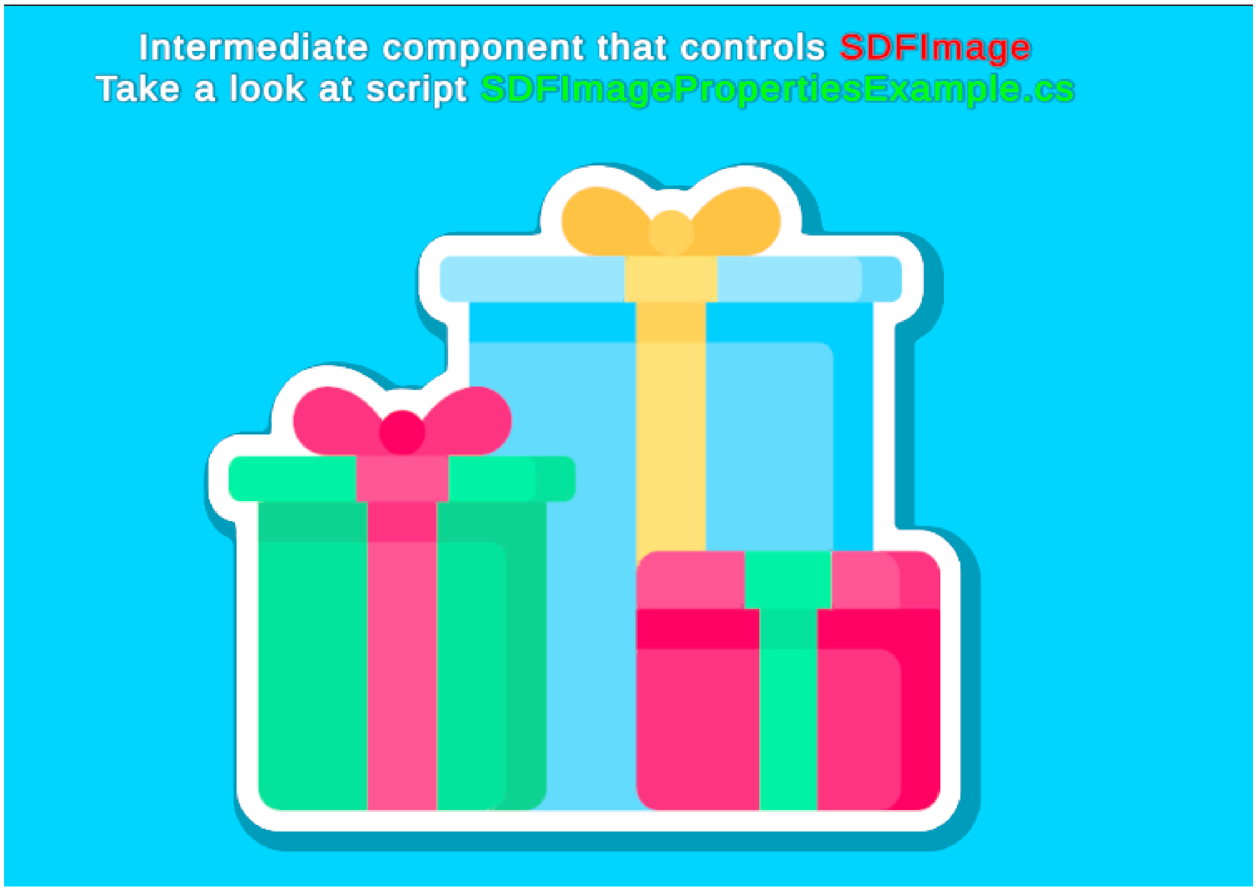
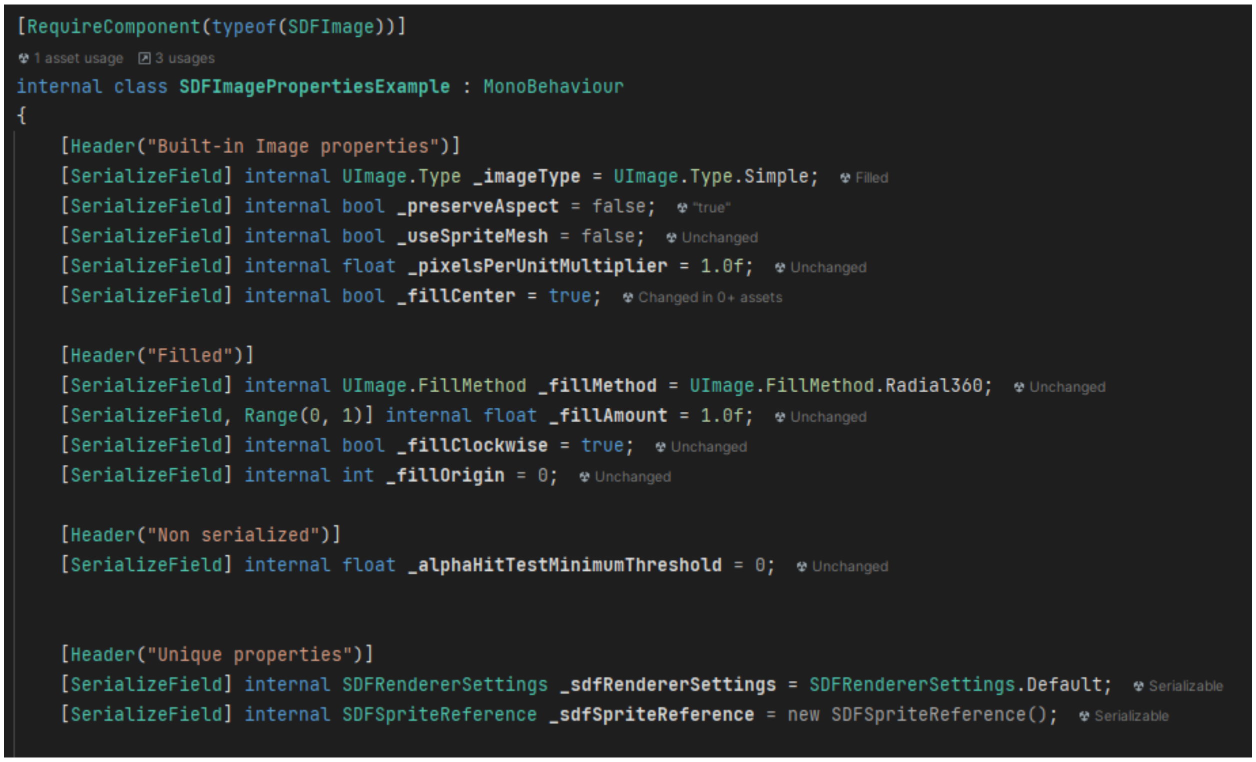
SDFImage Properties

