Overview UI Sprite Manager

Introduction

Thanks for downloading this package stay tuned this is just the beginning or you may call it proof of concept more features are in the queue.

After spending some years in Game Industry, I realize managing sprites in large projects is a headache, as artists have to iterate images a lot to make them better, cool and according to demand. While developers have to integrate those images and to show the feel of those images in the game.

Also, due to this hassle, artists and developers choose a messy workaround, and instead of working in spritesheet, they start working in individual sprites.

Purpose

In big Unity projects, artists have to modify images frequently to make it better, and developers have to reflect that change, as the old reference becomes Missing (if the name of the updated sprite is not the same)

Unity Asset Store Link

https://www.assetstore.unity3d.com/#!/content/89387

GitHub Link

https://github.com/mohsinkhan26/UISpriteManager

BitBucket Link

https://bitbucket.org/unbounded-eagle/ui_sprite_manager/

Blog Link

http://cslearners.blogspot.com/2017/04/unity3d-ui-sprite-manager.html

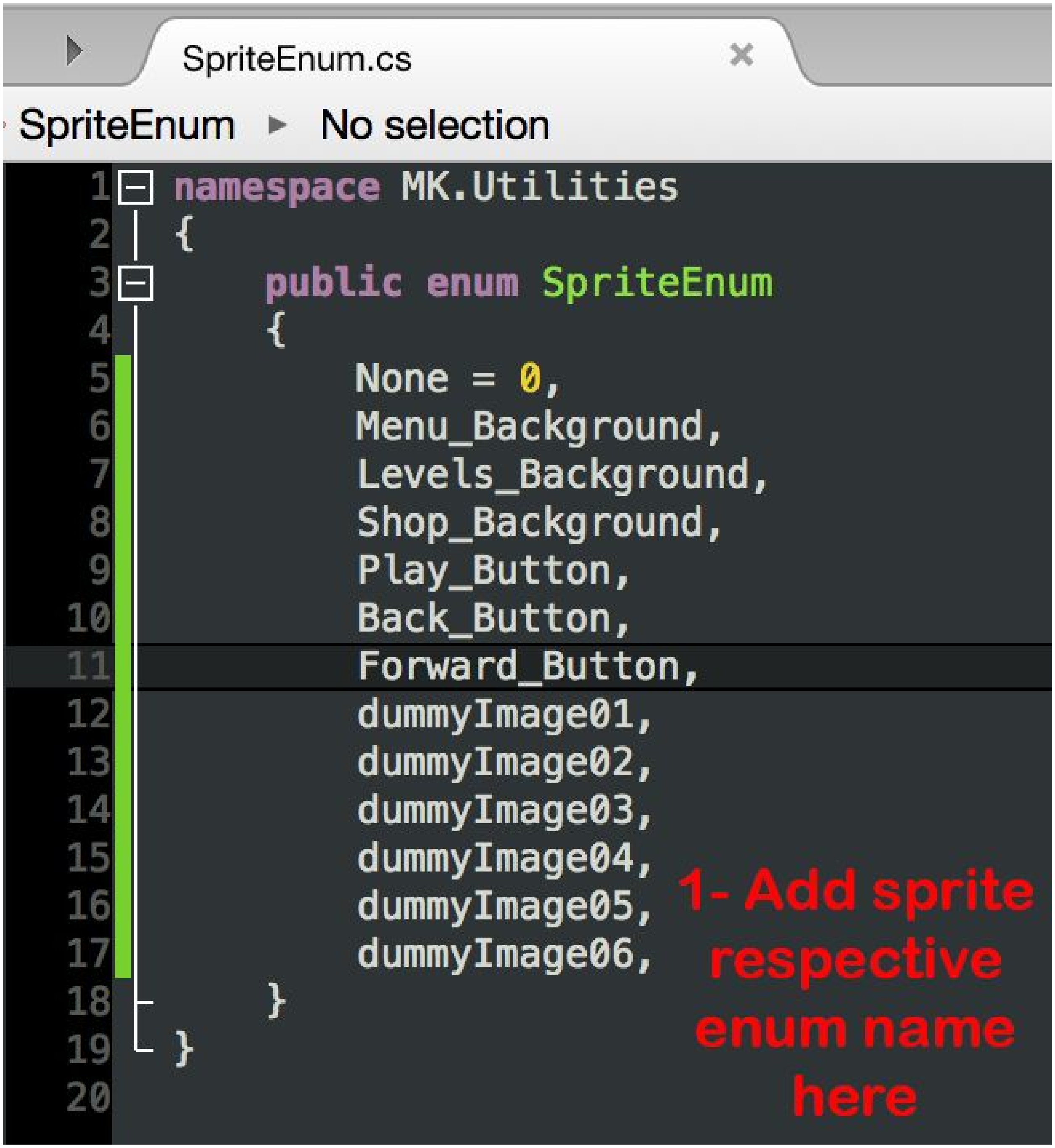
Features

- Easily make reskin of your game, if the game is made using UI Sprite Manager
- Promotes the use of spritesheet/atlas
- Get full use of spritesheet and packaging tag
- Place images anywhere in Assets/, as if the images are placed in Assets/Resources packaging tag don't work

