ULTIMATE TIMER

EVENT ENGINE FOR UNITY

USER GUIDE 1.1.0

What's New

1. TimeScale has been added as a new output. This allows for slowing down or speeding up time passed over a specified time frame. A common use for this would be for a slow-motion or bullet time effect.
2. Note: A second Ultimate Timer script will be needed to return the TimeScale value back to 100% if you're looking to undo the initial change. See it in the inspector

Overview

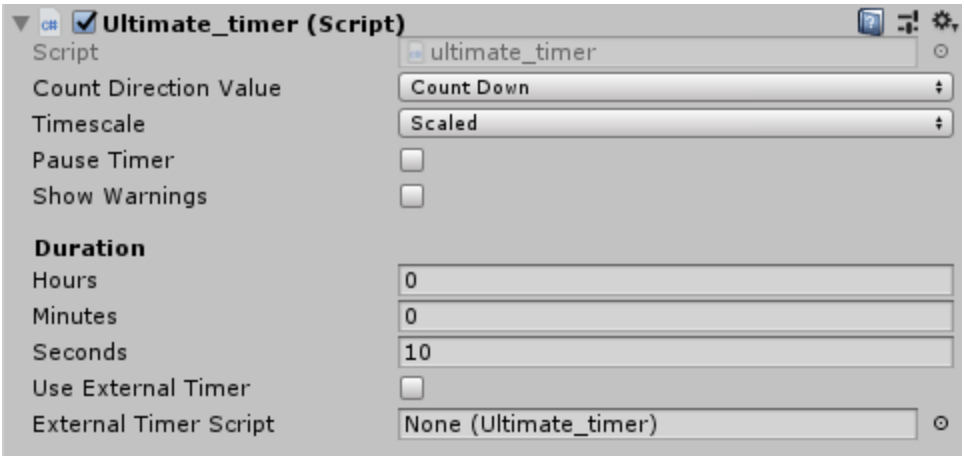
Ultimate Timer is more than just a stopwatch, it's an event engine. This asset allows for endless amounts of control over hundreds of objects and components over a specified time frame.

Whether you're looking to track time, create compelling cinematic sequences by lerping camera movement and rotation, racking up points, auto-saving player progress, cross-fading scenes, or calling multiple custom functions over time, this asset is built to handle these tasks and more.

Features:

- Countdown from any positive value
- Count up to any value
- Count up infinitely
- Supports scaled and unscaled time
- Duration accepts hours, minutes and seconds
- TextMesh Pro ready
- 8 output modes
- 12 public functions
- Call custom functions using Event Intervals
- 5 built-in timer complete functions with the ability to add your own

TImer Options



Count Direction Value

- Countdown
- Count Up
- Count Up Infinite

Timescale
This is the scale at which time passes. This can be used for slow motion effects or to speed up your application. When timeScale is 1.0, time passes as fast as real time. When timeScale is 0.5 time passes 2x slower than realtime.

- Scaled - When selected, Ultimate Timer will slow down or speed up with the project's time scale.
- Unscaled - When selected, Ultimate Timer will run independently from the project's time scale. For example, if the project requires a slow motion that affects animation but not the timer, then Unscaled would be selected.

Pause Timer
Pauses the timer on initial load.

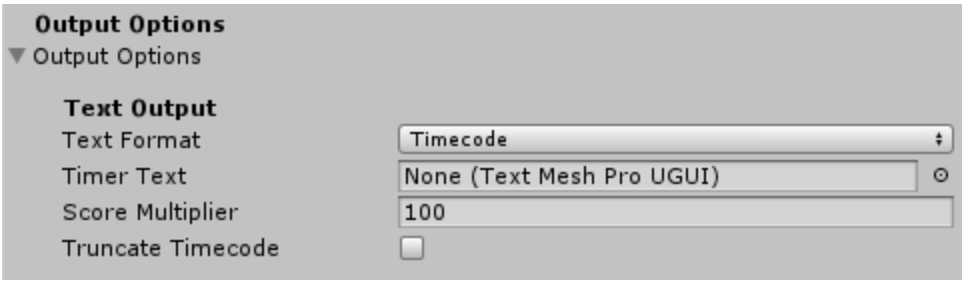
Show Warnings
Custom awnings were written to help troubleshoot. Turn this on for additional help.

