**Spis treści**

[***Introduction 2***](#_Toc503454874)

[***Tooltip Designer 2***](#_Toc503454875)

[***Tooltip Creator 3***](#_Toc503454876)

[***Observer 3***](#_Toc503454888)

[***Tooltip Layout 4***](#_Toc503454890)

# **Introduction**

***This tool is compatible with TextMeshPro only !***

***using TP.Tooltip; // If you want to use some references to Tooltip you need to add it.***

***Tooltip Creator is manager for whole functionality, you want only single instance of it.***



***Correct TooltipLayout Hierarchy is as on example:***

***Script is looking for all TextMeshProUGUI, Image and Button which have their parent and their parent are child of TooltipPanel.***

***TooltipCanvas is just a canvas, its needed to render tooltip.***

***Images, Buttons, Texts are important Transform. You must have different parents for Images, Buttons and Texts. They are containers for them.***

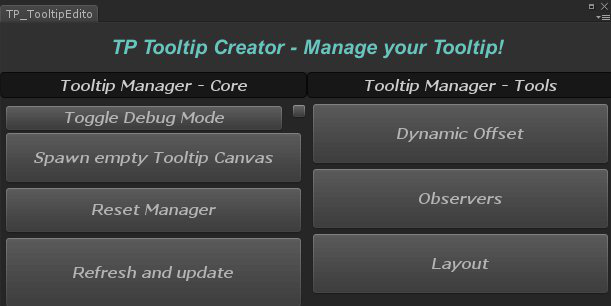
***Observer – GameObject with with this script will execute events if IsObserving boolean is true.***

***TooltipEditorGUIData is located in “Assets/TP\_Creator/TP\_TooltipCreator/EditorResources/TooltipEditorGUIData.asset”***

***You can change:***

***Empty Tooltip Hierarchy Prefab – prefab of example of tooltip’s hierarchy.***

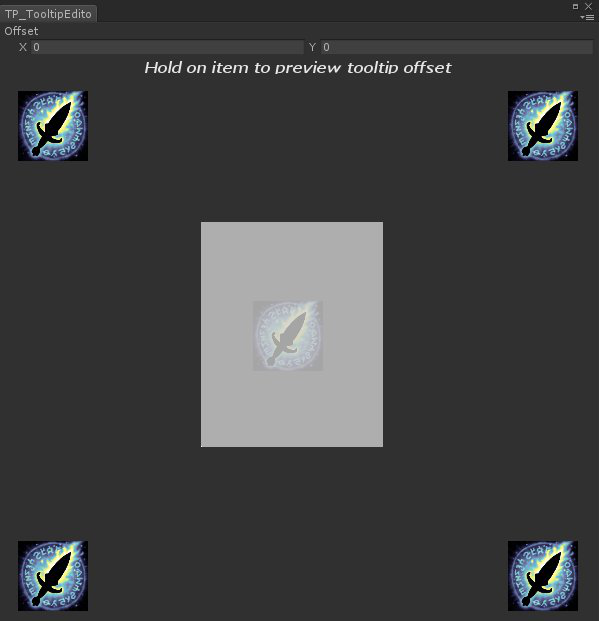
# **Tooltip Designer**

***Spawn Empty Tooltip Canvas – it spawns empty example of tooltip hierarchy.***

***Reset Manager – it resets reference to TooltipCreator – manager.***

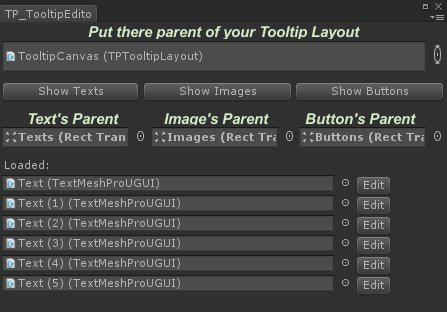
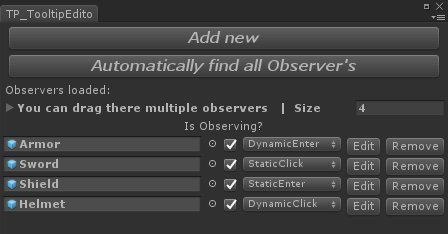
***Refresh and update – it refreshes references if you’ve made some changes and they haven’t updated.***

