nullsave

STATS COG

Version 1.9 – Released August 2020

Contents

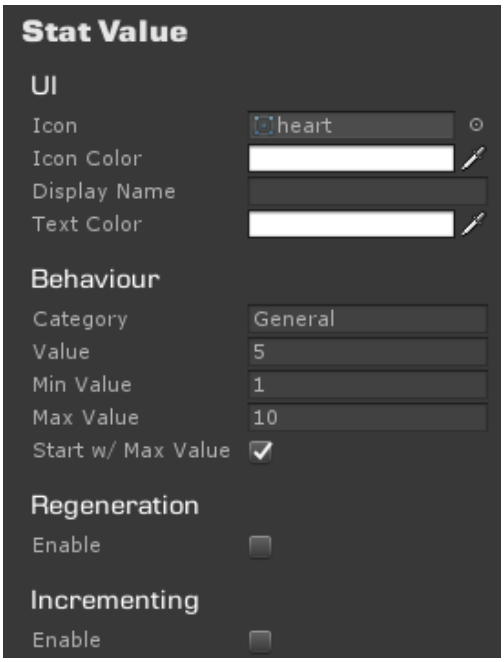
Introduction

The **Stats Cog™** is designed to allow you to create interdependent, formula-based stats and effects for your characters and objects without coding.

The full API is available here: http://www.nullsave.com/api/stats-cog

Simple StatValues

StatValues represent the current values of Stats for your character and can have numerical or equation values. To create a new StatValue right-click in your project window and select Create > TOCK > Stats Cog > Stat Value.

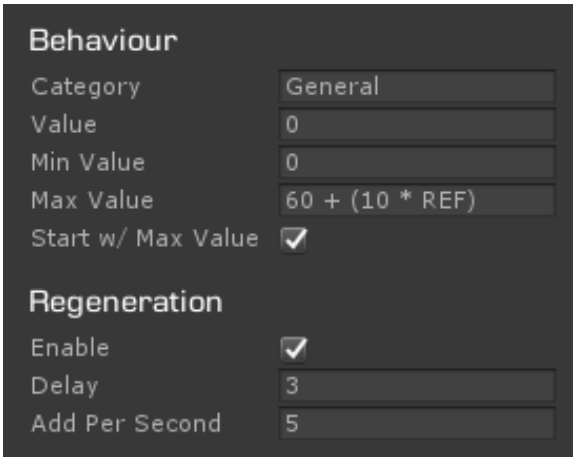


The UI section is provided for your use when displaying effects and does not change how effects are used by Stats Cog. Values here can be safely ignored if desired.

In this example we have a simple StatValue with a minimum value of 1, a maximum value of 10 and a starting value of 5. If we checked "Start w/ Max Value" our initial value would be 10 even though we have a supplied value of 5. The Category can be used to help filter effects.

Complex StatValues (Equations and Regeneration)

StatValues with equations can reference other StatValues to set their minimum, maximum, and current values. You can also add Regeneration to your StatValue to have it automatically restore itself to its maximum value over time.



In this example the maximum value will always be 60 plus 10 times the value of the REF StatValue. If the value of the REF StatValue changes, the maximum value of AP will change too.

Since Regeneration is enabled here 3 seconds after any loss of value AP will raise its value by 5 each second.

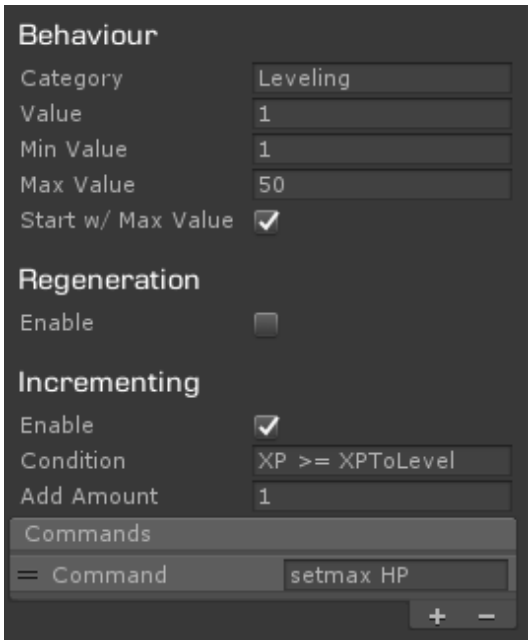
Complex StatValues (Incrementing)

In addition to equations and regeneration we can specify conditions to automatically increment our StatValue and even execute commands when that happens.