Status	Up-to date

Sample files overview


Intermediate component that controls SDFImage
Take a look at script SDFImagePropertiesExample.cs

Scene view

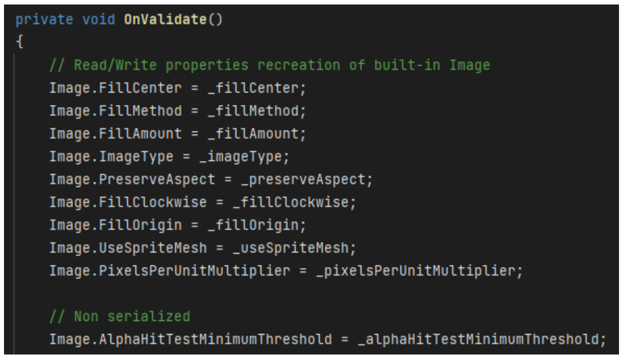


```
[RequireComponent(typeof(SDFImage))]
1 asset usage   3 usages
internal class SDFImagePropertiesExample : MonoBehaviour
{
    [Header("Built-in Image properties")]
    [SerializeField] internal UImage.Type _imageType = UImage.Type.Simple;   Filled
    [SerializeField] internal bool _preserveAspect = false;   "true"
    [SerializeField] internal bool _useSpriteMesh = false;   Unchanged
    [SerializeField] internal float _pixelsPerUnitMultiplier = 1.0f;   Unchanged
    [SerializeField] internal bool _fillCenter = true;   Changed in 0+ assets

    [Header("Filled")]
    [SerializeField] internal UImage.FillMethod _fillMethod = UImage.FillMethod.Radial360;   Unchanged
    [SerializeField, Range(0, 1)] internal float _fillAmount = 1.0f;   Unchanged
    [SerializeField] internal bool _fillClockwise = true;   Unchanged
    [SerializeField] internal int _fillOrigin = 0;   Unchanged

    [Header("Non serialized")]
    [SerializeField] internal float _alphaHitTestMinimumThreshold = 0;   Unchanged


    [Header("Unique properties")]
    [SerializeField] internal SDFRendererSettings _sdfRendererSettings = SDFRendererSettings.Default;   Serializable
    [SerializeField] internal SDFSpriteReference _sdfSpriteReference = new SDFSpriteReference();   Serializable
```