***Toggle Debug Mode – Toggles Debug Mode, you can’t make any changes, but you can see everything serialized in inspector***

***Dynamic Offset – it opens new editor window for preview of dynamic offset.***

***Observers – it opens new editor window to manage all your observers.***

***Layout – it opens new editor window to manage your Layout.***



# **Tooltip Creator**

## ***TPTooltipCreator creator;***

## ***creator.Observers; // Gets List of Observers;***

## ***creator.Offset; // Get/Set dynamic tooltip offset***

## ***creator.OnObserver; // Gets Observer which is actually tooltip on.***

## ***creator.StaticTransform; // Get/Set transform for tooltip for static observers***

## ***creator.TooltipLayout; // Get/Set TooltipLayout***

## ***creator.Refresh; // Refreshes list and referencesh on Creator***

## ***creator.SetTooltipActive(…); // Set TooltipLayout true/false***

## ***creator.SetOnActive(…); // Set delegate void - it's executing before Enter/ExitObserver and allows you to make custom behaviour on changing activtion of tooltip***

## ***creator.SetOnEnterObserver(…); // Method that allows you to add your own behaviour, it's executing just before tooltip is activated***

## ***creator.SetOnExitObserver(…); // Method that allows you to add your own behaviour, it's executing just before tooltip is deactivated***

# **Observer**

***TPTooltipObserver observer;***

***bserver.IsObserving; // If it returns false, your observer will not execute any events***

## ***observer.SetType; // Observer type (enum);***

# **Tooltip Layout**

## ***TPTooltipLayout layout;***

## ***layout.Refresh; // Refreshes Texts, Images and Buttons***

## ***layout.PanelTransform; // Gets Tranform of panel (fist child of canvas)***

## ***layout.Texts; // List of Texts - must be TextMeshPro***

## ***layout.TextsParent; // Transform - parent of all texts***

## ***layout.Images; // List of Images***

## ***layout.ImagesParent; // Transform - parenf of all images***

## ***layout.Buttons; // List of Buttons***

## ***layout.ButtonsParent; // Transform - parent of all buttons***

## ***layout.GetText(…); // Get Text on index (2 overloads)***

## ***GetText(index)***

## ***GetText(params int[] indexes)***

## ***layout.GetImage(…); // Get Image on index (2 overloads)***

## ***GetImage(index)***

## ***GetImage(params int[] indexes)***

## ***layout.GetButton(…); // Get Button on index (2 overloads)***

## ***GetButton(index)***

## ***GetButton( params int[] indexes)***

## ***layout.SetText(…); // Sets text of specific TextMeshProUGUI (5 overloads)***

***SetText(string text, int index)***

***SetText(string text, params int[] indexes)***

***SetText(string[] texts, params int[] indexes***

***SetText(string text, TextMeshProUGUI Text***

***SetText(TextMeshProUGUI[] TextMeshes, params string[] texts)***

## ***layout.SetImage(…); // Sets sprite of specific Image (5 overloads)***

***SetImage(Sprite sprite, int index)***

***SetImage(Sprite sprite, params int[] indexes)***

***SetImage(Sprite[] sprites, params int[] indexes)***

***SetImage(Sprite sprite, Image image)***

## ***SetImage(Image[] images, params Sprite[] sprites)***

## ***layout.SetButtonClick(…); // Sets listener onClick on specific Button (5 overloads)***

***SetButtonClick( UnityEngine.Events.UnityAction action, int index)***

***SetButtonClick(UnityEngine.Events.UnityAction action, params int[] indexes)***

***SetButtonClick(UnityEngine.Events.UnityAction[] actions, params int[] indexes)***

***SetButtonClick(UnityEngine.Events.UnityAction action, Button button)***

## ***SetButtonClick(Button[] buttons, params UnityEngine.Events.UnityAction[] actions)***