In this example the value is a simple integer and changes based on the “Condition”. Here we can see anytime the value of XP is greater than or equal to the value of XPToLevel we add 1 to our current value.

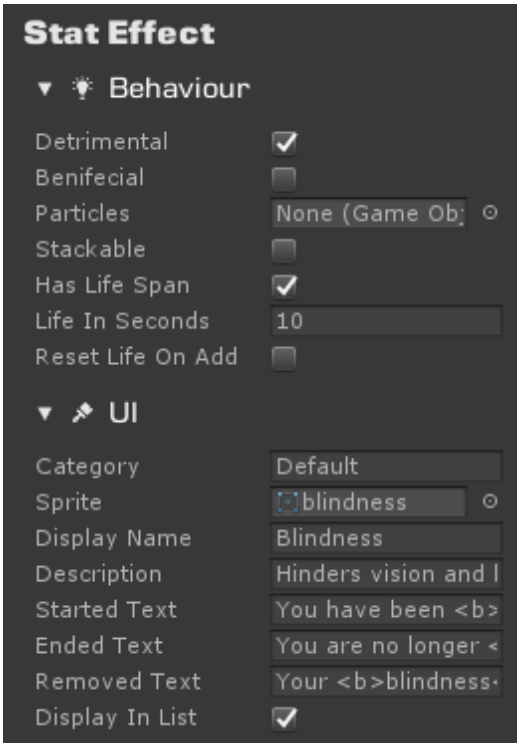
If we desired, we could also send commands to the StatsCog when we increment our value. In this example we send “setmax HP” which will set the value of our HP StatValue to its maximum allowed value.

This allows us to automatically restore our character to full health on level up without having to do any coding.



StatEffects

StatEffects are effects that can be applied to your StatsCog. They can affect **multiple** StatValues or just one. StatEffects can be permanent or automatically end after a time.



Here we are looking at the "BlindnessSpell" StatEffect in your example project (Prefabs > StatEffects).

The UI section is dedicated to display information and is not used by the StatsCog at all, it is simply there for you to use in your game or ignore completely if you like.

The Behaviour section tells us how the effect is used. If the effect is marked as "Detrimental" it will be removed by the StatsCog.RemoveDetrimentalEffects method. "Benificial" effects are removed by the RemoveBenificialEffects method and effects not marked as either are removed by the RemoveNeutralEffects method. And effect can be both Benificial and Detrimental if you like.

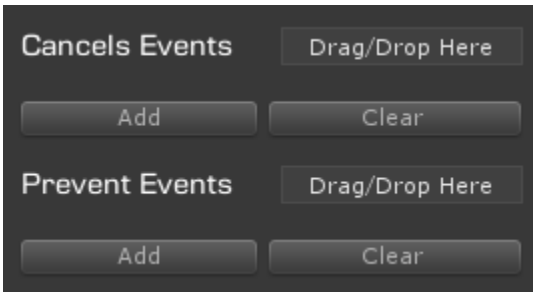
If a "Particles" prefab is provided it will be instantiated on the GameObject when the effect starts and destroyed when the effect ends or is removed.

If "Stackable" is selected the StatsCog will allow multiple instances of this effect. "Has Lifespan" means that the effect will automatically end on its own after a number of seconds (specified by "Life in Seconds"). If "Reset Life On Add" is checked when this effect is applied to a character and that effect already exists the lifetime of that effect will be reset to the original value.

In this example the effect is Detrimental, will automatically be removed 15 seconds after it is added and only 1 instance will be allowed at a time.

The Modifiers section allows you to specified modifiers for the effect. You can have as many modifiers on an effect as you which targeting multiple Stat Values.

