DT INVENTORY

User guide

DT Inventory v1.0 | DarkTree Development 2020 | Support : darktreedevelopment@gmail.com

Content

- How it works
- Getting started with Inventory Wizard
- Components overview
- Basic scripting
- FAQ

*mobile supported by default

How it works

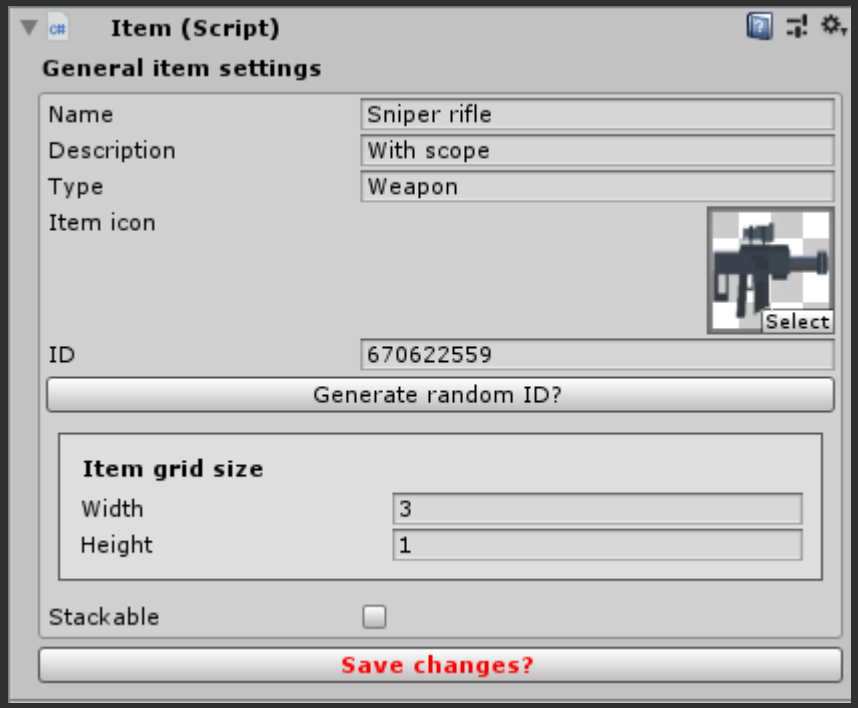
First of all, thank you for purchasing my asset! I sincerely hope that my work will help you to develop your games and save a lot of time for you. Inventory always was a sort of magic for me. There is almost no tutorials about how to make inventory with Diablo/S.T.A.L.K.E.R. grid placement. In this overview I want to give you a knowledge how this inventory works so that you understand what you are doing and there was no mystery left.

Inventory is a collection of grid cells with X and Y indexes with free/occupied bool state. It's all! When we trying to place an item to the grid we ask inventory next questions. Do we have enough space to store item? Where we can place item with specified width and height?

Each slot has own index in the grid. [x][y]

Each item has width and height. Width and height it is a space which item will take on a grid.

Here we have an item with a size of width 3 and size of height 1. Width 3 means that our item will take 3 grid cells on horizontal and 1 on vertical for height.



Since we know space required for item storing, we can tell inventory to find suitable cell for us.

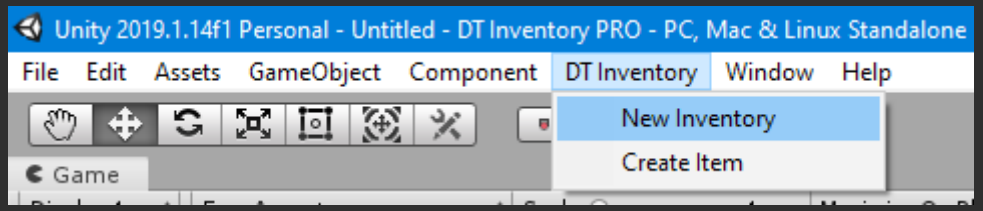
Inventory find cells with this construction x+width, y+height. For example, if inventory will get slot 0,0 as a starting point, and slots 0,0;1,0;2,0 will be free, we can store that item on a grid. Otherwise, if one of the cells is occupied by another item, then we will not be able to put the item there.

And finally, if we place item to the grid, inventory marks x+width, y+height cells as occupied. If we start dragging item to other position, we just mark these cells as free and mark new ones as not.

It's a brief explanation of how DT Inventory works. Of course we have a lot of other different methods but it's essential knowledge give you vision of inventory construction. You can meet with other utility methods in source code.

