Ultimate Text Damage

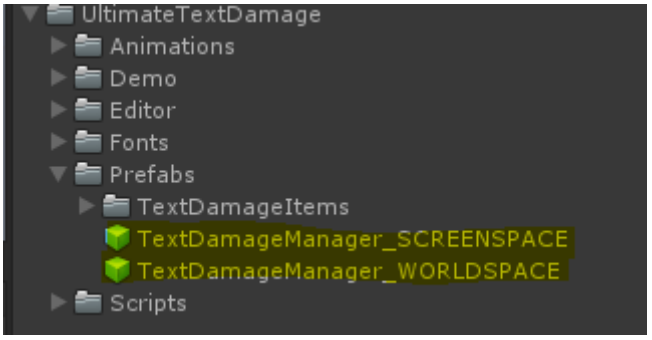
UPDATE 1.1.0

Old Unity Text is now obsolete, TextMeshPro is recommended for optimal results.
Be sure to **install TextMeshPro through the package** manager and **enable 'use text mesh pro'** in the UltimateTextDamage **preferences**. Please see further down the how to use the asset with TextMeshPro section.

Thank you for purchasing Ultimate Text Damage. uTextDamage uses a pool system for all the floating text, each different text is a prefab that needs to be assigned to the TextDamageManager.

Here's a quick start guide on how to set it up.

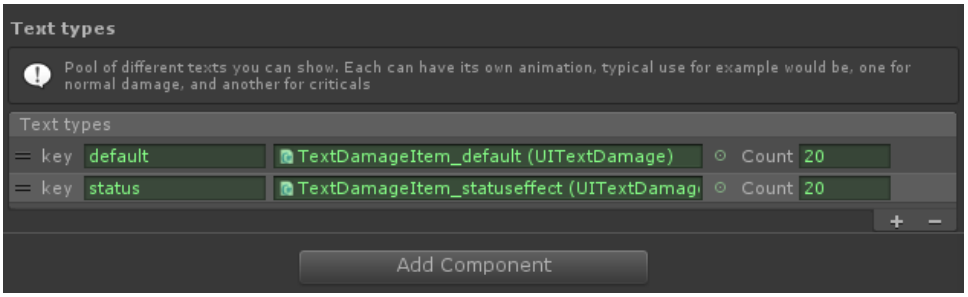
1. **Drag and drop** on the scene the **TextDamageManager** (World space, Screen space or UI). Note that you can have multiple managers in the scene.



2. **Set up the Canvas and Camera references**. The Camera reference will be used only when the "convert from camera" option is enabled. See example scenes to select the best configuration for your goal.

3. **Add the UITextDamage items** that will be used in the list. You can add as many items as you want, just be sure **NOT to duplicate any key**. Each item can have its new visuals and animation (see the section below on how to customize UITextDamage items).

By default the number of instances of the pool for every item is 20, you can change this to any number. just to note that if the pool runs out of available items a new item will be instantiated.



4. **Call the method** from an instance of UltimateTextDamageManager.Add with the **text**, a **transform** that will act as where it will show and the **key** to use. For example:

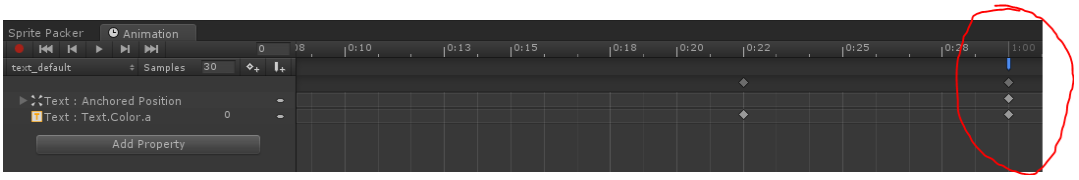
```
textManagerInstance.Add( damage , transform , "critical" );
```

