.

List Utilities

A static class that allows for helpful extensions of Lists.

Public Static Methods

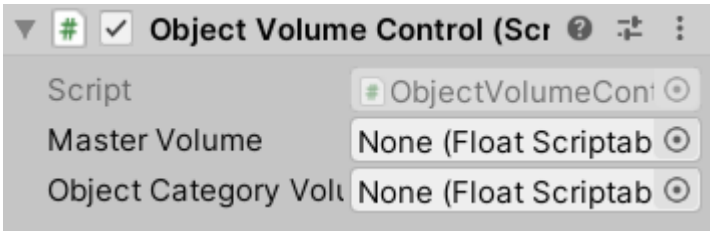
Property:	Function:
Get Next List Element	Returns the next item in a list, or if the current element is the last element in a list, will return the beginning of the list.
Get Prior List Element	Returns the prior item in a list, or if the current element is the first element in a list, will return the last element of the list.

Details

A helpful script that enables developers to make more use of Lists.

Object Volume Control

Allows designers a very simple tool to control volume and volume categories.



Properties

Property:	Function:
Master Volume	A **Float Scriptable** that serves as the master volume. Should be between 0 and 1.
Object Category Volume	A **Float Scriptable** that serves as the category volume. Should be between 0 and 1.

Details

Sets the **Volume** of an attached **Audio Source** to be **Master Volume** * **Object Category Volume**.

Persistent Object

Allows designers to have a very easy way to make an object preserve between scenes.

Properties

Property:	Function:
Disabled Scenes	A list of strings that are names of scenes that the **Persistent Object** should be disabled for.

Details

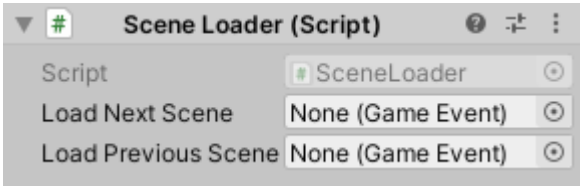
Dragging this component onto a **Game Object** automatically registers it to persist across scenes. You may use this for a music player to prevent the music from stopping and restarting when loading new scenes.

Persistent Objects Pool

The behind the scenes script that allows for the extended functionality of **Persistent Objects**.

Scene Loader

Allows designers to easily control scene loading using Unity Events or **Game Events**.



Properties

Property:	Function:
Load Next Scene	A **Game Event** reference. If the event is assigned, the next scene will automatically be loaded when the event is raised.

Load Previous Scene	A **Game Event** reference. If the event is assigned, the previous scene will automatically be loaded when the event is raised.

Public Methods

Property:	Function:
Load Scene	Takes either a string for the scene name, or an int for the build index to load.
Load Next Scene	Loads the next valid scene in the build index. Bound to the corresponding **Game Event**.
Load Previous Scene	Loads the previous valid scene in the build index. Bound to the corresponding **Game Event**.

Details

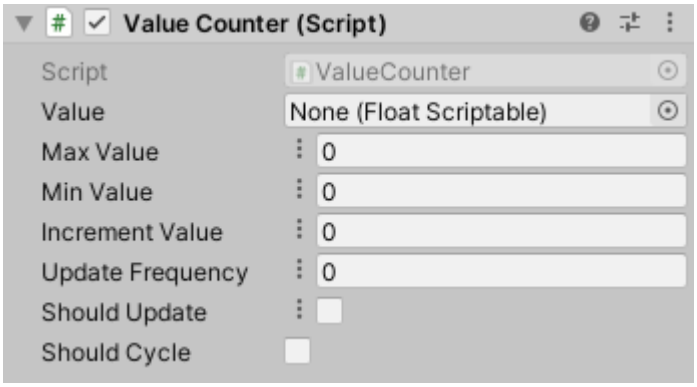
Gives designers the ability to easily to control scene loading without needing to write unique code.

Scriptable Resetter

Allows **Scriptable Variables** to have their reset functionality, setting their value back to default at the start and end of the editor play session, and before building the scenes.

Value Counter

Allows designers to easily set up increasing and decreasing values.



Properties

Property:	Function:
Value	A **Scriptable Float** reference. The value that should increment.
Max Value	A float. The highest the value should be before being set back to the **Min Value.**
Min Value	A **Scriptable Float Reference**. The lowest the value should be and will be

	reset to.
Increment Value	A **Scriptable Float Reference**. The amount that **Value** should be incremented by.
Update Frequency	A **Scriptable Float Reference**. The time between increments, in seconds.
Should Update	A **Scriptable Bool Reference.** Determines whether the **Value** should continue incrementing.
Should Cycle	A bool. Determines whether **Value** should cycle between the **Min Max** values, or should stop incrementing when either value is reached.