Duration
Express the length of your timer in Hours, Minutes or Seconds. You do not need to type a value in minutes or hours if your seconds equal the same length. 120 seconds will be the same as 2 minutes. Also, the inputs accept floats, so 0.5 minutes will be the same as 30 seconds.

Use External Timer
Check this box if you plan to use another timer's duration instead of this timer.

External Timer Script
Drag the script or the game object containing another Ultimate TImer script that you want to link it's duration to. This allows for chaining multiple timers together.

Output Options



Text Output
Text Format

- Timecode - Outputs text in 00:00.0 format
- Percent - Outputs text as a percent complete. When it reaches 100%, the timer is complete

- Score - Converts time passed to a score. Use Score Multiplier to adjust how many points seconds are worth.
- Seconds - Time passed represented as rounded up seconds. Example: 3, 2, 1, GO!

Timer Text

This is the TextMesh Pro text field that will be rendering your timer's text output

Score Multiplier

When "Score" is selected from *Text Format*, this value will multiply against each passing second allowing seconds to equal a score amount.

Truncate Timecode

When "Timecode" is selected, check this box to remove excess 00's preceding your countdown. For example, a timer at 5 seconds into counting up to 10 minutes would look like this 00:05.00. Checking Truncate Timecode will format it to 05.00 instead.

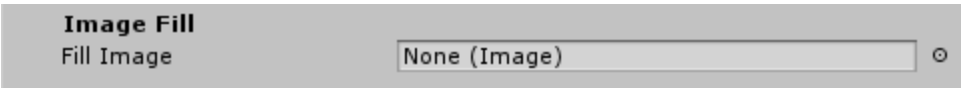


Image Fill

If your image type is set to Filled, Ultimate Timer can access the fill amount and calculate against time passed. See Using Image Fills for more info.

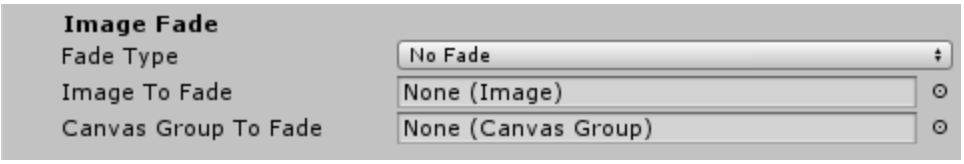
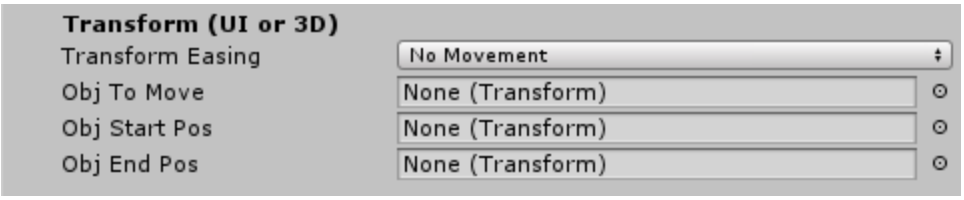


Image Fade

Targets the alpha color of Images or Canvas Groups which can be faded in or out over time.

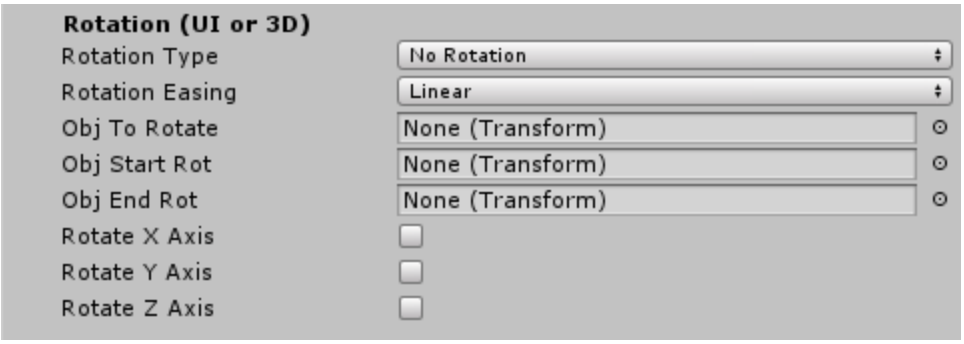
- Fade Type - No Fade, Fade In, Fade Out
- Image to Fade - This is the UI image object that will be fading
- Canvas Group to Fade - This is the Canvas Group you will be fading



Transform (UI or 3D)

Set start and end positions to move objects from one point to another over time.