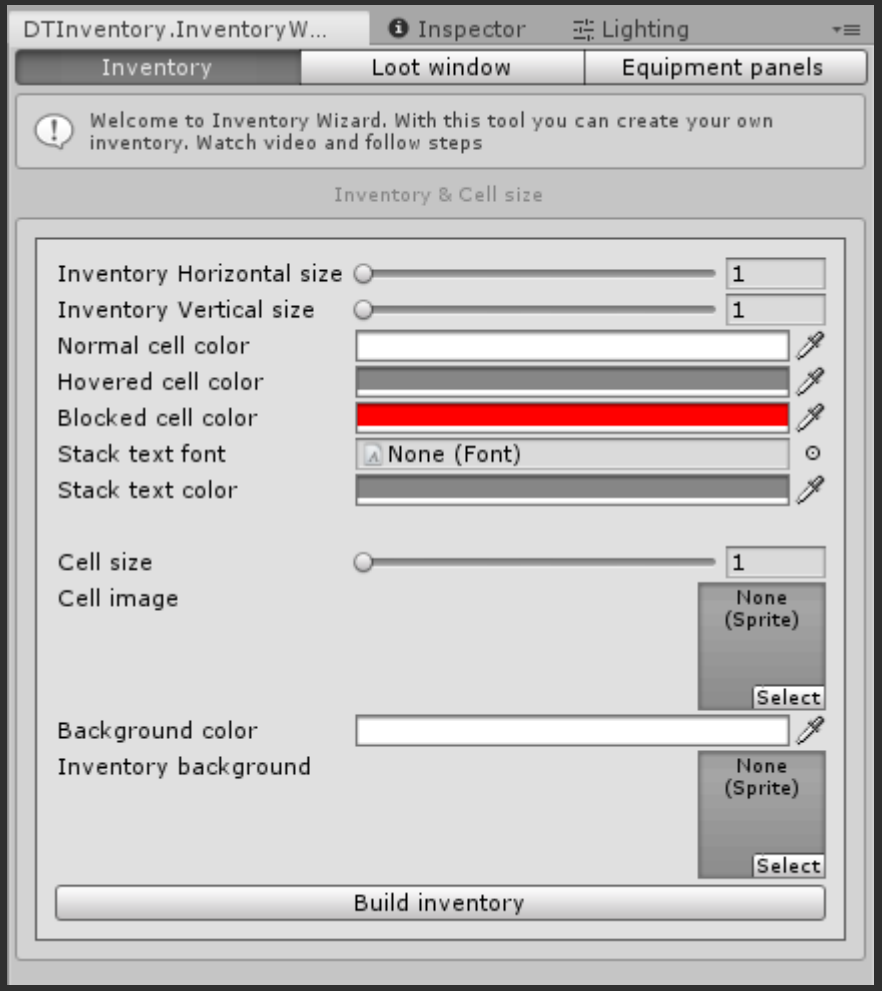
Getting started with inventory wizard

1) Open the Inventory Wizard with the toolbar at the top and press ‘New Inventory’



2) Set up fields of Wizard and press ‘Build Inventory’

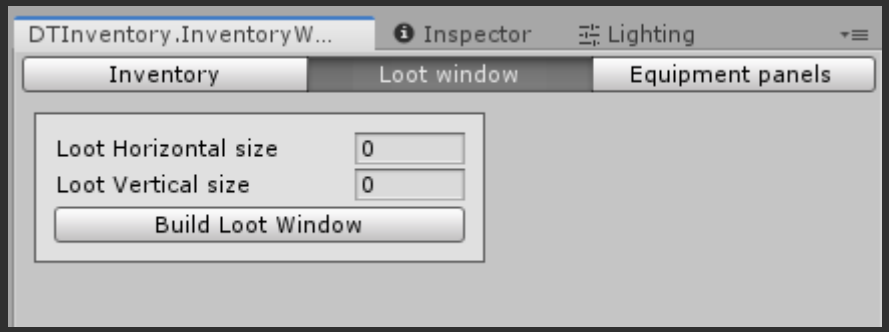
Inventory Horizontal size	Number of grid cell on vertical/horizontal axis of the grid
Inventory Vertical size	
Normal cell color	Set of colors to draw normal/hovered/ blocked state of the cell
Hovered cell color	
Blocked cell color	
Stack text font	Font of stack text drawer
Stack text color	Color of stack text
Cell size	Size of grid cell in pixels
Cell image	Sprite image to draw cell
Background color	Color of background panel
Inventory background	Sprite image for background panel



After you've created a new inventory you can also create a loot window and equipment slots. I'll tell about their functionality little bit later. So lets go on!

Loot window

4) Click on 'Loot window' tab and complete fields. Click 'Build Loot Window' as you will done. Loot Horizontal size and Vertical size are the same with inventory settings. It's an X and Y size of our grid. Set position of the window on screen.