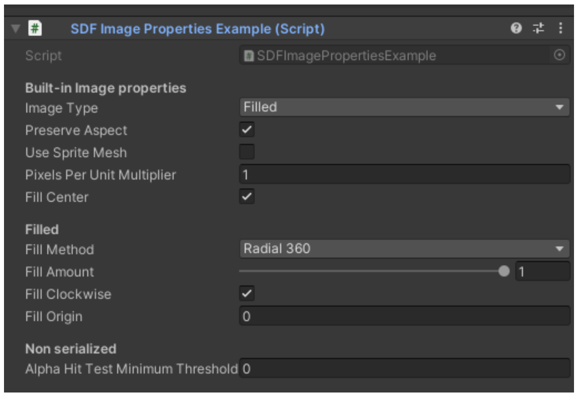
This sample provide look at how to interact with SDFImage through script.

Properties

Script contains same fields as SDF Image has



```
private void OnValidate()
{
    // Read/Write properties recreation of built-in Image
    Image.FillCenter = _fillCenter;
    Image.FillMethod = _fillMethod;
    Image.FillAmount = _fillAmount;
    Image.ImageType = _imageType;
    Image.PreserveAspect = _preserveAspect;
    Image.FillClockwise = _fillClockwise;
    Image.FillOrigin = _fillOrigin;
    Image.UseSpriteMesh = _useSpriteMesh;
    Image.PixelsPerUnitMultiplier = _pixelsPerUnitMultiplier;

    // Non serialized
    Image.AlphaHitTestMinimumThreshold = _alphaHitTestMinimumThreshold;
```



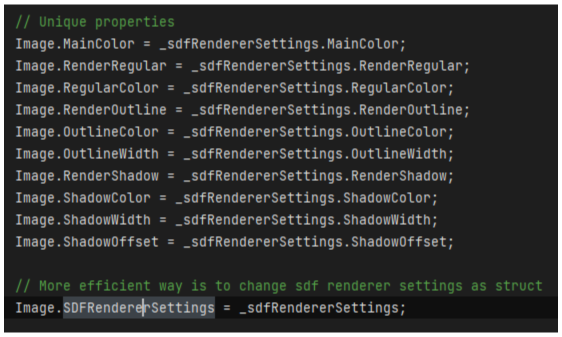
SDF Image Properties Example (Script)

Script	SDFImagePropertiesExample
Built-in Image properties	
Image Type	Filled
Preserve Aspect	✓
Use Sprite Mesh	
Pixels Per Unit Multiplier	1
Fill Center	✓
Filled	
Fill Method	Radial 360
Fill Amount	1
Fill Clockwise	✓
Fill Origin	0
Non serialized	
Alpha Hit Test Minimum Threshold	0

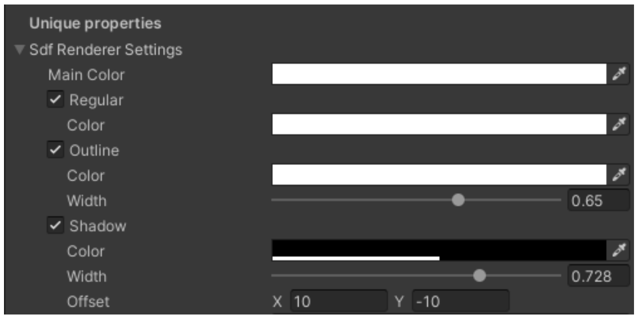
SDFImage has recreated all properties that built-in Image has, since it no longer inherits from it. But all public properties named in PascalCase according to .NET standards.