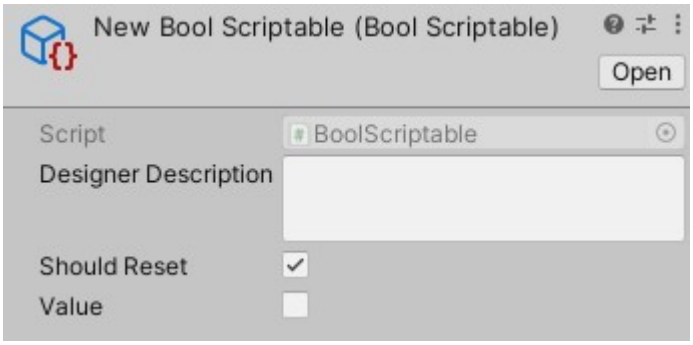
Details

This tool, especially when used in conjunction with **Conditional Event Listeners** allows designers to set up things like timers or regenerating health without writing any unique code.

Variables

Bool Scriptable

A **Scriptable Object** containing a bool. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	**Function:**
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game.

Public Methods

Property:	Function:
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Invert Value	Sets **Value** to it's opposite.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows true or false data to be changed and read without needed hard coded references to it's origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. Example uses include toggles in the UI, and flagging player progress.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>Bool.

Bool Scriptable Reference

Extends the utility of **Bool Scriptables**.

Properties

Property:	Function:
Use Local Value	Determines whether the **Local Value** or the **Reference Value** should be used.
Local Value	Set directly in the editor by a designer.
Reference Value	A **Scriptable Bool** that must be assigned if **Use Local Value** is false.

Public Methods

Property:	Function:

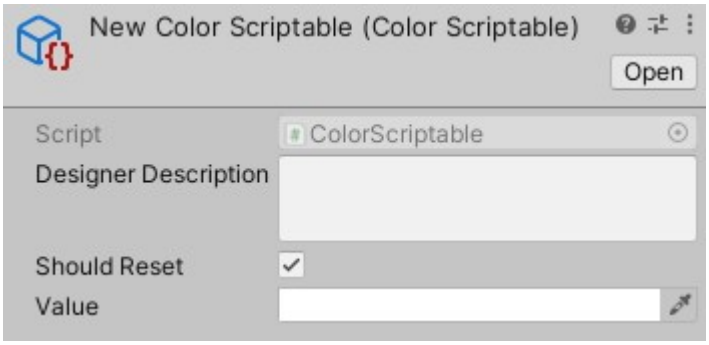
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful expansion of the **Bool Scriptable**. In most cases it is better to have a developer use a **Scriptable Reference** instead of a **Scriptable**, they function the same way in code, but **Scriptable References** have the ability to be set in the editor by designers, giving them more fine-tuned control and eliminating the need to create unnecessary **Scriptables** and cluttering up the folder structure.

Color Scriptable

A **Scriptable Object** containing a color. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	**Function:**
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game.

Public Methods

Property:	**Function:**
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.

Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows color data to be changed and read without needed hard coded references to it's origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. It is used by the **Color Setter**.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>Color.

Color Scriptable Reference

Extends the utility of **Color Scriptables**.

Properties

Property:	**Function:**
Use Local Value	Determines whether the **Local Value** or the **Reference Value** should be used.
Local Value	Set directly in the editor by a designer.
Reference Value	A **Scriptable Color** that must be assigned if **Use Local Value** is false.

Public Methods

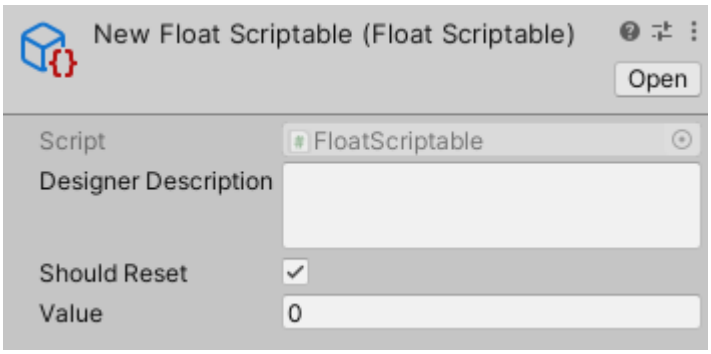
Property:	**Function:**
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful expansion of the **Color Scriptable**. In most cases it is better to have a developer use a **Scriptable Reference** instead of a **Scriptable**, they function the same way in code, but **Scriptable References** have the ability to be set in the editor by designers, giving them more fine-tuned control and eliminating the need to create unnecessary **Scriptables** and cluttering up the folder structure.