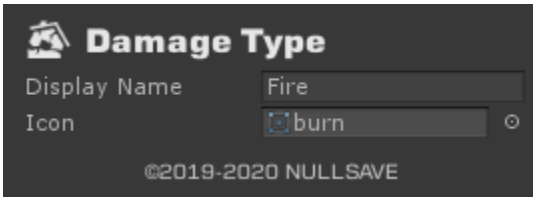
Icon – Icon to display for modifier in UI
Display Text – Text to display for modifier in UI
Text Color – Color to apply to text in UI
Hide In List – If checked modifier will be ignored by UI
Affected Stat – Name of Stat Value or Damage Modifier to affect.
Effect Type – Type of modification (Instant, Sustained, Recurring)
Target Value – Value to Target with modifier (Value, Minimum, Maximum, Regen Delay, Regen Amount)
Value Type – Math to use when applying modifier (Add, Add Multiplier, Subtract, Subtract Multipler)
Value – Value to use, this can contain simple values or equations just like a Stat Value



Any StatEffects listed in "Cancel Effects" will automatically be ended when this effect is added to the StatsCog.

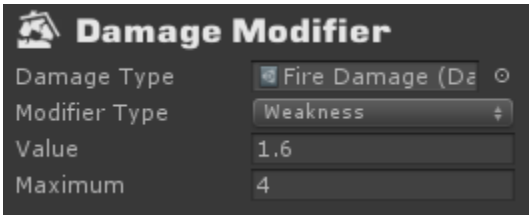
No StatEffect listed in "Prevent Effects" will be allowed to be added while this effect is active. Effects already present will be cancelled.

Damage Types



In order to deal or receive damage, Damage Types need to be created. This allows for resistances and weakness. Creating them as ScriptableObject also helps reduce the chance of typos between entries. You can create a new damage type via Create > TOCK > Combat > Damage Type

Damage Modifiers



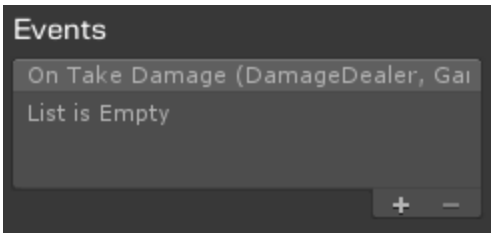
Damage Modifiers (Create > TOCK > Combat > Damage Modifier) tells Stats Cog how to modify damage.
Damage Type – Type of damage
Modifier – Resistance or Weakness
Value – Amount to modify damage
Maximum – Maximum mod with other effects.

Damage Dealer

Damage Dealers are GameObjects attached to characters or weapons that inflict damage. A Damage Dealer requires a collider to be attached to your object. The collider should be a trigger and disabled by default. The Damage Dealer will enable and disable the collider as needed.

Sec Between Damage – The amount of time to wait after inflicting damage to inflict damage a second time. This helps prevent hitting an enemy twice in one animation. If you want to allow this, simply set the value to 0.
Damage Type – Dealers can inflict multiple types of damage if desired. Simply add one for each type you wish to deal. The **Base Amount** can be tied to a StatValue (as seen here) or a simple value.
Effects – List of effects to apply when hit by damage dealer

Damage Receiver

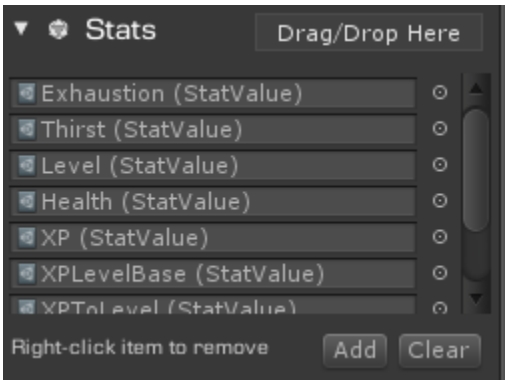


Damage Receivers allow your character or object to take damage. Like Damage Dealers you will need to have a collider on the object. You can place a single receiver for your entire character, or you can place multiple receivers.