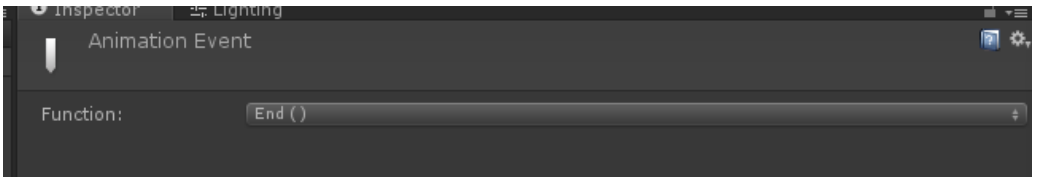
It is also possible to pass a position instead of a transform, however, the ‘no overalping’, ‘follow target’ and stacking features will not work.

```
textManagerInstance.Add( damage , transform.position , "critical" );
```

Customizing items

You can customize **UITextDamage** items to change its visuals or the animation. Simply change the Text or TextMeshProUGUI component. You can create new animations and animation controllers and assign them to the new item, just make sure the animation has an AnimationEvent at the end that calls the **End( )** method. This will allow the manager to know that the item can be used again.

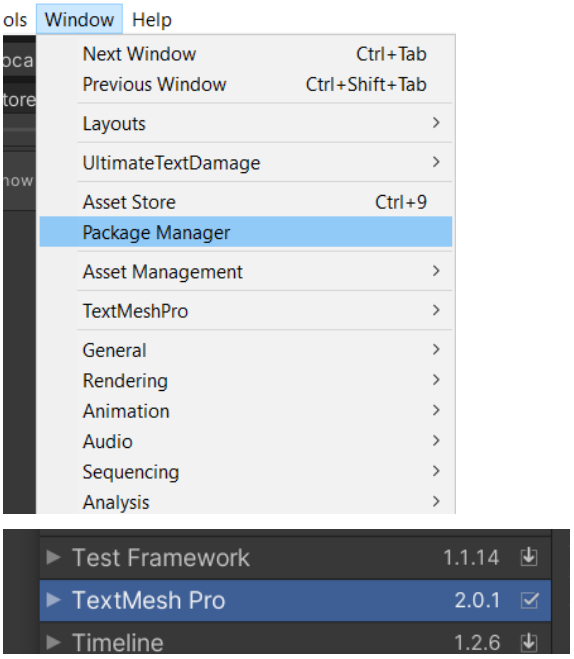




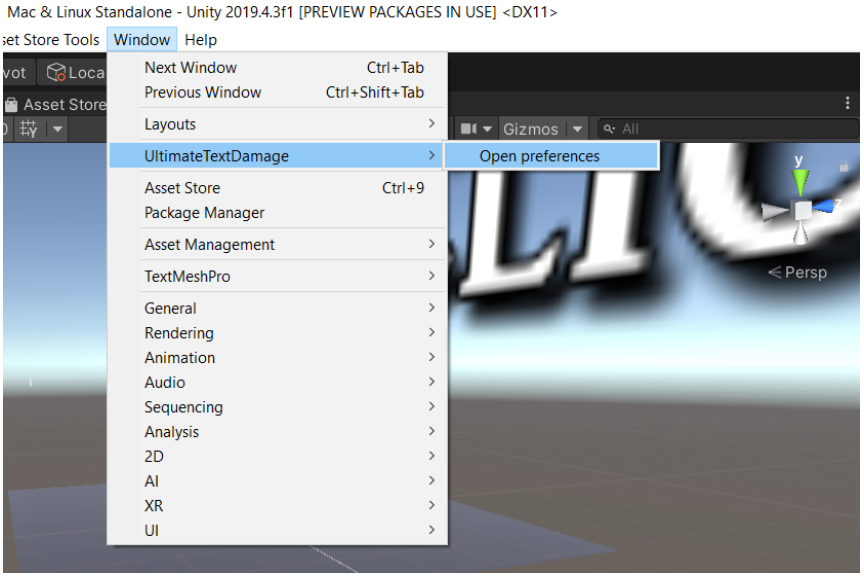
Using TextMeshPro

Since version 1.1.0 TextMeshPro is the recommended way.

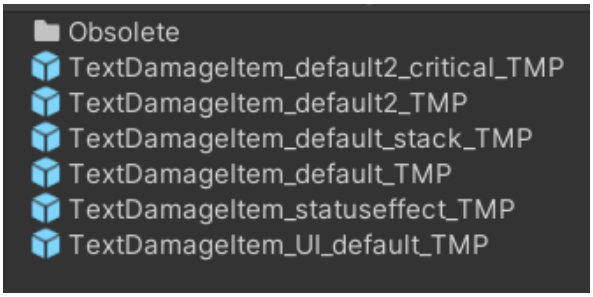
1. If you haven't already, through the Unity package manager install TextMeshPro.