Float Scriptable

A **Scriptable Object** containing a float. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	Function:
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game.

Public Methods

Property:	Function:
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Modify Value	Adds a float or **Float Scriptable** to the **Value**.
Increment To Limit With Cycle	Adds 1 to the **Value** up to the **Max Value** parameter. If the **Max Value** is exceeded, **Value** is reset to 0.
Increment To Limit	Adds 1 to the **Value** up to the **Max Value** parameter.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.

Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows floats to be changed and read without needed hard coded references to it's origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. Example uses include tracking player health, or use with the **Object Volume Control**.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>Float.

Float Scriptable Reference

Extends the utility of **Float Scriptables**.

Properties

Property:	**Function:**
Use Local Value	Determines whether the **Local Value** or the **Reference Value** should be used.
Local Value	Set directly in the editor by a designer.
Reference Value	A **Scriptable Float** that must be assigned if **Use Local Value** is false.

Public Methods

Property:	**Function:**
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

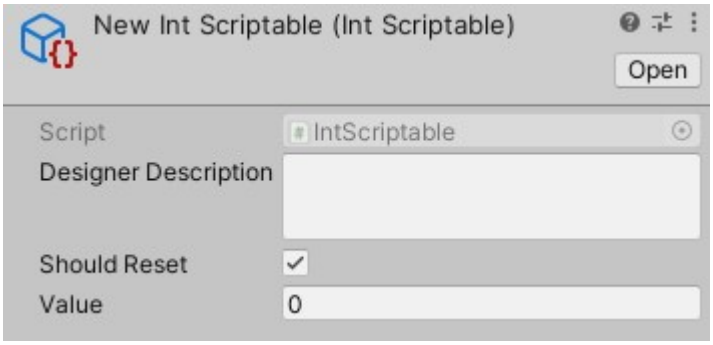
Details

A useful expansion of the **Float Scriptable**. In most cases it is better to have a developer use a **Scriptable Reference** instead of a **Scriptable**, they function the same way in code, but **Scriptable**

References have the ability to be set in the editor by designers, giving them more fine-tuned control and eliminating the need to create unnecessary **Scriptables** and cluttering up the folder structure.

Int Scriptable

A **Scriptable Object** containing an int. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	Function:
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game.

Public Methods

Property:	Function:
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Modify Value	Adds a float or an **Int Scriptable** to the **Value**.
Increment To Limit With Cycle	Adds 1 to the **Value** up to the **Max Value** parameter. If the **Max Value** is exceeded, **Value** is reset to 0.
Increment To Limit	Adds 1 to the **Value** up to the **Max Value** parameter.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows ints to be changed and read without needed hard coded references to it's origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. Example uses include timers or counters.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>Int.

Int Scriptable Reference

Extends the utility of **Int Scriptables**.

Properties

Property:	**Function:**
Use Local Value	Determines whether the **Local Value** or the **Reference Value** should be used.
Local Value	Set directly in the editor by a designer.
Reference Value	A **Scriptable Int** that must be assigned if **Use Local Value** is false.

Public Methods

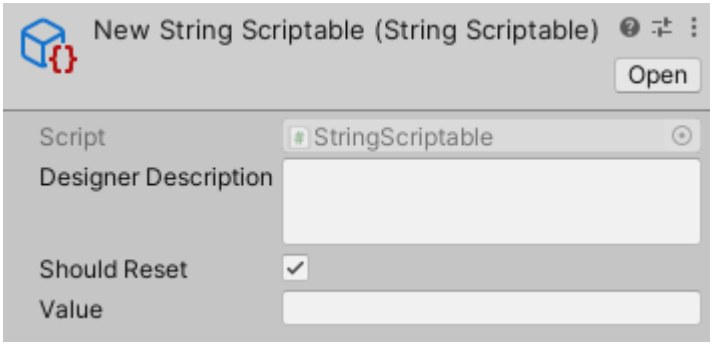
Property:	**Function:**
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful expansion of the **Int Scriptable**. In most cases it is better to have a developer use a **Scriptable Reference** instead of a **Scriptable**, they function the same way in code, but **Scriptable References** have the ability to be set in the editor by designers, giving them more fine-tuned control and eliminating the need to create unnecessary **Scriptables** and cluttering up the folder structure.

String Scriptable

A **Scriptable Object** containing a string. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	**Function:**
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game.