Unique to SDFImage

`SDFRendererSettings`



```
// Unique properties
Image.MainColor = _sdfRendererSettings.MainColor;
Image.RenderRegular = _sdfRendererSettings.RenderRegular;
Image.RegularColor = _sdfRendererSettings.RegularColor;
Image.RenderOutline = _sdfRendererSettings.RenderOutline;
Image.OutlineColor = _sdfRendererSettings.OutlineColor;
Image.OutlineWidth = _sdfRendererSettings.OutlineWidth;
Image.RenderShadow = _sdfRendererSettings.RenderShadow;
Image.ShadowColor = _sdfRendererSettings.ShadowColor;
Image.ShadowWidth = _sdfRendererSettings.ShadowWidth;
Image.ShadowOffset = _sdfRendererSettings.ShadowOffset;

// More efficient way is to change sdf renderer settings as struct
Image.SDFRendererSettings = _sdfRendererSettings;
```



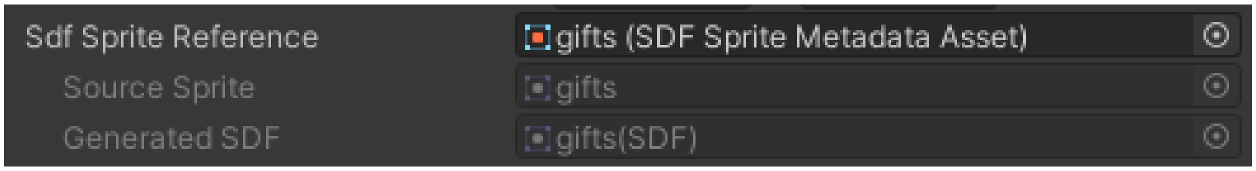
Unique properties

Sdf Renderer Settings

Main Color	
✓ Regular	
Color	
✓ Outline	
Color	
Width	0.65
✓ Shadow	
Color	
Width	0.728
Offset	X 10 Y -10

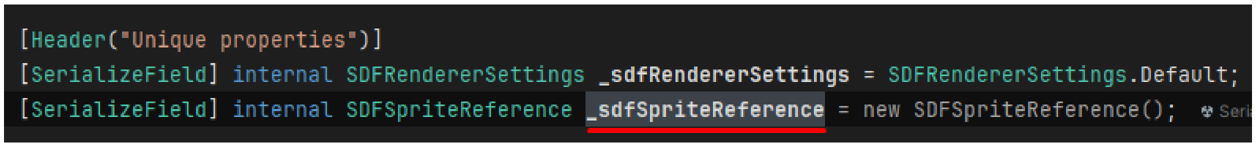
This is all properties related to mesh generation and vertex data. In code you can set them directly with `SDFImage.SDFRendererSettings` rather than each individually.

How to set sprite (`SDFSpriteReference`)



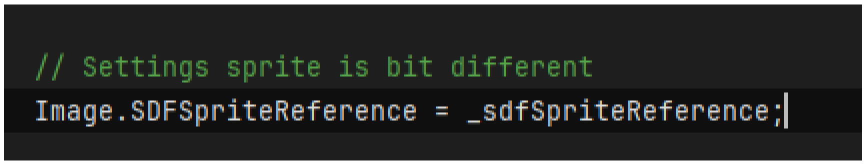
Sdf Sprite Reference	gifts (SDF Sprite Metadata Asset)
Source Sprite	gifts
Generated SDF	gifts(SDF)

Editor view



```
[Header("Unique properties")]
[SerializeField] internal SDFRendererSettings _sdfRendererSettings = SDFRendererSettings.Default;
[SerializeField] internal SDFSpriteReference _sdfSpriteReference = new SDFSpriteReference();
```