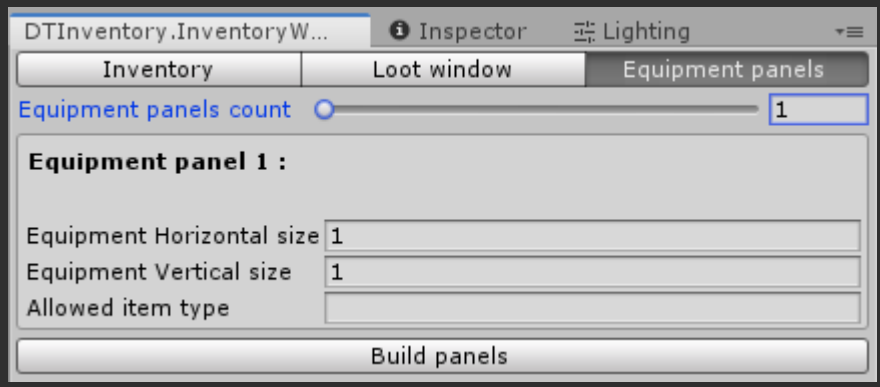
We made loot window but for a what? This window will be used as loot box/chest representation. It's a special grid that will contain items from chest and provide control on 'em for us.

Congratulations! You've made your own inventory from blank in a few minutes. Inventory is fully functional right now and it's working but who would to use equipment slots? Your game has armor and weapons? If yes you probably will be glad to know that equipment panels creation as simple as the actions above.

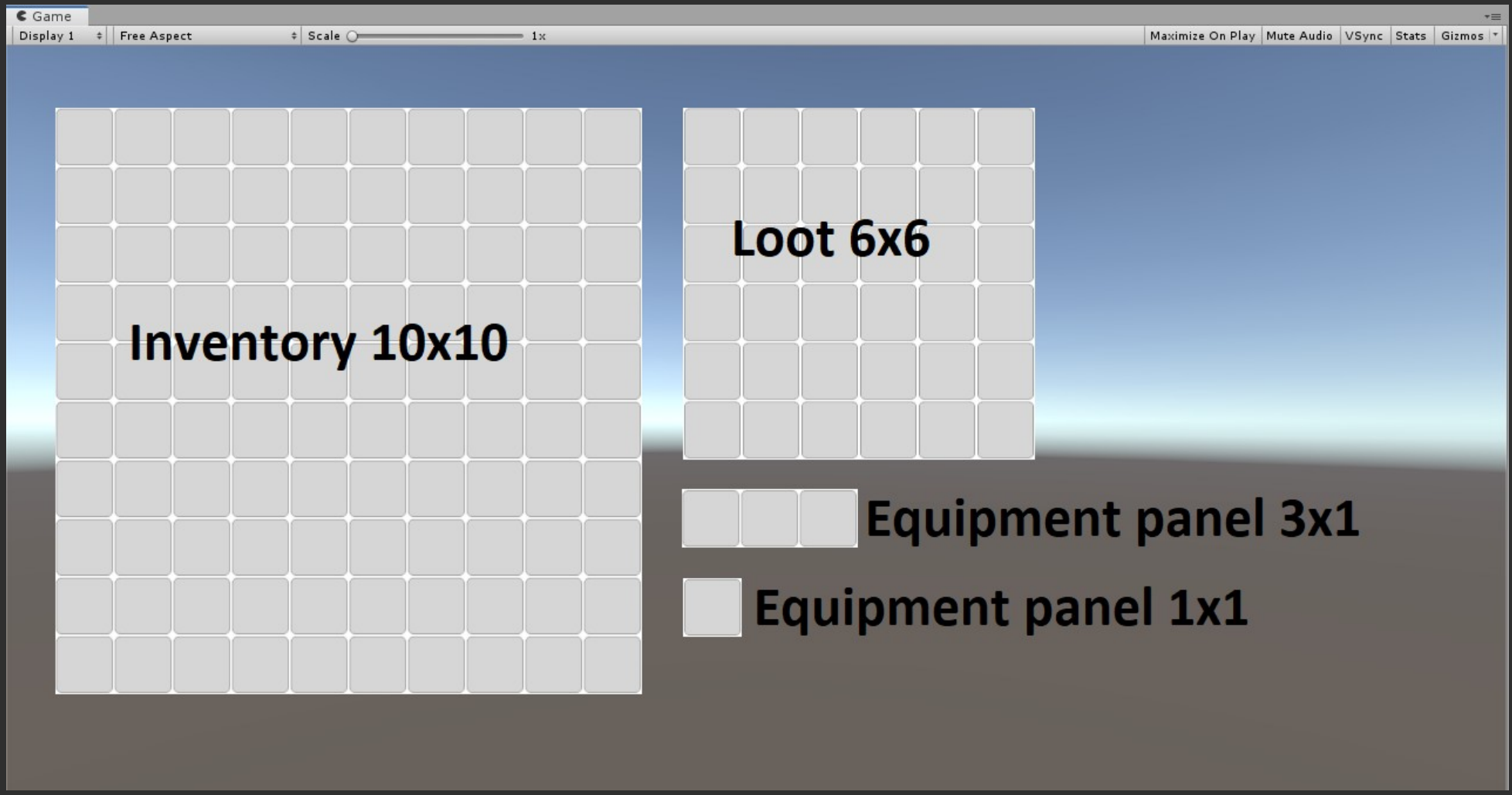
Equipment panels

5) Click on ‘Equipment panels’ tab and complete fields. Press ‘Build panels’ and set position of the equipment panels on the screen.