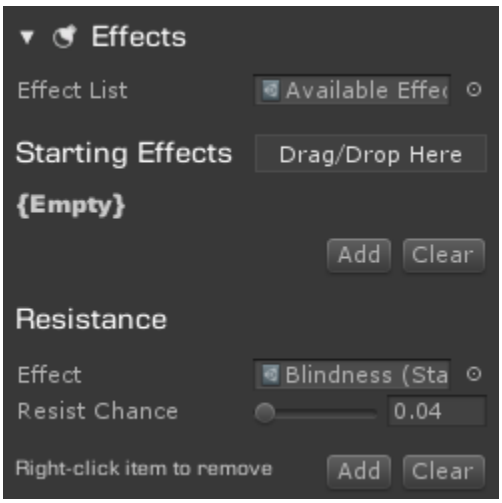
For example, a collider could be added to each limb to allow for a particle effect to spawn on a limb specific location or even allow for limb crippling.

Stats Cog

The **Stats Cog™** script can be attached to any GameObject and tracks the stats and effects for that object. The UI is divided into sections for easy use and provides a debug screen while the project is running.



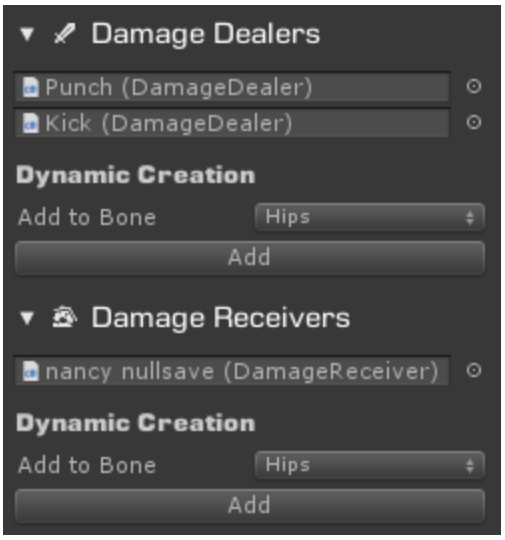
The first section allows you to drag and drop (or manually add) all the Stat Values available to this character. While these are Scriptable Objects you **do not need** to create a unique copy for each character, Stats Cog will instance these dynamically at runtime. You only need to have different copies if the values are calculated differently. For example, if HP on one character is a static value of 5 and the other has on equation of Level + 5, then they would need to be different.



The Effects section lets you deal with effects on the character.
Effect List – A list of all valid effects for this character. Effects not in the list will not be applied.
Starting Effects – These affects are immediately applied to the character at start
Resistance – This section allows you to create a chance to resist effects that are listed here.

The combat section tracks the characters health and damage.

Health Stat – Drop down of all stats on this character, allows you to pick which controls health
Damage Value – Calculates how damage is applied **after** resistances and weaknesses are considered. [Damage] is replaced with the base damage + weakness – resistance
Direction Immunity – Your character will ignore damage coming from direction selected here
Damage Mods – Allows you to add damage resistances and weaknesses in the form of Damage Modifiers to your character. These are considered when calculating final damage.



Damage Dealers and Damage Receivers are how your character deals and receives damage, respectively. If your character has an animator and human rig, you will be able to add them automatically to any bone from this interface. Otherwise, you will need to add them manually.

The **Events** section provides easy access to events fired from Stats Cog.

The Debug section is only available while running the project in Unity.

From this section you can see any Stat Value, active modifiers and effects as well as issue commands from the Console box.

Commands available are detailed in the next section.

Stats Cog (Send Command)

The SendCommand method on the StatsCog allows you to send commands from your scripts, StatEffects, the Debug tab, or an in-game console. Commands are not case-sensitive but StatValue and StatEffect names are.

Add Command

The add command allows you to add an active StatEffect by its name.

Example: add BlindnessSpell

Remove Command

The remove command allows you to remove an active StatEffect by name. If there are multiple instances, it will remove the first one on the list.

Example: remove Poison

RemoveAll Command

The removeall command removes all instances of an active StatEffect by its name.

