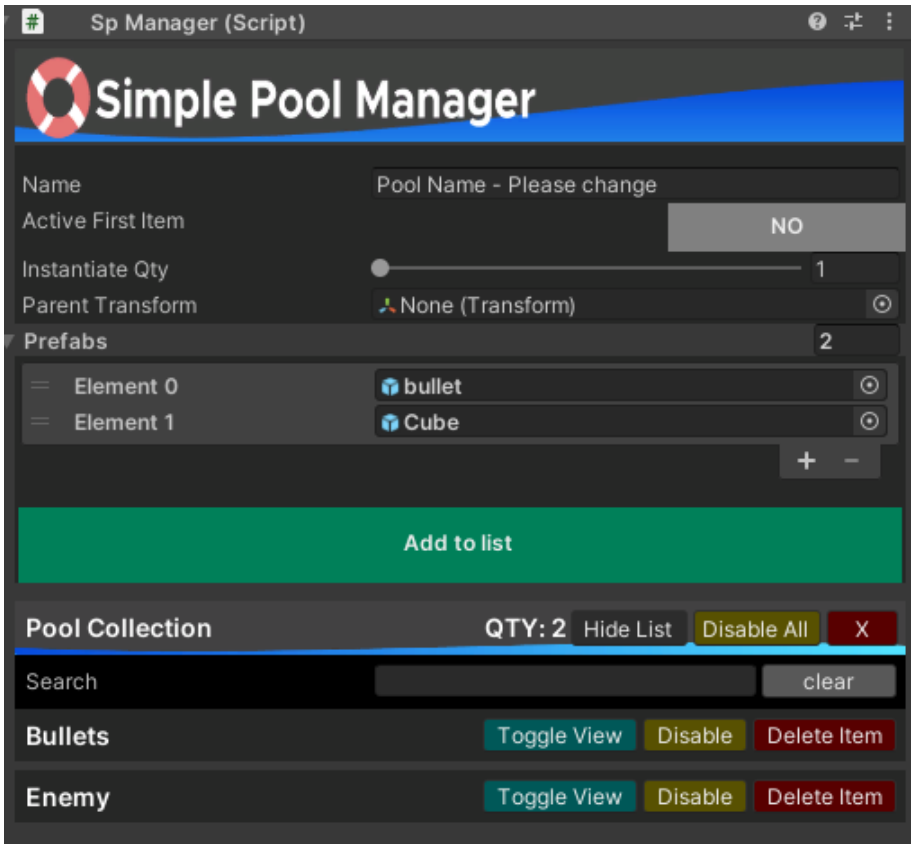
Simple Pool Manager

A simple pooling system that allows reusability and flexibility when requiring multiple instances of the same object (eg: Bullets, FX, etc)

About:

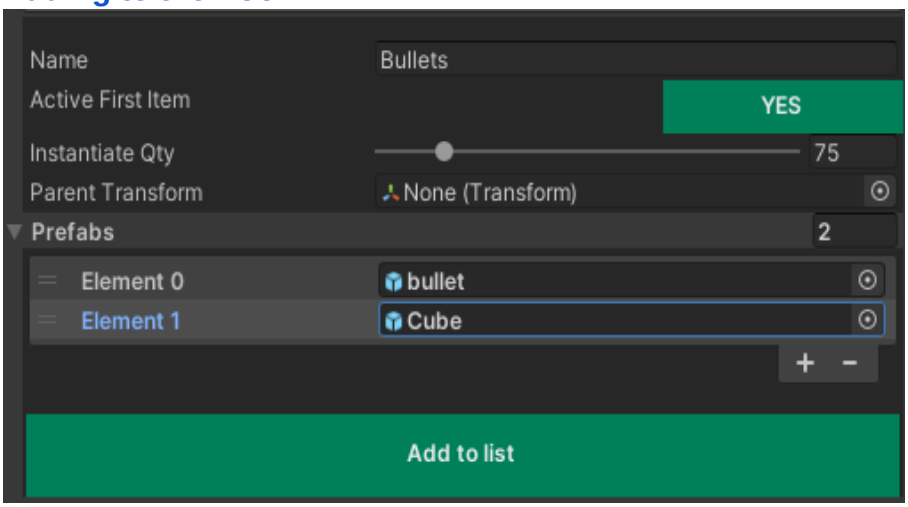
The **SPManager** aka ***SimplePoolManager*** script is to be attached to a GameObject on the Scene. Upon attaching the script, you will be able to add and remove items from the list in which you want to instantiate. Each **POOL** added is its own collection. All pool items are then instantiated (created) as soon as the PLAY is active.

SPManager Editor example:



Break Down:

Adding to the Pool -

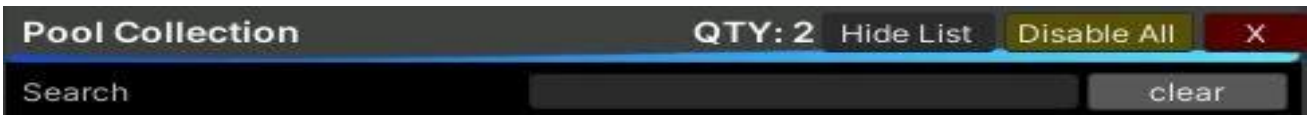


- **Name:** The name of the pool collection to add *(ex: Bullets, Splats, chickens, etc.)*
- **Active First Item:** True/False value that enables the first item on the list by default.