Equipment panels count	Number of equipment panels you want to create. You can setup panel for required item type
Equipment horizontal size	X grid size
Equipment vertical size	Y grid size
Allowed item type	Each Item has string field called ‘Type’. It’s like unique tag for different type of items. Here you can put allowed item type. Others are not will be equipped.



Do you see similar picture on your screen?

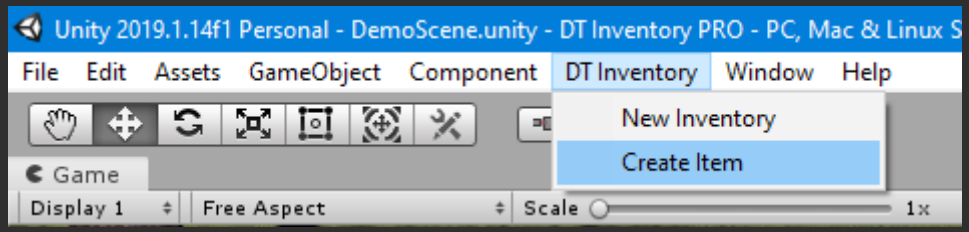


If yes, we continue with items and interactions. If no, something went wrong. You can try again or contact me at email below. Also look for video tutorial on YouTube.

Items

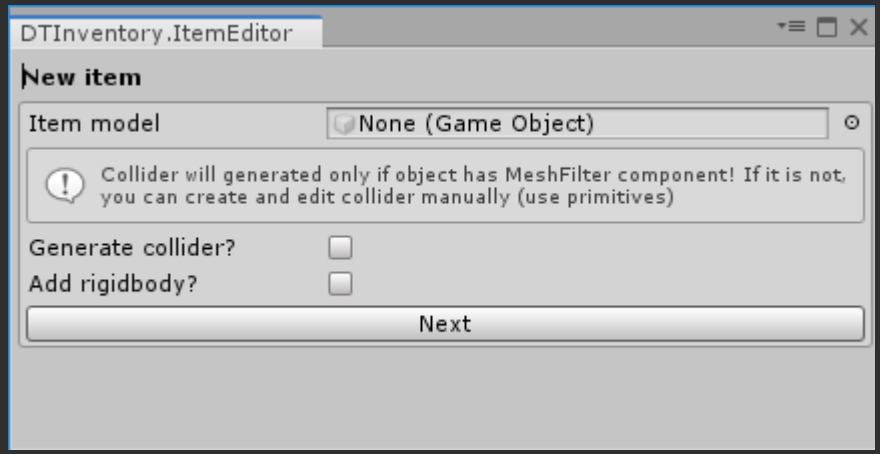
We can create items manually and with item wizard. Each item is a prefab with the Item.cs component. Item has to be created within scene and saved to prefab after.

1) Open the Item Wizard with the toolbar at the top and press ‘Create item’