Example: removeall BuffUps

Clear Command

The clear command removes all active StatEffects.

Example: clear

Max Command

The max command allows you to set the **base** maximum value for a StatEffect (base values are the values before any effects or modifiers are applied).

Example: max HP 300

Min Command

The max command allows you to set the **base** minimum value for a StatEffect (base values are the values before any effects or modifiers are applied).

Example: min HP 10

SetMax Command

The setmax command sets a StatValue to its current allow maximum value.

Example: setmax HP

Value Command

The value command sets the current value of a StatValue to the supplied value. You can reference other StatValues here just as you would in a value equation.

Example: value HP = (HP + 10) * (END / 2)

Change Log

Version 1.9

Breaking Changes

- Changed Damage Resistance/Weakness options to DamageModifier ScriptableObject
- Added DamageType ScriptableObject
- Updated HitDirection enum values to be compatible with bitmasking
- Removed Damage Resistance/Weakness from Stats Cog, replaced with Damage Modifiers
- Changed damageType from string to DamageType in Damage
- Removed FindResistence and FindWeakness

Fixes

- Fixed issue where effects with no modifiers but has cancel/remove were not added to active
- Fixed bug where current modifiers were not ended on Load
- Fixed issue where stats did not honor an Instant modifier at the same time as a Sustained modifier

Updates

- Added Direction Immunity to Stats Cog
- Added Effect List component
- Added Effect UI component
- Streamlined editor GUI
- Added Damage Dealers/Receivers list to Stats Cog editor
- Changed "Health Stat" to be a dropdown with all available stats
- Improved save/load

Version 1.8

Improvements

- StatsCog no longer derives from Damage Receiver
- StatsCog now subscribes to all Damage Receiver children
- Added restore-min command
- Fixed issue with Category not persisting for StatValues

- Fixed issue with inspecting StatsCog prefab at runtime
- Added more custom editors
- Additional 2D support
- Added welcome screen
- Removed DataStore from Shared
- Added onColliderEnabled and onColliderDisabled events to DamageDealer
- Added 2D support to DamageDealer
- Added onTakeDamage to DamageReceiver
- Updated base TakeDamage method on DamageReceiver

Version 1.7

Improvements

- Added ExpressionSubcription class
- Added GetSubscriptionRequirements method to StatsCog

Version 1.6

Improvements

- Added Immunity After Hit to Stats Cog
- Added HitDirection enum (FrontLeft,Front,FrontRight,Left,BackLeft,Back,BackRight,Right)
- Added onHitDirection event to StatsCog (raises direction of impact even if no damage taken)
- Added onHitDamageDirection event to StatsCog (raises direction of impact when damage is taken)
- Added resetLifeOnAdd to StatEffect

Version 1.5

Updates

- Updated Editors

Version 1.4.1

Updates

- Updated compatibility with Inventory Cog 1.3

Version 1.4

Improvements

- Lowered minimum version to 2018.4.0f1
- Added Starting Effects to StatsCog
- Created new demo with "Survival" stats and combat
- Moved "Hierarchy Icons" into Shared directory
- Updated custom editors
- Regen now uses Time.deltaTime to pause regen in menus, etc
- Added StatMonitorTMP MonoBehaviour
- Added custom editor to TextStatMonitor
- Added Effect Resistance to StatsCog
- Added onEffectResist event to StatsCog
- Added TriggerByCondition MonoBehaviour
- Made Debug the default tab while player is running for StatsCog
- Added Stat Effect List to streamline adding effects to multiple StatCogs
- Replaced Available Effects with Stat Effect List on StatsCog
- Added Events view to StatsCog
- Added UI fields to StatValue (icon, iconColor, displayName, textColor)
- Added TriggerByEffect MonoBehaviour

Fixes

- Fixed issue where instant negative modifiers didn't apply properly in some cases
- Fixed issue where negative modifers didn't reset RegenDelay
- Fixed issue with expanding/collapsing sections in custom editors
- Fixed issue in custom editor when inspecting a disabled item at runtime in debug view