2. Open the UltimateTextDamage preferences window and enable 'Use TextMeshPro'.



3. After enabling ‘Use TextMeshPro’ all current TextDamage items using the Unity Text’ won’t work. You will have to modify your current items to support TextMeshPro or use existing ones inside the Prefabs folder ending in ‘_TMP’ (see the demo scenes).



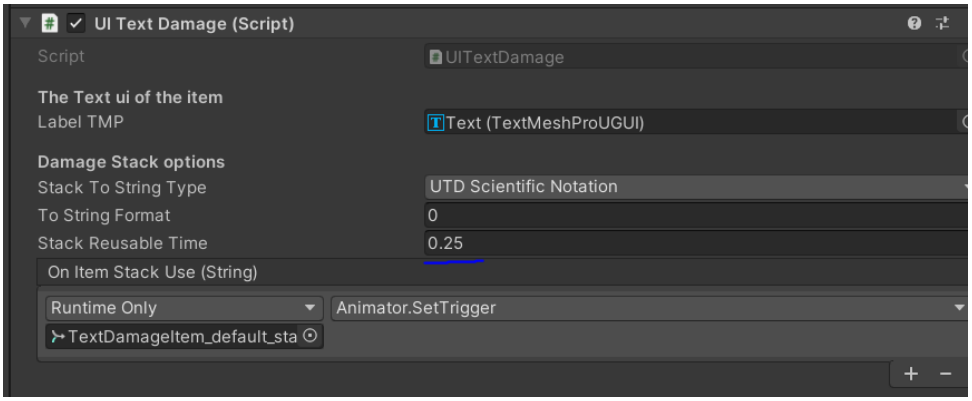
Stacking damage numbers

Stacking damage text is now supported, giving you the option to show the total damage done in a period of time instead of each individual one for a particular target. A good example is the multiplayer game Apex Legends.

To use stacking numbers call the following method instead.

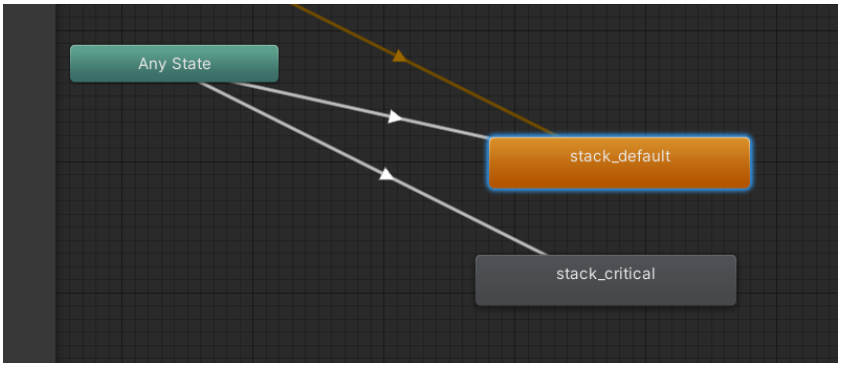
textManagerInstance.AddStack(damage, transform , “normal”);

Notice that **damage** is now a number, this is because internally it will add and accumulate all the damage until a certain amount of time passes, once this time passes, a new text element will be used and the accumulated total damage will start from zero.
You can specify the time that an item will be unable to be used again for the next call in the TextDamage prefab element field **stackReusableTime**. For example, a stackReusableTime of 0.25 seconds means that this element will be reused if the AddStack gets called again within less than 0.25 seconds.



In the **demo_Stack** scene, there’s only one text damage element that is used for both default and critical hits. Once a stack element is used it will invoke the Unity event ‘OnItemStackUse’ specifying the **stackKey** used in the AddStack method. This can be used for visually changing the text within the

same TextDamage element. In the example image above, we use the stack key “normal” or “critical” to trigger one animation or another in the animation controller.



If you have any technical issues or suggestions feel free to mail them at guirimarcunity@gmail.com