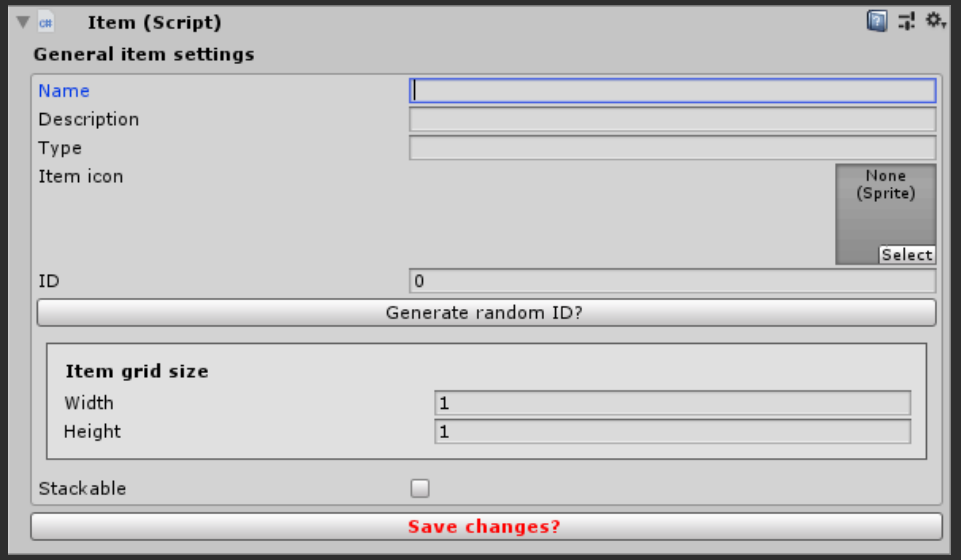
2) Complete fields and press Next

Item model	A 3d object we use as our item. Better to use single meshes
Generate collider	Should we create a collider for an object? If object has MeshFilter you will get convex mesh as result.
Add rigidbody	Should we create rigidbody for an object?



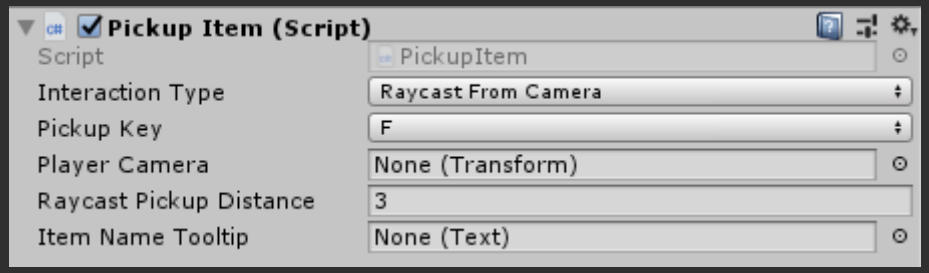
3) After previous step you will have new instantiated item in your scene. Our object has “Item” tag, and Item component. Your items must have “Item” tag! Otherwise pickup will not work at all. Let’s take a look at the Item component.

Name	Item name is the main item identificatior.
Description	Describes item with tooltip panel
Type	Special tag to store items in specified equipment panels
Item icon	Item icon to draw in inventory grid
ID	Unique item identifier
Item grid size - Width	How much space will item store on inventory grid on X axis
Item grid size - Height	How much space will item store on inventory grid on Y axis
Stack options	
Stackable	Can we stack this item with the same items?
Itemstack size	Current item stack size
Max stack size	Max value which stack size could take



Pickup system

You can pickup items with special component which need to be assigned to the main camera. It's PickupItems component. With this component you can provide interaction between player and items. The component is designed to introduce pickup interaction by the most part and can write your own pickup methods if you need.



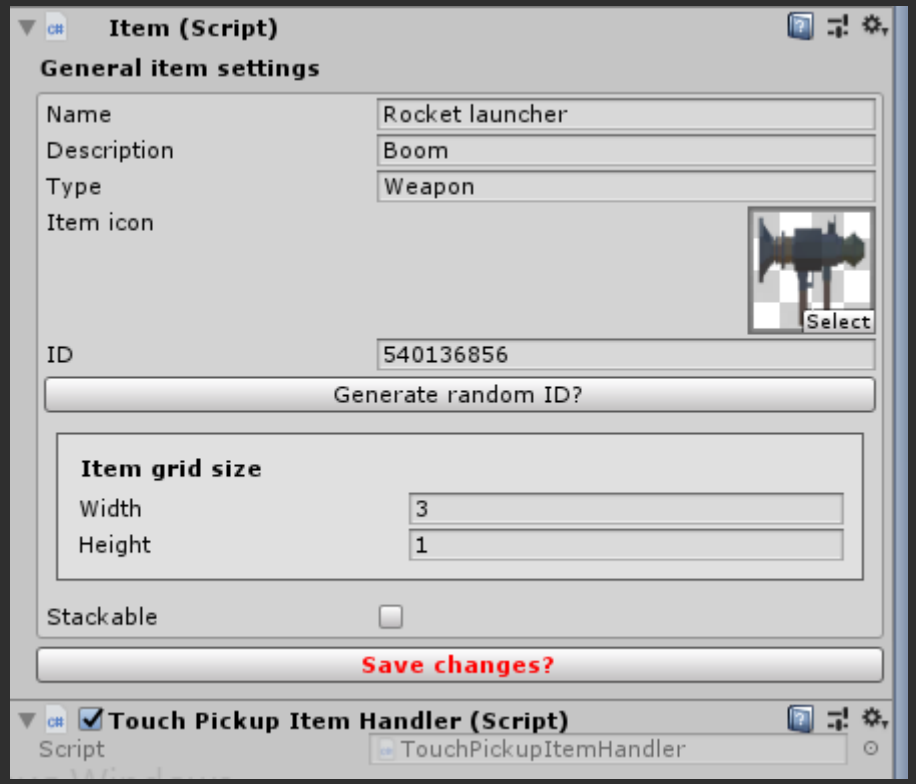
The component should be assigned to your player. Here you can choice interaction type. You can switch between Raycast from camera/Trigger pickup/Pickup by click. You can set pickup key with any other you want. You also need to set Player Camera field. Raycast Pickup distance controls distance for items pickup. And item name tooltip is the UI text field that we need to show item title if we looking to some item.

