SWIFT SHADOWS

by

User Manual

v. 1.1.1

General information

Swift Shadows is a lightweight substitute for Blob Shadow Projector, heavily optimized for mobile devices. It works by doing a raycast to the surface and drawing a transparent quad that hovers above it. Literally hundreds of shadows can be drawn on a mobile device with a single draw call.

Features:

- Works on any platform, does not requires Pro license to work. Heavily optimized for mobile.
- Supports shadow texture atlases. All shadows in the scene can be drawn in a single draw call, even when they have different shapes.
- Tons of customization options.
- Shadows match object rotations.
- "Static" shadows that are calculated only once and produce no overhead in run-time. It is possible to have hundreds of static shadows without a performance hit even on mobile devices.
- Ease of use, absolutely no scripting required.
- Shadow generator tool for creating nice looking shadows automatically.
- Sprite packer tool for easily packing multiple shadow textures into one to reduce the amount of drawcalls.
- *Blazing fast.* And I *really* mean it.

Limitations:

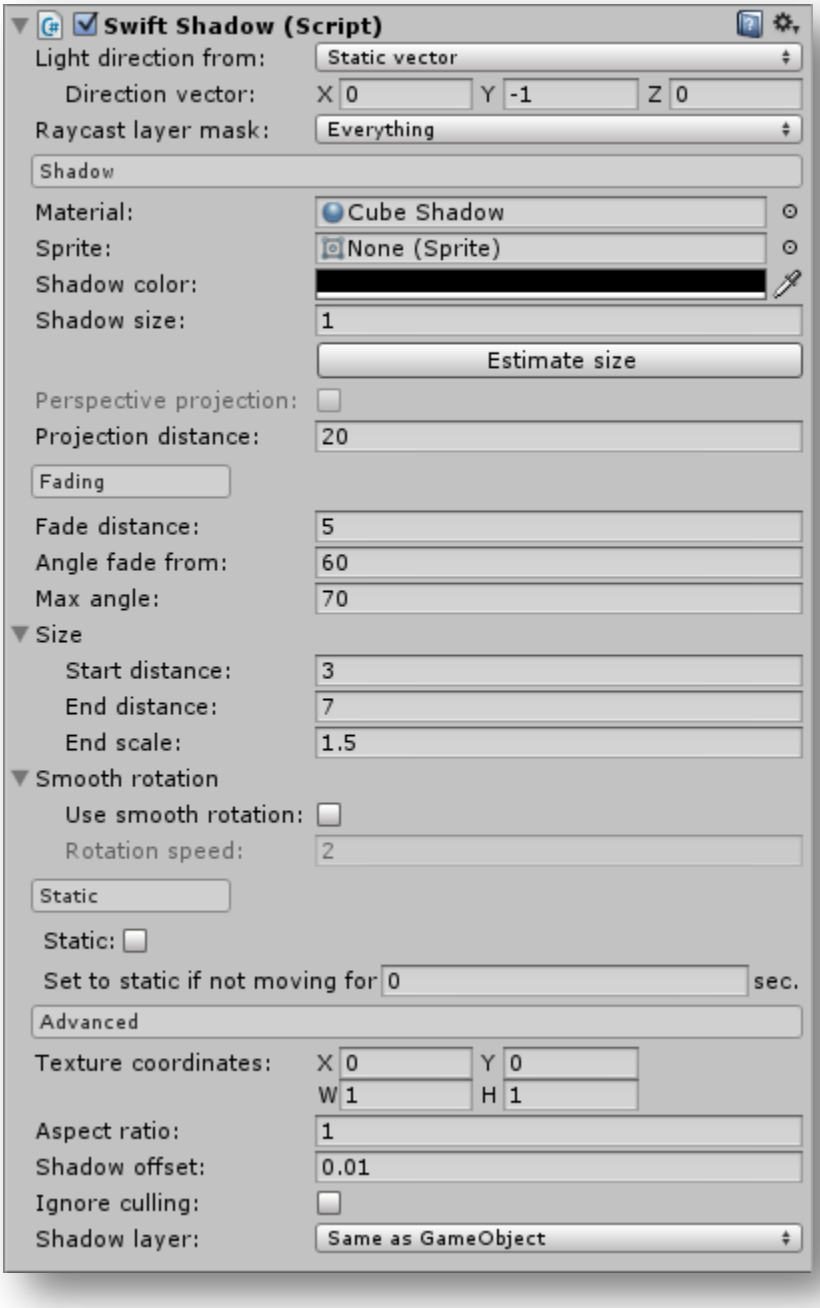
- Shadow textures are static. Shape of a single shadow is *not* updated in real-time. In some cases it is possible to work around this by using animated shadow cookie or by using multiple shadows.
- Each individual shadows only projects on one surface at a time. This leads to a discontinuity as shadow moves from one object to another.
- Shadow can extend the edges of object it is being projected on. It can also clip inside complex-shaped objects.