Public Methods

Property:	**Function:**
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Add String	Adds a string or **String Scriptable** to the **Value**.
Remove Characters	Takes an int. Subtracts that number of characters from **Value**.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows strings to be changed and read without needed hard coded references to it’s origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. Example uses include player name, or use in code checking.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>String.

String Scriptable Reference

Extends the utility of **String Scriptables**.

Properties

Property:	**Function:**
Use Local Value	Determines whether the **Local Value** or the **Reference Value** should be used.
Local Value	Set directly in the editor by a designer.
Reference Value	A **Scriptable String** that must be assigned if **Use Local Value** is false.

Public Methods

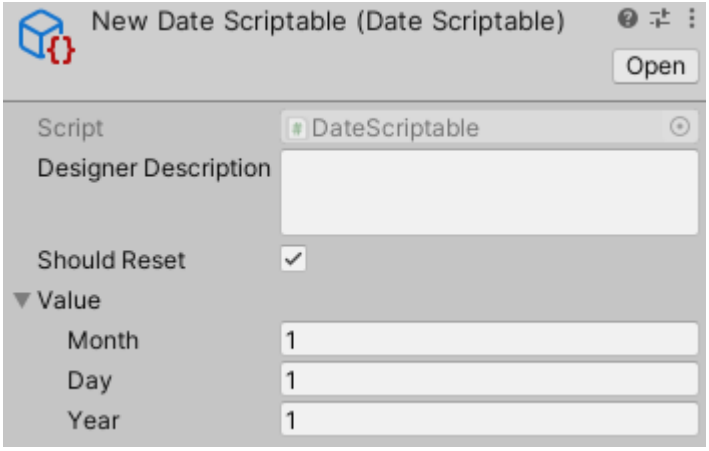
Property:	**Function:**
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful expansion of the **String Scriptable**. In most cases it is better to have a developer use a **Scriptable Reference** instead of a **Scriptable**, they function the same way in code, but **Scriptable References** have the ability to be set in the editor by designers, giving them more fine-tuned control and eliminating the need to create unnecessary **Scriptables** and cluttering up the folder structure.

Date Scriptable

A **Scriptable Object** containing a date. One of the foundations of the Scriptable Object Architecture used in this tool set.



Properties

Property:	Function:
Designer Description	A string that is not in game. Used as a comment or simple description by the designer.
Should Reset	Determines whether the **Value** should be reset when exiting play mode, or if any changes made during play mode should be preserved.
Value	The value used in game. The year value should be in the four digit format.

Public Methods

Property:	Function:
Init	Saves the **Value**. Called automatically when entering play mode.
Reset	Resets the **Value**. Called automatically when exiting play mode.
Register Listener	Takes an **Action** or an **Event Handler** and binds it to this scriptable. Calling the registered method whenever the **Value** changes.
Unregister Listener	Takes an **Action** or an **Event Handler** and unbinds it from this scriptable. It is good practice to always unregister a method when the attached object is destroyed to prevent null references.

Details

A useful tool that allows dates to be changed and read without needed hard coded references to it's origin, and allows for reduction of dependencies and extends control to designers, rather than restricting it to developers. Example uses referencing the system date.

Scriptables' data is preserved between scenes, and the reset functionality is used both when entering or exiting play mode, and before scenes are built. The **Value** can be changed by designers during play mode and the registered listeners will still be invoked.

They can be quickly made by right clicking in the project window and selecting Create>Scriptable Variables>Date.

Credits and Attributions

Conditional Hide Attribute

Made by Brechtos, modified by me. http://www.brechtos.com/hiding-or-disabling-inspector-properties-using-propertydrawers-within-unity-5/

Conditional Hide Property Drawer

Made by Brechtos. http://www.brechtos.com/hiding-or-disabling-inspector-properties-using-propertydrawers-within-unity-5/

Game Event, Game Event Listeners & Event Editor

Made by Ryan Hipple, modified by me. https://github.com/roboryantron/Unite2017, https://www.youtube.com/watch?v=raQ3iHhE_Kk

Float Scriptable, String Scriptable, Float Scriptable Reference, String Scriptable Reference, Float Scriptable Reference Drawer & String Scriptable Reference Drawer

Inspired by Ryan Hipple's code. modified by me. https://github.com/roboryantron/Unite2017, https://www.youtube.com/watch?v=raQ3iHhE_Kk

UI Image Fill Setter, UI Text Updater

Made by Ryan Hipple, extensively modified by me. https://github.com/roboryantron/Unite2017, https://www.youtube.com/watch?v=raQ3iHhE_Kk

FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.