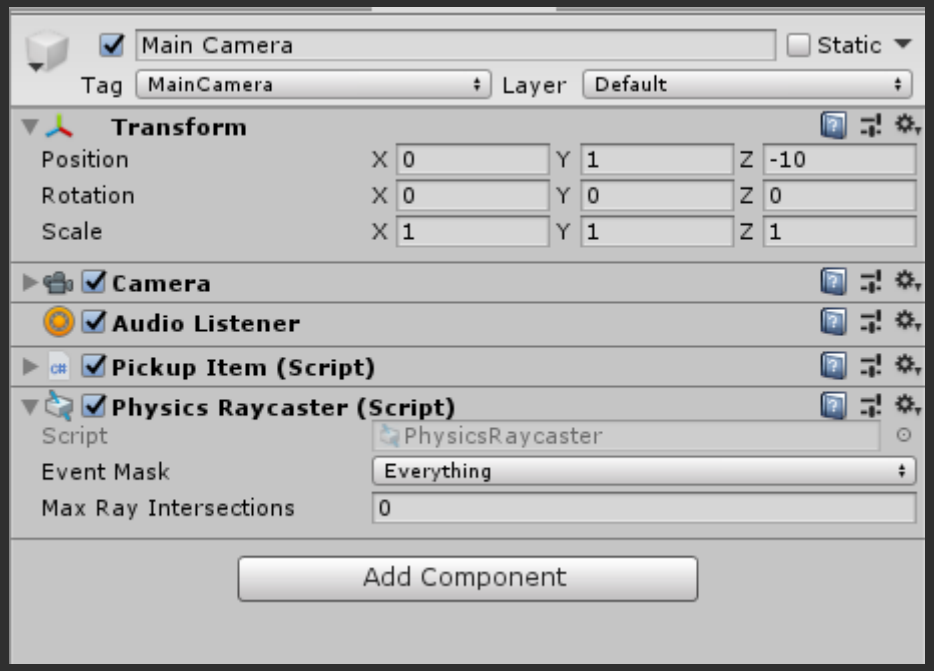
Now about interaction types more close.

Raycast from Camera – it's fps pickup style. You can pickup an item if you looking straight on it and distance between camera and item is equal or less to pickup distance.

Trigger pickup – in this case you need to have trigger collider on object that store pickup item component. After that you will able to pickup all items which in collider radius with pickup key button.

And last one is Click to Pickup. In order to get this method work you need to assign Physics Raycaster component to your camera and also assign Touch Pickup Handler to each item prefab you have!