Version 1.3

Improvements

- Updated editor to make Commands a reorderable list for StatValues
- Added GetEffectsByCategory to StatsCog
- Added GetValueByCategory to StatsCog
- Added version to Save/Load data to prevent future breaking changes

- Added Damage class to Shared
- Added DamageDealer MonoBehaviour to Shared
- Added DamageReceiver MonoBehaviour to Shared
- Added DamageResistance class to Shared
- Added DamageWeakness class to Shared
- Added StatDamageResistence
- Added StatDamageWeakness
- Updated StatsCog to derive from DamageReceiver
- Added resistances to StatsCog
- Added weaknesses to StatsCog
- Added FindResistance to StatsCog
- Added FindWeakness to StatsCog
- Updated AddEffect to apply to resistances and weaknesses
- Added new Initialize overloads to StatModifier for resistances and weaknesses
- Added new CalculateAppliedValue overloads to StatModifer for resistances and weaknesses
- Added new GetSustainedValue overloads to StatModifier for resistances and weaknesses
- Added UpdateDamageDealers to StatsCog
- Added healthStat and damageValue to StatsCog
- Added onDamageTaken and onImmuneToDamage events to StatsCog

Fixes

- Corrected Category to appear in editor for StatValues and StatEffects
- Corrected editor issue displaying Modifiers

Breaking Changes

- Default save/load structure changed, current saved data will not load properly. Replace with new save.

Version 1.2

Improvements

- Added ability to regen by percent
- Added ability to increment by percent
- Added category to StatEffects
- Added RemoveEffectsByCategory
- Added category to StatValues (game use only, does not affect StatsCog)
- Added icon, displayText, textColor, and showInList to StatModifier
- Added GetModiferChange method to StatsCog and StatValue
- Added AddInventoryMods and RemoveInventoryMods to StatsCog

- Added support for Inventory Cog
- Added CalculateAppliedValue to StatModifier
- Changed ShowInList to HideInList on StatModifer
- Added AddInstantInventoryMods to StatsCog
- Added AddInstantModifer to StatValue

Fixes

- Corrected RemoveNeutrallEffects to RemoveNeutralEffects

Breaking Changes

- Default save/load structure changed, current saved data will not load properly. Replace with new save.

Version 1.1

Improvements

- Added 3rd person demo
- Added onEffectAdded event to StatsCog
- Added onEffectEnded event to StatsCog
- Added onEffectRemoved event to StatsCog
- Added EndEffect methods to StatsCog
- Added addedText, endedText, and removedText to StatEffect
- Added effectParticles to StatEffect
- Added isDetrimental to StatEffect
- Added isBenificial to StatEffec
- Added isRemoveable ot StatEffect
- Added RemoveBenificialEffects to StatsCog
- Added RemoveDetrimentalEffects to StatsCog
- Added RemoveNeutralEffects to StatsCog
- Added incrementCommand to StatValue
- Added a debug editor that allows you to see current and base values, active effects and send commands
- Updated all StatValue Changed events to return old and new values
- Replaced EditorIcons with our new HeirarchyIcons
- Re-ordered file structure
- Changed LogicExtensions namespace to NullSave.TOCK.Stats
- Moved TextStatMonitor and SliderStat into main project
- Added Hierarchy icons to TextStatMonitor and SliderStat
- Made StatValue's increment command an array to allow for running multiple commands on increment

Fixes

- Fixed issue where changing a StatValue on one GameObject affected the same StatValue on other GameObjects
- Fixed issue that prevented stacking effects.
- Fixed an issue that allowed stat values to regenerate too quickly
- Fixed issue that allowed stat value to be set outside of min/max values
- Fixed issue where StatValues would subscribe multiple times if the same StatValue was used multiple times in an equation
- Fixed issue when removing a modifier that would have pushed value beyond maximum value is calculated incorrectly
- Fixed issue where RegenAmount and RegenDelay sustained effects were not honored.

Breaking Changes

- StatsCog.stats should no longer be reference directly, please use StatsCog.Stats
- StatsCog.ActiveEffects has been renamed StatsCog.Stats