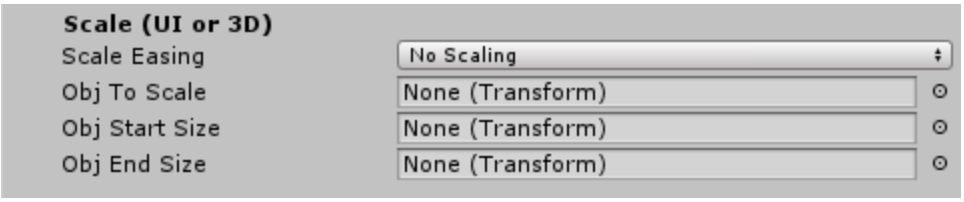
- Transform Easing - No Movement, Linear, Ease In, Ease Out, Smooth
- Obj to Move - This is the Game Object that will be moving
- Obj Start Pos - The Game Object that represents the original point for *Obj to Move*
- Obj End Pos - The Game Object that represents the final position for *Obj to Move*



Rotate (UI or 3D)

Rotate objects infinitely around specific axis or inherit start and stop rotation values

- Rotation Type - No Rotation, Clockwise, Counterclockwise, Inherit from start and end rotation
- Rotation Easing - Linear, Ease In, Ease Out, Smooth
- Obj to Rotate - This is the Game Object that will be rotating
- Obj Start Rot - The Game Object that represents the original rotation value for *Obj to Rotate* when *Inherit from start and end rotation* is selected
- Obj End Rot - The Game Object that represents the final rotation value for *Obj to Rotate* when *Inherit from start and end rotation* is selected
- Rotate X Axis - Check this to rotate along the X axis of this Game Object
- Rotate Y Axis - Check this to rotate along the Y axis of this Game Object
- Rotate Z Axis - Check this to rotate along the Z axis of this Game Object



Scale (UI or 3D)

Increase or decrease a Game Object's scale over time

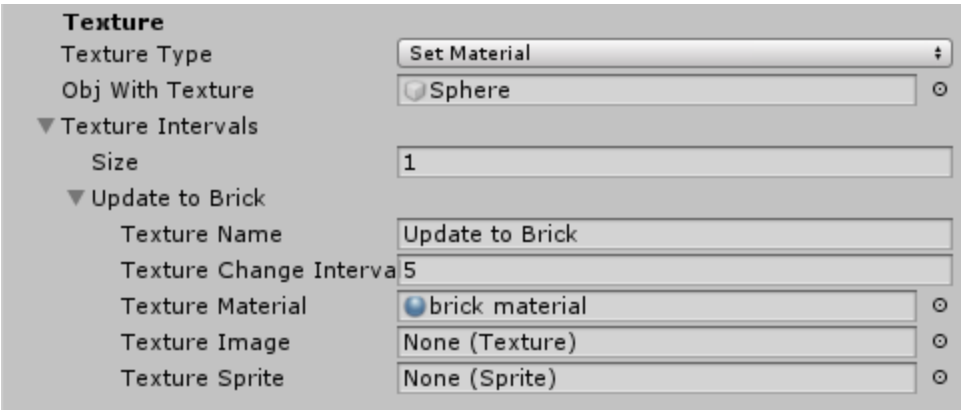
- Scale Easing - No Scaling, Linear, Ease In, Ease Out, Smooth
- Obj to Scale - This is the Game Object that will be scaling
- Obj Start Size - The Game Object that represents the original scale value for *Obj to Scale*
- Obj End Size - The Game Object that represents the final scale value for *Obj to Scale*

TimeScale (Slow Motion)

TImeScale is the scale at which time passes. This can be used for slow motion effects or to speed up your application. When the timeScale is 1.0, time passes as fast as real time. When the timeScale is 0.5, time passes 2x slower than realtime.