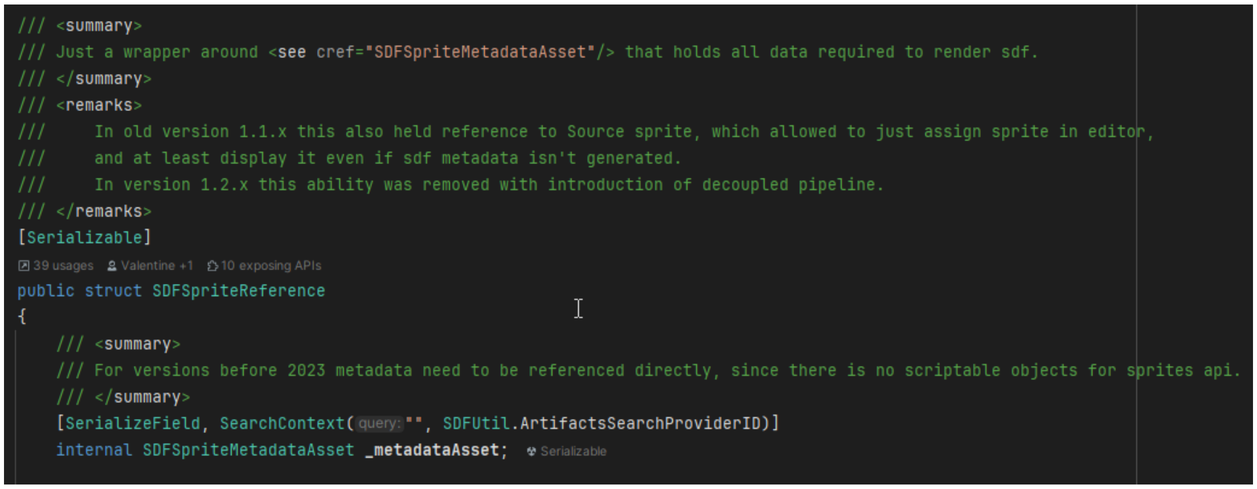
Fields definition



```
// Settings sprite is bit different
Image.SDFSpriteReference = _sdfSpriteReference;
```

Property setting

This how you change sprites at runtime, by settings struct `SDFSpriteReference` to corresponding property.



```
/// <summary>
/// Just a wrapper around <see cref="SDFSpriteMetadataAsset"/> that holds all data required to render sdf.
/// </summary>
/// <remarks>
///     In old version 1.1.x this also held reference to Source sprite, which allowed to just assign sprite in editor,
///     and at least display it even if sdf metadata isn't generated.
///     In version 1.2.x this ability was removed with introduction of decoupled pipeline.
/// </remarks>
[Serializable]
39 usages   Valentine +1   10 exposing APIs
public struct SDFSpriteReference
{
    /// <summary>
    /// For versions before 2023 metadata need to be referenced directly, since there is no scriptable objects for sprites api.
    /// </summary>
    [SerializeField, SearchContext(query: "", SDFUtil.ArtifactsSearchProviderID)]
    internal SDFSpriteMetadataAsset _metadataAsset;   Serializable
```

Docs with detailed explanation on why this struct exist

`SDFSpriteReference` is exist for multi-version compatibility, in versions before 1.2.x it was holding more properties, and handled some custom serialization, but right now its just an wrapper around

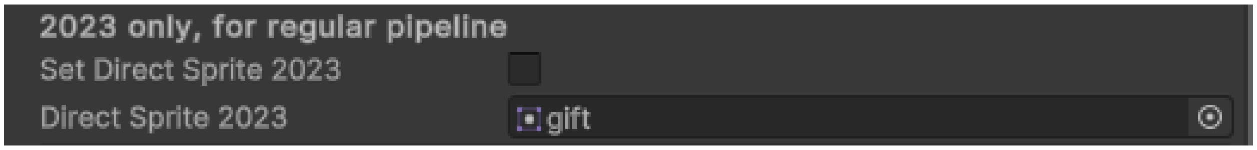
`SDFSpriteMetadataAsset`.

It will be kept in future versions as its useful to have some intermediate serialization layer, instead of plain object reference.