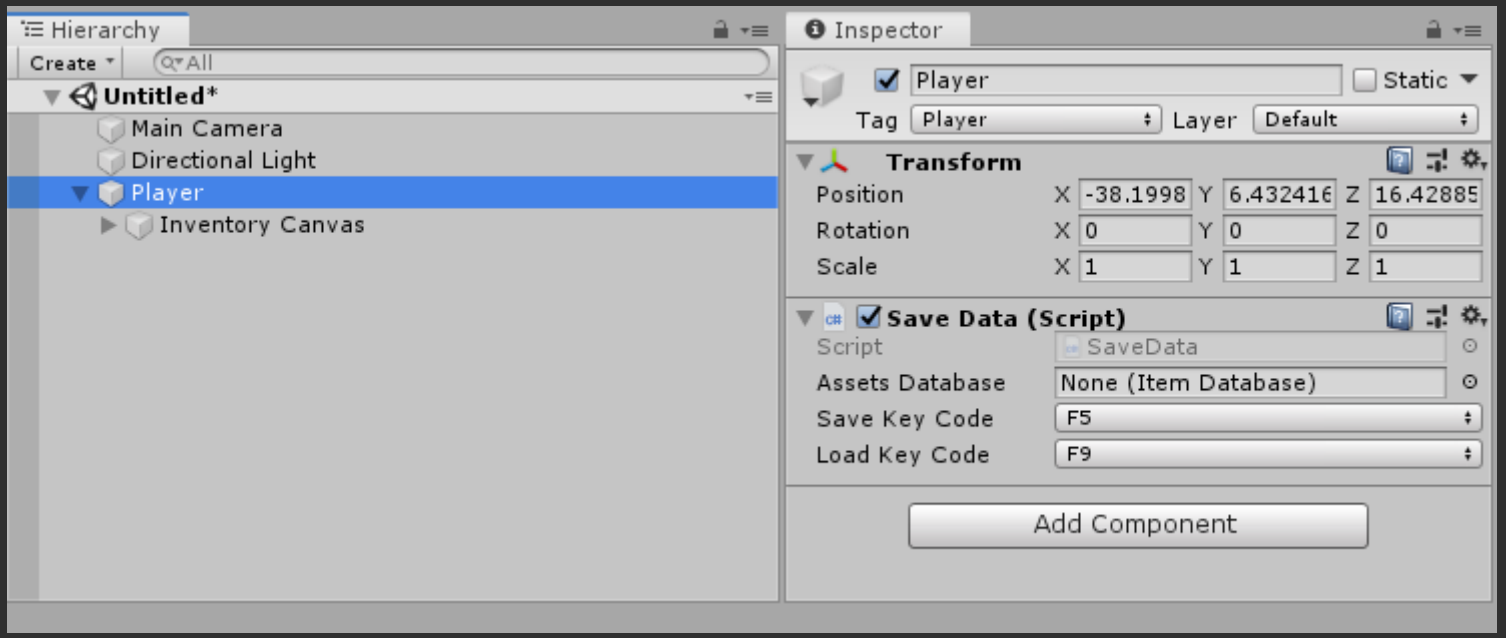


If you've done everything right you can pickup items with clicking 'em with mouse or mobile input.

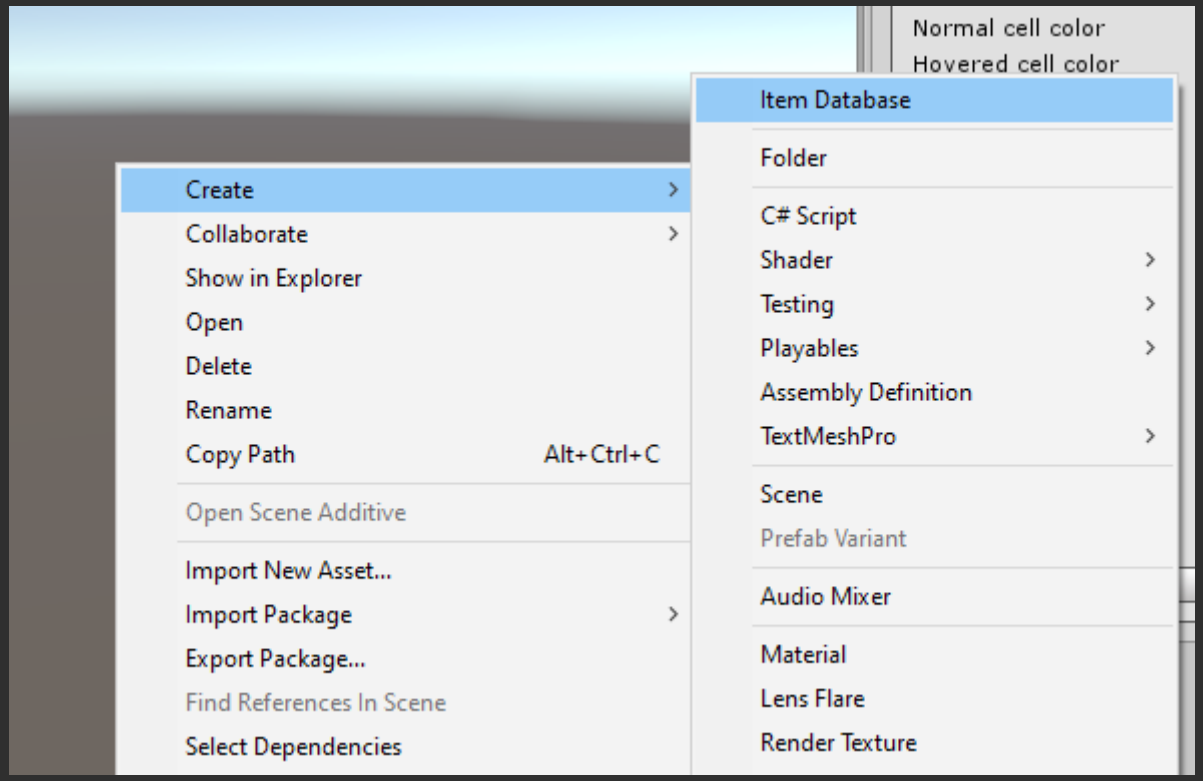
Save system

DT Inventory support items persistence. It means that you can save and load scene state for items/inventory/lootboxes. Also you can save state of the scene you enter or leave. All items will be at the same place in both cases.