- Enable Time Scale - Check this box and the timer will calculate Start and End speeds over the duration of your timer
- Start Speed - This is the initial % of speed at which time passes in your project when the timer is activated. 1 = 100% (real time), 0.5 = 50% (half speed), etc.

- End Speed - This is the final % of speed at which time passes in your project when the timer has reached completion.

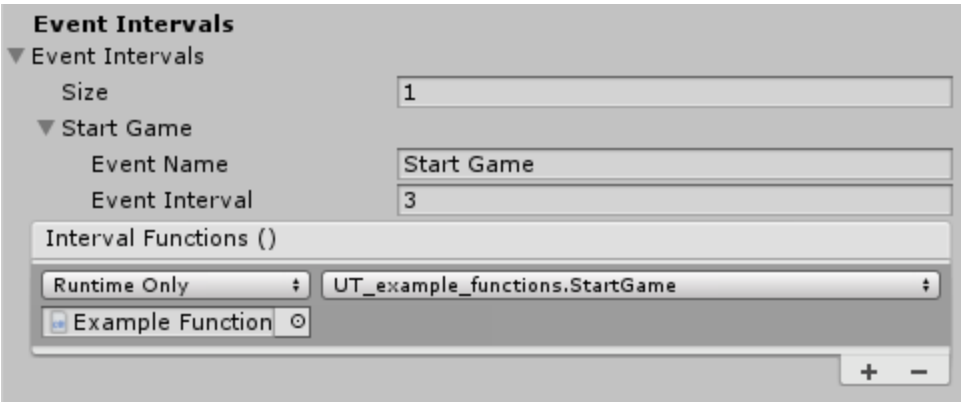


Texture

Update textures, sprites, and materials at custom intervals. In the example above, a brick material will be applied to the Sphere game object after five seconds pass.

- Texture Type - Set Sprite, Set Image, Set Material
- Obj with Texture - This is the Game Object that will be updating textures on
- Texture Intervals - The array of intervals for swapping textures
- Size - How many intervals there are
- Interval Element
    - Texture Name (optional) - Use this field to name the texture for this interval in the example above, it's been named *Update to Brick*
    - Texture Change Interval - The amount of seconds passed since the timer started
    - Texture Material - Targets a Game Object's material to swap
    - Texture Image - Targets a Game Object's material's main texture (Albedo) to swap
    - Texture Sprite - Target's the sprite file of an Image component from a Canvas Game Object to swap

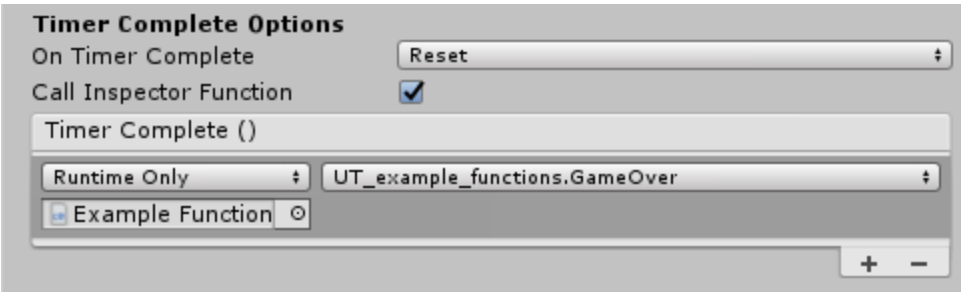
Event Intervals



Event Intervals

Set timed intervals to call custom events. In the example above, the interval titled *Start Game* will execute a function three seconds after the timer starts. Add as many intervals and functions as you like.