Other way to set sprite (2023 version)

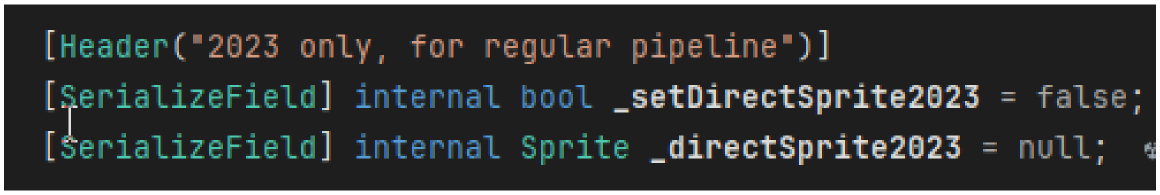
There is another way to set sprite in unity 2023 with the introduction of Scriptable Objects for Sprites API. This way you can extract `SDFSpriteMetadataAsset` directly from sprite.

> ⚠ **Note that this won't work for source sprites in decoupled pipeline, see more** Decoupled Pipeline Scripting



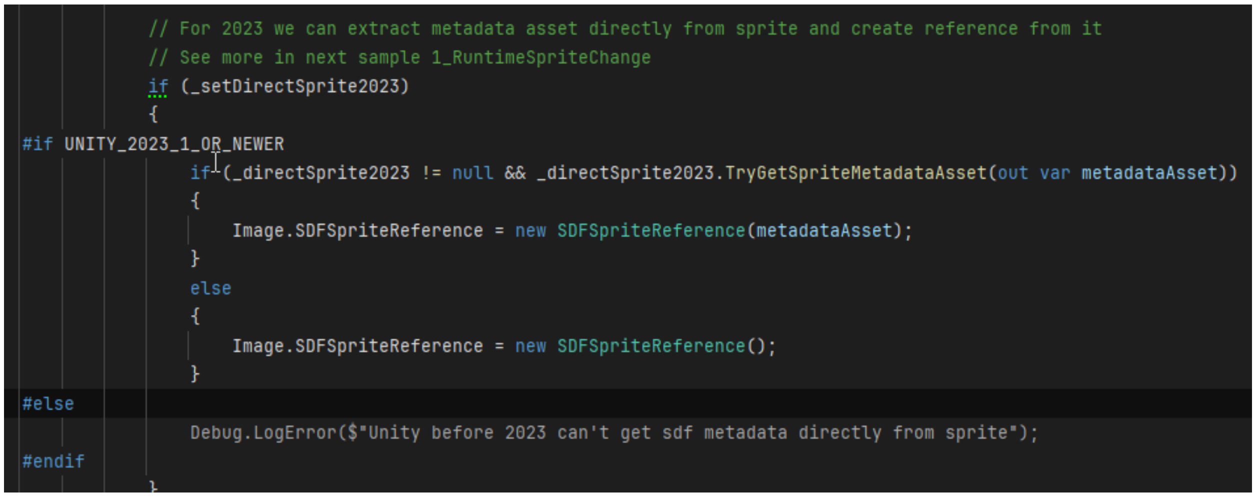
2023 only, for regular pipeline
Set Direct Sprite 2023
Direct Sprite 2023 gift

Editor view



```
[Header("2023 only, for regular pipeline")]
[SerializeField] internal bool _setDirectSprite2023 = false;
[SerializeField] internal Sprite _directSprite2023 = null;
```

Fields definition



```
            // For 2023 we can extract metadata asset directly from sprite and create reference from it
            // See more in next sample 1_RuntimeSpriteChange
            if (_setDirectSprite2023)
            {
#if UNITY_2023_1_OR_NEWER
                if (_directSprite2023 != null && _directSprite2023.TryGetSpriteMetadataAsset(out var metadataAsset))
                {
                    Image.SDFSpriteReference = new SDFSpriteReference(metadataAsset);
                }
                else
                {
                    Image.SDFSpriteReference = new SDFSpriteReference();
                }
#else
                Debug.LogError($"Unity before 2023 can't get sdf metadata directly from sprite");
#endif
            }
```

Property setting

💡 **Try flipping Set Direct Sprite 2023 toggle and see how sprite changes**

Runtime sprite change

There is multiple ways to change sprite at runtime, here we quickly covered some basic ones.

But there is other ways in depth explained in Runtime Sprite Change, Decoupled Pipeline Scripting and Addressables Support