- Still you can change the sprite on runtime with ImageExtension public methods
- Easy to use, Plug n play
- Open Source code without any DLL
- Done by extending Image component

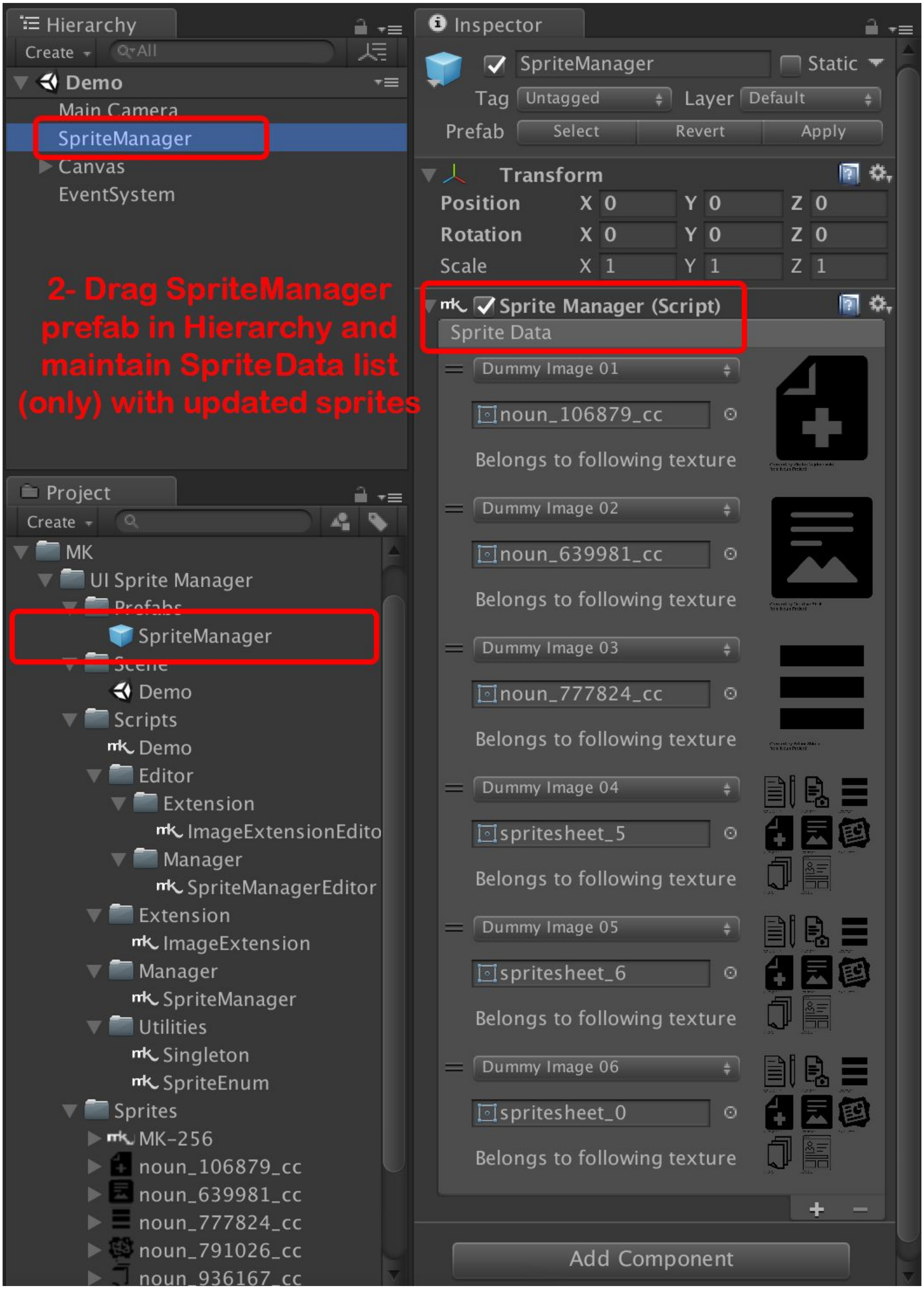
Setup

- Import plugin
- Add enum against your sprite in SpriteEnum.cs
- Add respective sprite in SpriteManager.prefab
- Now, just add component ImageExtension.cs instead of Image in Unity UI and use it as is



```
namespace MK.Utilities
{
    public enum SpriteEnum
    {
        None = 0,
        Menu_Background,
        Levels_Background,
        Shop_Background,
        Play_Button,
        Back_Button,
        Forward_Button,
        dummyImage01,
        dummyImage02,
        dummyImage03,
        dummyImage04,
        dummyImage05,
        dummyImage06,
    }
}