Timer Complete Options



Call Built-In or Custom Functions when Timer Completes

Ultimate Timer has 7 built-in functions that can be called when a timer reaches its end:

- On Timer Complete
    - Stop - Timer stops at the end of it's cycle and does not reset
    - Reset - Sets timer and outputs back to original state of 0 and pauses timer
    - Restart - Sets timer and outputs back to original state and plays the timer again
    - Disable Script - Disable the Ultimate Timer script until re-enabled
    - Disable Game Object - Disables the game object that the Ultimate Timer script is on
    - Destroy Game Object - Permanently removes the game object that the Ultimate Timer script is on while in play mode.
    - Call Custom Functions
- Call Inspector Function - Check this if you plan to call a custom function on timer completion
- Timer Complete() - Press + to add scripts into the field on the left and use the drop down to the right to select any public functions from that script

Public Functions

Ultimate Timer can be controlled from outside scripts in order to set desired states for different circumstances.

- SetDuration(float *duration*)
    - Sets total duration of timer (in seconds)
- IncreaseTime(float *time added*)
    - Adds time to the current timer (in seconds)
    - The latest event or texture interval will trigger if the increase spans over multiple
- DecreaseTime(float *time removed*)
    - Removes time from the current timer (in seconds)
    - Past texture or event intervals will not execute again
- Start Timer()
    - Enables game object and starts timer
- Pause Timer()
    - Pauses timer progress
- ResumeTimer()

    - Resumes timer progress
- ResetTimer()
    - Resets and auto pauses timer
- RestartTimer()
    - Resets and auto plays timer
- Clear Timer()
    - Resets timer listens for pause value
- TimeCompleted()
    - Forces timer to complete.
    - End of timer functions will execute
- DisableGameObject()
    - Disables parent game object
- DestroyGameObject()
    - Destroys parent game object

Supporting Scripts

UT_restart_timers

- A batch action script that carries out the Clear Timer() function on any Ultimate Timer scripts added to this array on enable. This is used to reset certain timers when the user pages back and forth through the timer demo examples.

UT_example_functions

- Simple pre-made functions that carry out a console Debug.Log messages for testing

UT_timescale_manager

- This script will speed up or slow down the overall time scale of your project.

Tips to Get Started

Make sure your existing project is free of any console errors

Sometimes errors from other scripts will affect the performance of this asset. Please be sure that there are no errors prior to attempting to debug the script.

Interval Order Matters!

When using Event and Texture Intervals, be sure to set them in the order they appear. If you declare an event 5 seconds in, then another event at 3 seconds in after that, it will produce unexpected results.

Chain Multiple Timers Together

Any timer can rely on another external timer. This allows for interesting results giving you the flexibility to manipulate countless objects based off a single parent timer. See timer options for more.

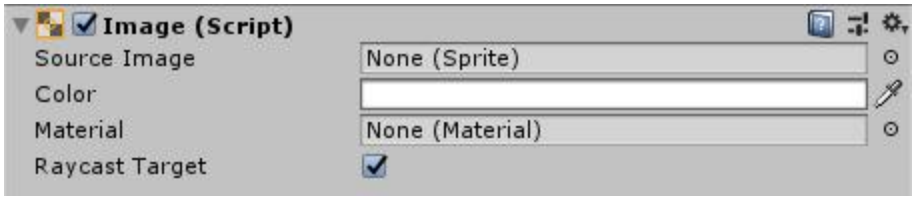
The Demo Looks Broken

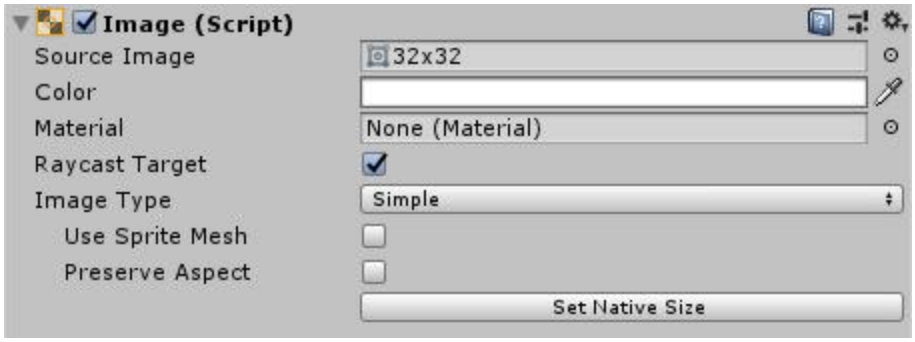
The demo is optimized for a 1920x1080 viewing experience. If you're having issues viewing the demo, check your project resolution and update.