  (sets it active and visible wherever it is instantiated.)
- **Instantiate Qty:** The quantity of items you want to create for this given pool
- **Parent Transform:** The location in which you want the items to be instantiated UNDER.

  (Viewed as a “holder” of the pool items. Ex: a game object named Bulled Holder)
- **Prefab:** The items that will be instantiated. **UPDATED- You can now add more than one prefab object to be instantiated, for Variety!**

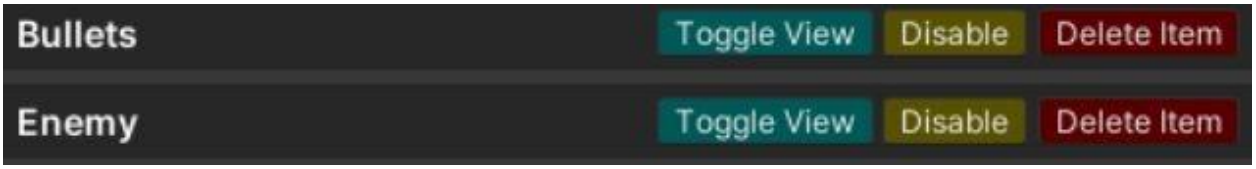
Viewing / Editing / Searching the pool -



- **QTY (~):** Displays the Pool quantity
- **Hide List / Show List:** This button toggles to display the pool list and hide it
- **Disable All:** This button disables ALL of the pool items that are active on the scene

- **[X]:** Deletes ALL of the pool items in the collection list
- **Search:** Allows you to search the pool list by typing the name or the first 3 characters of the name. It returns ALL items that have the provided value entered
- **Clear:** Clears the search result

Pool Items -



- **Name:** The first value on the list is the name of the Pool that was provided when created (This example shows Bullets and Enemy)
- **Toggle View:** This button toggles the details for the specific pool it is on
- **Disable:** This button disables all the pool items for the pool it is under ONLY
- **Delete Item:** This button deletes the specific pool item in the Pool collection

Good to know:

UPDATED

- The **SPManager** inherits from **IPoolActions** which is used to avoid requesting the entire object's information. Better performance, only show what you need.

- The following image will show you how easy it is to add SPManger actions to any other script.

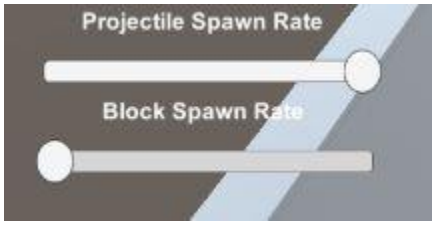
```
private IPoolActions _spManager;
Event function  new *
private void Awake()
{
    _spManager = FindObjectOfType<SpManager>();
}
```

NOTE how we look for our object within the AWAKE method, since it is an request that should only happen once and will never change, along with better performance.

Scene Example:

In the Scene example provided you can see how the SPManager works and how it has been set up with basic objects. On the Hierarchy you will find the SPManager game object that displays the tool. Everything else are just objects to support the example scene.

In this scene you can view two (2) sliders that will allow you to adjust the spawn rate for both the projectile and the “enemy” (boxes) that are created.



Both the Character and Block Spawner GameObject have scripts that show how to interact with the SPManager exposed functions. Such as the **GetNextAvailablePoolItem**. The Bullet prefab has a Despawn script that also uses an SPManager function called **DisablePoolObject** ***(Please see list of functions below for details or visit the SPManager script. The code is documented accordingly)***

SPManager accessible Functions:

Actions	Parameters	Notes
GetSpecificPoolItemInCollection	poolName, index	This method is overloaded with additional parameters to pass through
GetNextAvailablePoolItem	poolName	Grabs the next available *inactive* pool item
GetRandomPoolItem	poolName	Grabs a random *inactive* pool item. Great for random spawning or other
GetAllItemInACategory	poolName	Returns a list of all the items in a specific pool
GetAllActiveCategoryItem	poolName	Returns a list of all the items that are active on the scene (In case you need to ...blow them up?)

GetPoolItemParentTransform	poolName	Returns the parent of the pool item list (Wherever the list was instantiated into as a child)
DoesPoolExist	poolName	Returns a *Boolean* to inform if the specific pool exists (Great to check before creating a new pool on the fly)
DoesPoolItemExist	GameObject	Checks to see if the given Game object is present on any pool (To be expanded)
GetAllItemInACategory	poolName	Returns a list of all the items in a specific pool

AddNewItemToCollection	List	Let's you add a *NEW* pool collection to the pool list
DisablePoolObject	poolName, GameObject/Transform	Disables the given pool object that belongs to the provided Pool. (Overloaded Method)
DisableAllPoolObjects	null	Disables ALL pool objects that are active
DisableAllPoolObjectByCategory	poolName	Disables all pool objects that belong to a specified pool