```

1- Add sprite respective enum name here



2- Drag SpriteManager prefab in Hierarchy and maintain SpriteData list (only) with updated sprites

☑ SpriteManager □ Static

Tag Untagged Layer Default

Prefab

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

☑ Sprite Manager (Script)

Sprite Data

Dummy Image 01
noun_106879_cc
Belongs to following texture

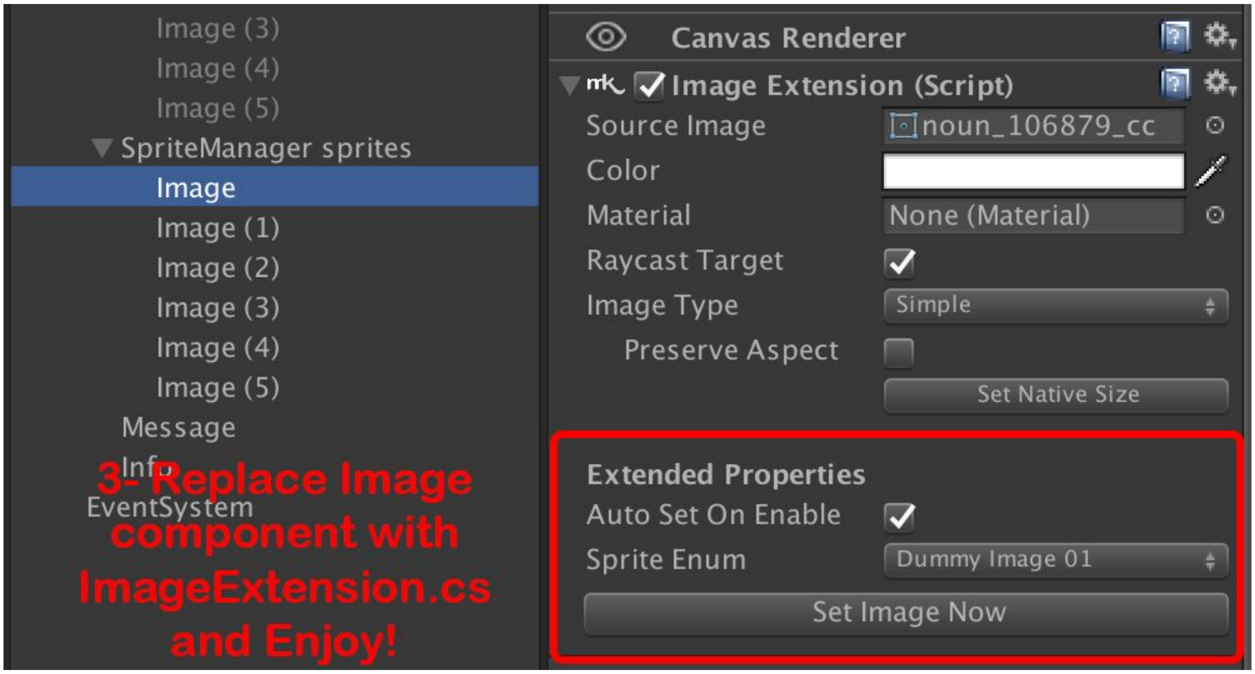
Dummy Image 02
noun_639981_cc
Belongs to following texture

Dummy Image 03
noun_777824_cc
Belongs to following texture

Dummy Image 04
spritesheet_5
Belongs to following texture

Dummy Image 05
spritesheet_6
Belongs to following texture

Dummy Image 06
spritesheet_0
Belongs to following texture



☑ Image Extension (Script)

Source Image: noun_106879_cc

Color

Material: None (Material)