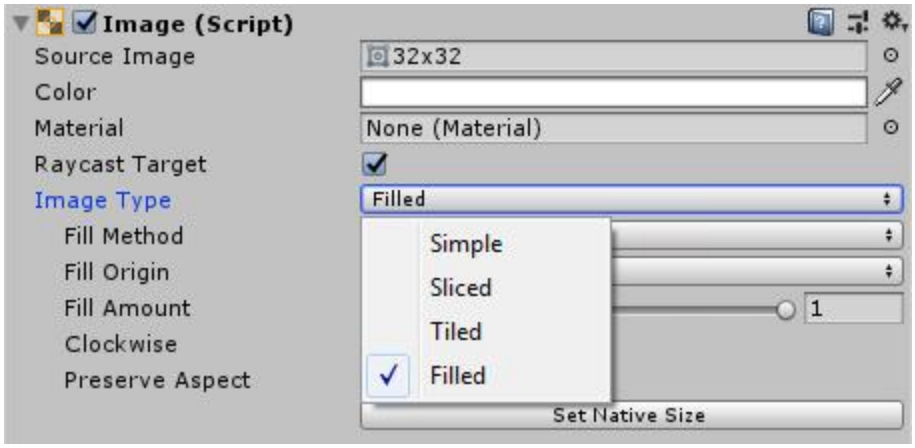
The demo was created in Unity 2020.3.5f1 and may not render well in earlier versions. If you'd like to view the demo click here.

Using Image Fills

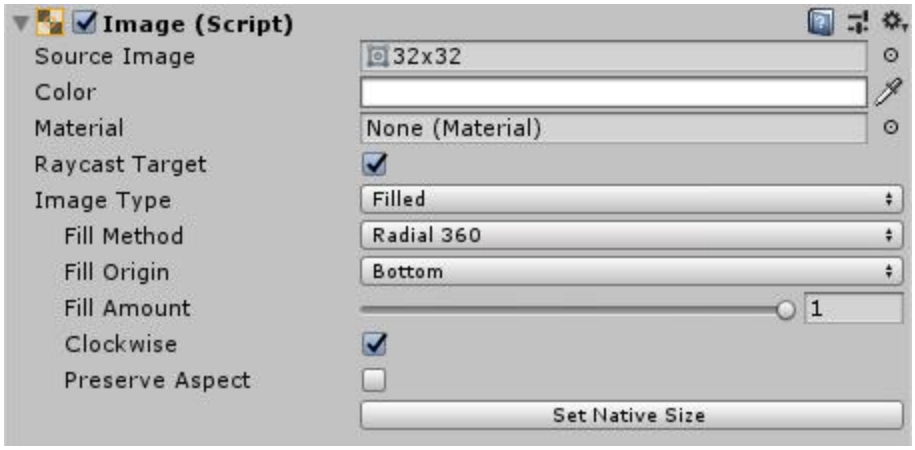
Using image fills is a great visual way to show progress. Here's how:

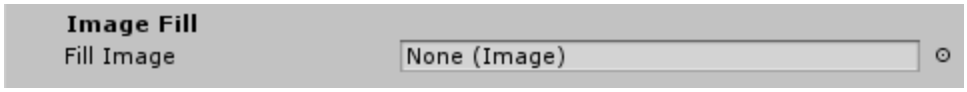
1. Right-Click Hierarchy window and select UI > Canvas
2. Highlight Canvas and Right-Click to select UI > Image
3. 

