**Spis treści**

[***Introduction 2***](#_Toc503030384)

[***Inventory Designer 3***](#_Toc503030385)

[***Inventory Creator 4***](#_Toc503030386)

[***Item 4***](#_Toc503030397)

[***Slot 5***](#_Toc503030408)

[***Modifier 5***](#_Toc503030418)

[***Stat 5***](#_Toc503030426)

[***Sound 6***](#_Toc503030430)

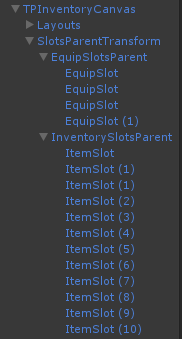
[***Type 6***](#_Toc503030434)

# **Introduction**

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

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

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



***Script is looking for all slots which have their parent and their parent are child of SlotsParentTransform.***

***TPInventoryCanvas is just a canvas, its needed to render images of inventory.***

***Layouts is pseudo-folder for responsible for the appearance of inventory.***

***SlotsParentTransform is very important Transform. You must have different slot’s parents in it. EquipSlotsParent and InventorySlotsParent are pseudo-folders which contains all slots.***

***Slots – they are empty containers(images) for items, so you need to make a mask(simply slot layout) if you want your slots look like slots.***

***Items, Types and Stats are ScriptableObjects which you can easily edit on inspector(or on designer).***

***Sound is struct which contains AudioType(enum) and AudioClip.***

***Type is ScriptableObject and it hasn’t any special functionality, just to specife type.***

***Modifier is class which has two methods – Modify() and UnModify() ant it has reference to Stat, ModifierType (Percentage or Flat), ModifyCommand(Increase or Decrease) and Value - float to changing Stat’s Value***

***InventoryData is ScriptableObject which keeps all inventory’s data – lists of Items, Stats and Types.***

# **Inventory Designer**

***Change Parent – it allows you to change SlotsParentsTransform – it’s needed to load automatically slots and add to them required components.***

***Spawn Empty inventory hierarchy – it spawns empty example of correct inventory hierarchy.***

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

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

***Items – it opens new editor window to manage all your items.***

***Types – it opens new editor window to manage all your types.***

***Stats – it opens new editor window to manage all your stats.***

***Slots – it opens new editor window to manage all loaded slots.***

# **Inventory Creator**

## ***TPInventoryCreator creator;***

## ***creator.SlotParentsTransform; // Transform of parent's slot***

## ***creator.Slots; // A List<TPSlot> of all slots loaded in SlotParentsTransform***

## ***creator.Data; // Inventory Data ScriptableObject contains all data of inventory***

## ***creator.Data.Items; // A List<TPItem> of all loaded Items exist in project***

## ***creator.Data.Stats; // A List<TPStat> of all loaded Stats exist in project***

## ***creator.Data.Types; // A List<TPType> of all loaded Types exist in project***

## ***creator.RefreshSlots(); // Refresh slots in SlotParentsTransform;***

## ***creator.IsFull(); // Returns true if there is no more free slots***

## ***creator.IsFullOfType(TPType type); // Returns true if there is no more free slots of type;***

## ***creator.FreeSlotsLength; // Returns length of free slots;***

## ***creator.FreeSlotsOfTypeLength(TPType type); // Returns length of free slots of type;***

## ***creator.FindAnyFreeSlot(); // Returns TPSlot, if there exist free one;***

## ***creator.FindAnyFreeSlotWithType(TPType type); // Returns TPSlot, if there exist free one of type;***

## ***creator.FindFreeEquipSlot(); // Returns TPSlot, if there exist free one and is equip slot;***

## ***creator.FindFreeEquipSlotWithType(TPType type); // Returns TPSlot, if there exist free one of type and is equip slot;***

## ***creator.FindFreeNoEquipSlot(); // Returns TPSlot, if there exist free one and is not equip slot;***

## ***creator.FindFreeNoEquipSlotWithType(TPType type); // Returns TPSlot, if there exist free one of type and is not equip slot;***

***creator.FindAnySlotWith(TPItem item); // Returns TPSlot with item***

***creator.FindNoEquipSlotWith(TPItem item); // Returns TPSlot which is not equip slot with item***

***creator.FindEquipSlotWith(TPItem item); // Returns TPSlot which is equip slot with item***

***creator.AddItem(TPItem item); // Add item to inventory, it’ll find any free slot***

***AddItem(TPSlot slot, TPItem item);***

***creator.RemoveItem(TPItem item); // Remove item from inventory, it’ll find first slot with this item***

***creator.RemoveItem(TPSlot slot) // Remove item from inventory from specific slot***

***creator.SetOnBeforeAddItem(); // Sets delegate event which is called before Add Item***

***creator.SetOnAfterAddItem(); // Sets delegate event which is called after Add Item***

***creator.SetOnBeforeRemoveItem(); // Sets delegate event which is called before Remove Item***

***creator.SetOnAfterRemoveItem(); // Sets delegate event which is called after Remove Item***

## 

# **Item**

## ***TPItem item;***

## ***item.Description; // Gets Item's description string;***

## ***item.ID; // Gets unique ID of item.***

## ***item.Modifiers; // Gets Array of Item's Modifiers***

## ***item.Sounds; // Gets Array of Item's Sounds***

## ***item.Name; // Gets Item's name string***

## ***item.Sprite; // Gets Item's sprite***

## ***item.Type; // Gets Item's Type***

## ***item.Worth; // Gets Item's worth;***

## ***item.OnSlot; // Gets Slot if it is on any.***

# **Slot**

## ***TPSlot slot;***

## ***slot.Type; // Gets Slot's Type;***

## ***slot.IsEquipSlot; // Returns true if it's equip slot***

## ***slot.Item; // Gets Item which is on this slot***

## ***slot.OnBeginDrag; // virtual Unity's Event, so you can override it.***

## ***slot.OnDrag; // virtual Unity's Event, so you can override it.***

## ***slot.OnEndDrag; // virtual Unity's Event, so you can override it.***

## ***slot.OnPointerClick; // virtual Unity's Event, so you can override it.***

## ***slot.OnPointerEnter; // virtual Unity's Event, so you can override it.***

## ***slot.SetOnSelection; // Sets delegate void which executes after slot is selected(is it’s selectable).***

## ***slot.IsSelected; // returns true after first click on slot.***

***slot.IsSelectable;*** ***// if true script’ll equip after second click***

# **Modifier**

## ***TPModifier modifier;***

## ***modifier.ModifyCommand; // Gets value of enum: Increase or Decrease***

## ***modifier.ModifyType; // Gets value of enum: Percentage or Flat***

## ***modifier.Value; // Value of modifier (to change Stat)***

## ***modifier.Stat; // Gets Stat which this modifier changes***

## ***modifier.Modify(); // Modify Stat***

## ***modifier.UnModify(); // Undo for Modify();***

# **Stat**

## ***TPStat stat;***

## ***stat.Value; // Gets Stat's value;***

## ***stat.BeforeChange; // Delegate OnChange – executes before value is changed***

## ***slot.AfterChange; // Delegate OnChange – executes after value is changed***

# **Sound**

## ***TPSound sound;***

## ***sound.AudioClip; // AudioClip of sound***

## ***sound.AudioType; // Gets value of enum: GetItem, RemoveItem, WearItem, MoveItem, UseItem***

# **Type**

## ***TPType type; // It’s just a ScriptableObject***