Raycast Target ☑

Image Type: Simple

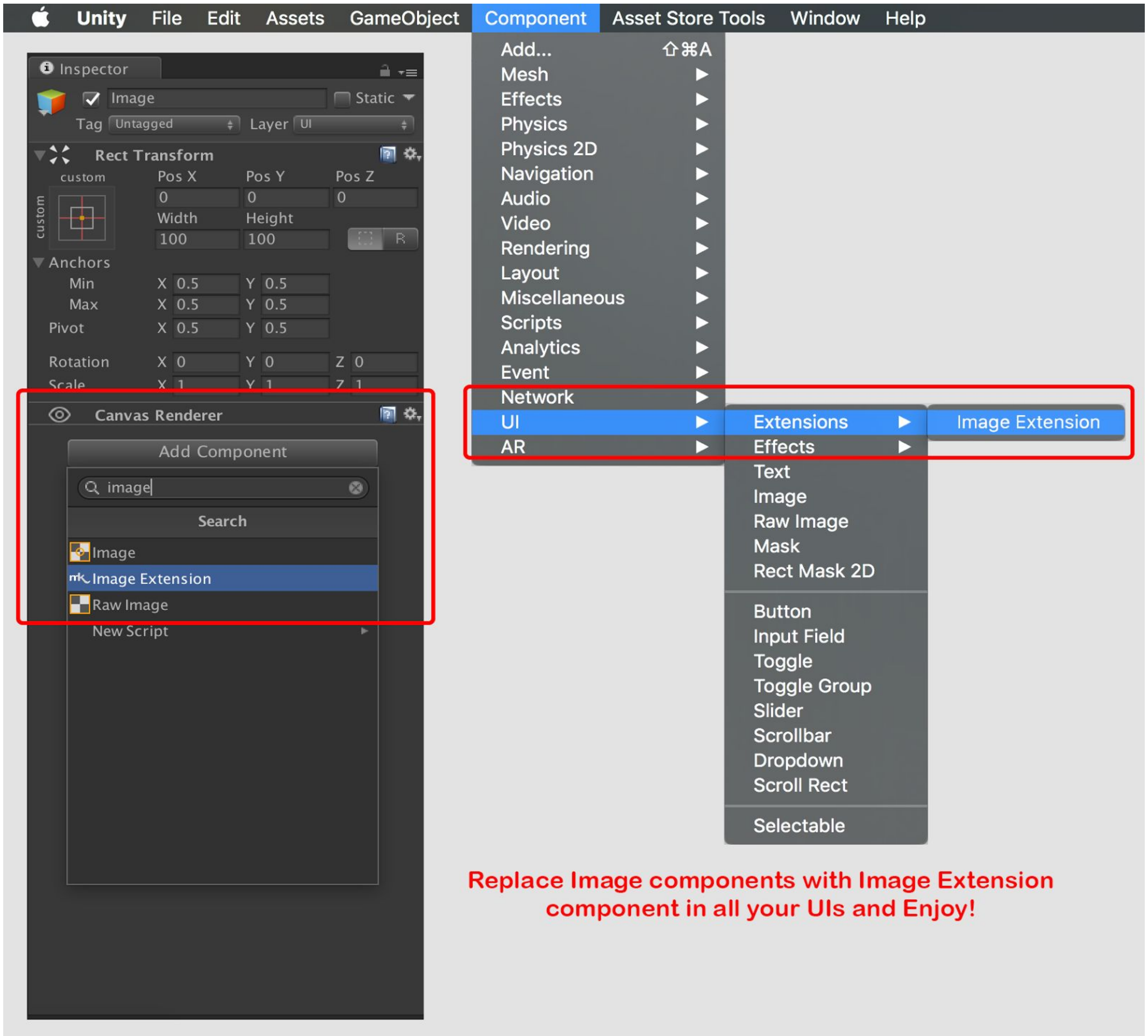
Preserve Aspect □

Extended Properties

Auto Set On Enable ☑

Sprite Enum: Dummy Image 01

3- Replace Image component with ImageExtension.cs and Enjoy!



☑ Image

Tag Untagged Layer UI

Pos X 0 Pos Y 0 Pos Z 0

Width 100 Height 100

Anchors

Min X 0.5 Y 0.5

Max X 0.5 Y 0.5

Pivot X 0.5 Y 0.5

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

image

Replace Image components with Image Extension component in all your UIs and Enjoy!

Remember

- Drag SpriteManager.prefab in the first scene and it will persist

Special Thanks

- All the users who provide feedback and suggestions to improve
- All the users who gave reviews on Asset store

Thanks for your support!