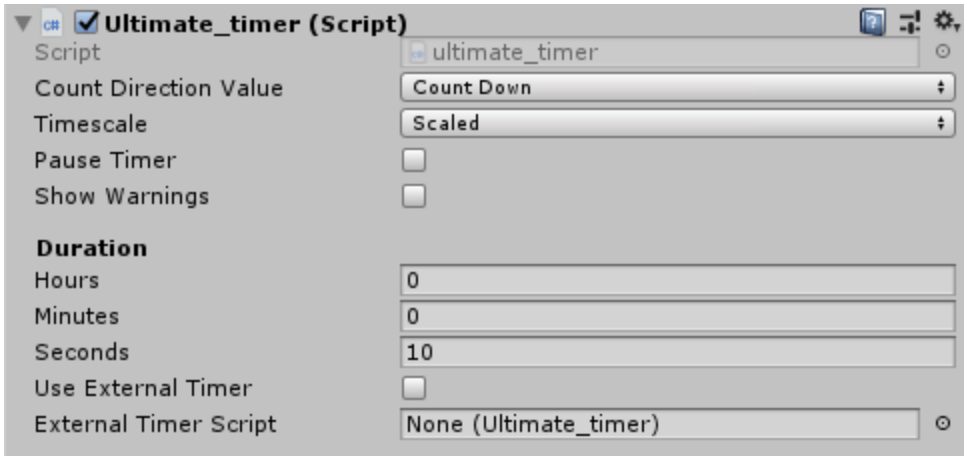
4. Under Source Image, select or drop any optimized sprite into that field
5. 

6. In this case, I'm using an optimized sprite cropped to 32x32
7. 

8. Click the drop down next to Image Type and select "Filled"

9. 

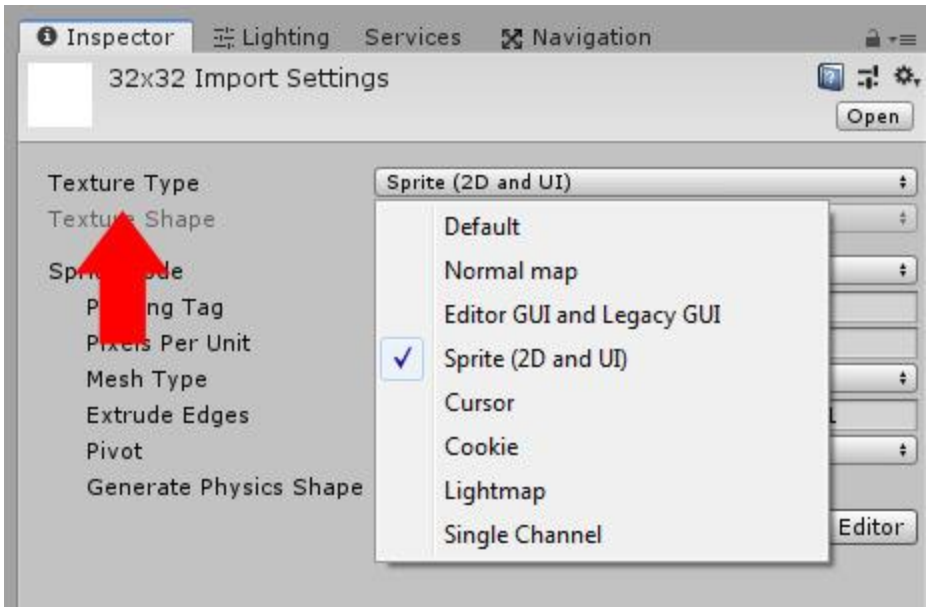
10. Under “Fill Method” you will have various options to fill your image.
11. 

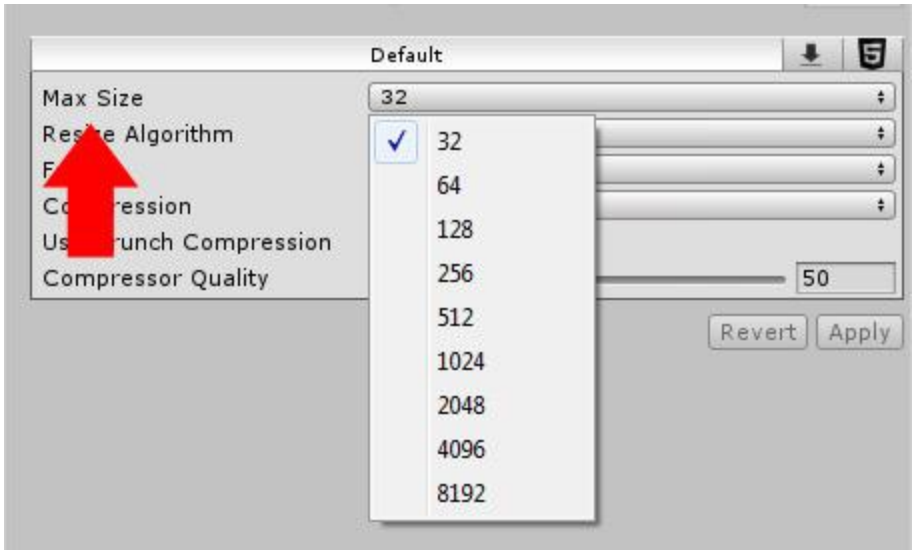
12. Drag your image into the “Fill Image” field in the Ultimate Timer script.
13. 