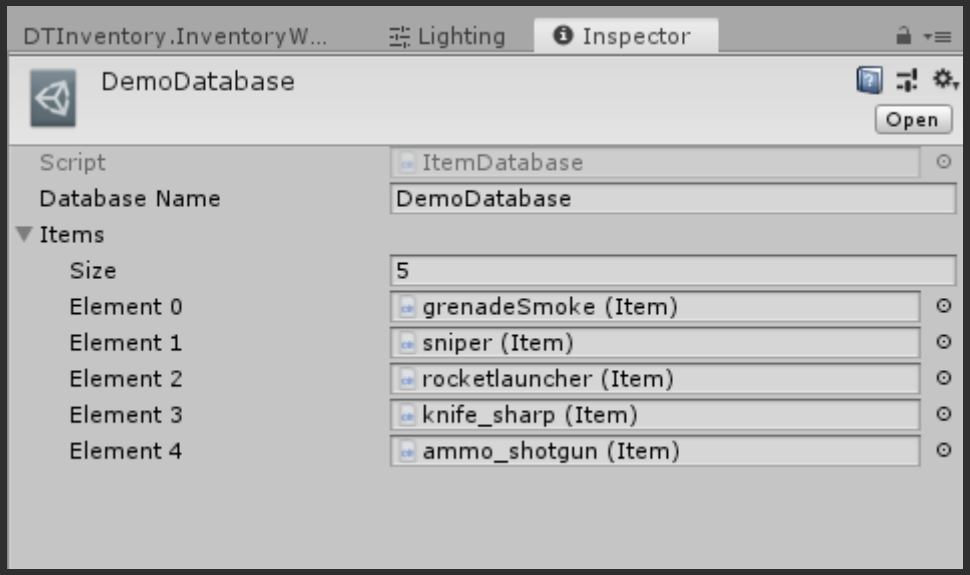
To make save/load work we need to assign SaveData component to your player gameobject. Also drag inventory canvas to your player. After this step you should get something familiar with the picture.



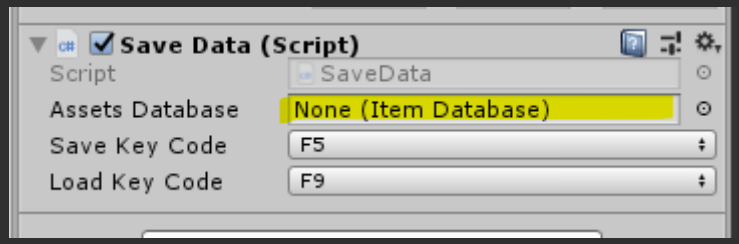
Now we need to create asset database that will store all item objects of your project. To do that click right button in project view and select ‘Item database’.



Now assign each item to Items list. If you missed item, you won't be able to restore it on load methods and SaveData script will throw exception.



And last step is to assign your new database to this field. Set load/save keys and hit play. Save/Load works now!



P.S. For users of DTFPS 1.4. Inventory use own Assets Database script (see on GamePrefab)

Persistence system

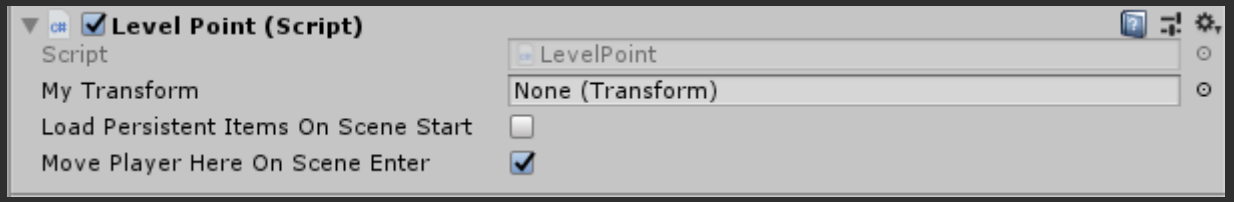
Now we can touch persistence system… For example we have 2 scenes in our project ‘Scene A’ and ‘Scene B’, and we want to keep all items that we have with their last state. We don’t want to lost our scene state when we go from ‘Scene B’ to ‘Scene A’ and want our data to be persistent.

First, our player gameobject must exist regardless of the current scene. Just attach DontDestroyInstance to your player and set player gameobject tag to ‘player’. Attach RegisterPlayerInstance script on player aswell. Now drag your inventroy to player hierarchy. Now your inventory and player are persistence and will not be destroyed during scene transition.

To walk between the levels you have we need to set them in build settings.

Create cube and set its collider as trigger. Attach LevelTransition script and set scene index you want to go.

Also we must have spawn points on the level. Create empty gameobject and attach LevelPoint script. Do it for second scene as well. Set Load Persistent Item On Scene Start if you want to leave items on the same place if you going from one level to another.



After some test iterations you will see that we still keep persistence information from other last sessions. To remove that find inventory script and press clear persistence data. It will remove all scene persistence information you have.