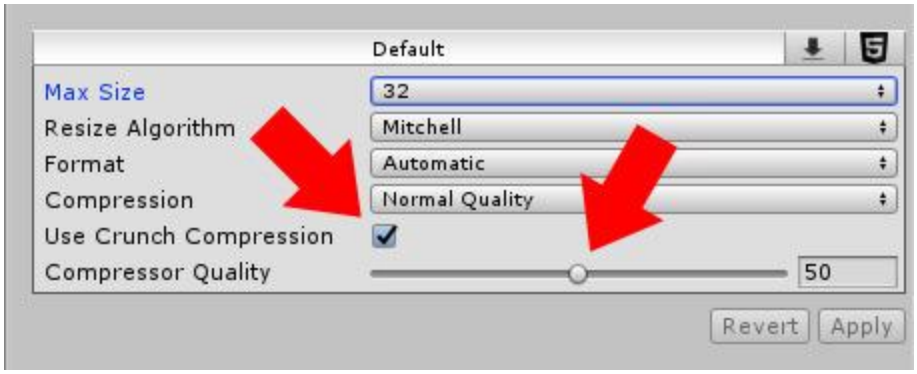
14. Make sure you set a duration and hit play.

Optimizing Sprites

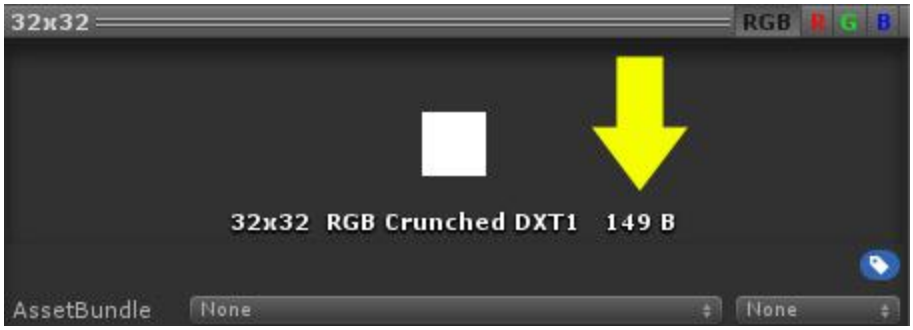
Optimize your sprites drastically without losing visual quality by using Crunch Compression. In order to take advantage of this, your image must be cropped by a power of 8. So 8x8, 16x16, 32x32. Width and height do not need to match, you can also use 32x16 or 256x1024 and so on.

1. 

2. Highlight your image and select Texture Type > Sprite (2D and UI)
3. 

4. In the Compression box, select Max Size and choose the maximum size of your image by width or height. If your image is 512x128, select 512
5. 

6. Check “Use Crunch Compression” and select a quality of 50

7. 

8. Check the preview window below to see the new sprite size

FAQ

The Demo Looks Broken

The demo is optimized for a 1920x1080 viewing experience. If you're having issues viewing the demo, check your project resolution and update.

The demo was created in Unity 2020.3.5f1 and certain components (like UI > Image) may not render in earlier versions. If you'd like to view the demo click here.

Try viewing the demo in Unity 2020.3.5f1 if you'd like to experiment with it..

Although the demo may not be rendering properly, rest assured, the asset itself works across older versions of unity as advertised on the store page.

Contact

If you're having any issues at all with this script, please contact me. I'm not happy unless you are! Also, if the script is working out for you and you're super satisfied, I'd love to hear from you! Click the contact link below to send me a message.

peter.tracy@yahoo.com

@studio11508 | http://ptracy.com/