These peculiarities are rarely noticeable for moving objects with blurry edges.

Plugin is tested in Unity 4.x – 5.x. Pro license is not required.

Integration

Just add *SwiftShadow* component to your object, hit Play and you'll see a fuzzy dark blob shadow right below the object. No scripting required!



Let's take a closer look on available options.

LIGHT DIRECTION

Specifies the light direction vector. It can be entered manually, or calculated in run-time from a selected light source. You can pick any `GameObject` and it will act a point light source. Directional light sources also work as expected.

RAYCAST LAYER MASK

Specifies layers on which the shadow will be projected. Usually, you must exclude the layer of the object itself from the layer mask, otherwise the shadow may collide inside of object and behave weird.

MATERIAL

Defines the material to be used for rendering the shadow. Shadows with the same material are automatically batched.

SPRITE

Defines the sprite to be used for rendering the shadow. Sprites must not be packed, must have Rect mesh type, and the main texture of MATERIAL must be the same as the sprite texture.

SHADOW COLOR

Defines the shadow color. Changing this does not break batching! The color data is written in the vertex color attribute.

SHADOW SIZE

Defines the size of the shadow.

PERSPECTIVE PROJECTION

Shadow will grow and distort relative to the light source. This usually makes shadows look more realistic, but may result in artifacts at extreme angles.

Be aware that perspective shadows have lower performance that orthographic and are also harder to cull. Because of that, you may want not to enable this option when you have hundreds of shadows on mobile device.

This option can only be enabled if light source object is used.

PROJECTION DISTANCE

Maximal distance from the object to the surface. Shadow will not appear beyond this distance.

FADE DISTANCE

Defines distance to the surface at which the shadow will start fading out. Shadow will totally fade out when reaching the PROJECTION DISTANCE.

ANGLE FADE FROM

Defines the angle at which shadow will start fading out. Shadow will totally fade out when reaching MAX ANGLE.

MAX ANGLE

Target won't be drawn if shadow falls on the surface at an angle that is larger than this value.

SIZE – START DISTANCE & END DISTANCE

These two options determine the interval in which the shadow will change size accordingly to the distance to the ground. When the distance is smaller than START DISTANCE, the size will remain the same as set with SHADOW SIZE option. When approaching END DISTANCE, the SHADOW SIZE will be smoothly multiplied with END SCALE value.

This is useful for imitating shadows getting more blurry as the object moves away from the ground.

SIZE – END SCALE

This value is a multiplier applied to the SHADOW SIZE when the distance to the ground is equal or larger that END DISTANCE.

USE SMOOTH ROTATION

When this option is enabled, shadow will smoothly rotate to the calculated orientation instead of changing it instantly, which may be distracting in some situations.

ROTATION SPEED

When Use smooth rotation is enabled, this value controls how fast the shadow will orient itself to the calculated orientation. The bigger the value – the faster the shadow rotates.

STATIC

If you have shadows that are not moving throughout the gameplay, you can set them as static to get a massive performance boost. Static shadows are only calculated on the `Start()` and not calculated again automatically. Hundreds of static shadows can be drawn even on a mobile device with very little resource consumption.

Be aware, though, that static shadows are always drawn with a separate draw call.

If you want to force an update for static shadows, just use this line of code:

```
ShadowManager.Instance.UpdateStaticShadows();
```

SET STATIC IF NOT MOVING FOR X SEC.

Sometimes objects only move occasionally and stand totally still for rest of the time. This option conveniently allows you to make the object's shadow static automatically if the object is not moving for given time. The object will be automatically returned to non-static state if moved or rotated. This saves a lot of performance without the need of thinking whether the shadow should be manually set to static or not.

Note that setting STATIC manually is still the best option if you know the shadow won't ever change.

TEXTURE COORDINATES

Defines the texture UV coordinates for the shadow. This is used if you have a shadow atlas to render multiple different shadows in a single draw call.

ASPECT RATIO

Defines the *width/height* aspect ratio of shadow frustum. Use this to get rectangular-shaped shadows.

SHADOW OFFSET

This value defines the distance at which the shadow hovers above the surface. It is usually recommended to keep this value as close to 0 as possible, but you may need to increase if you see shadow flickering with the ground (so-called Z-fighting).

IGNORE CULLING

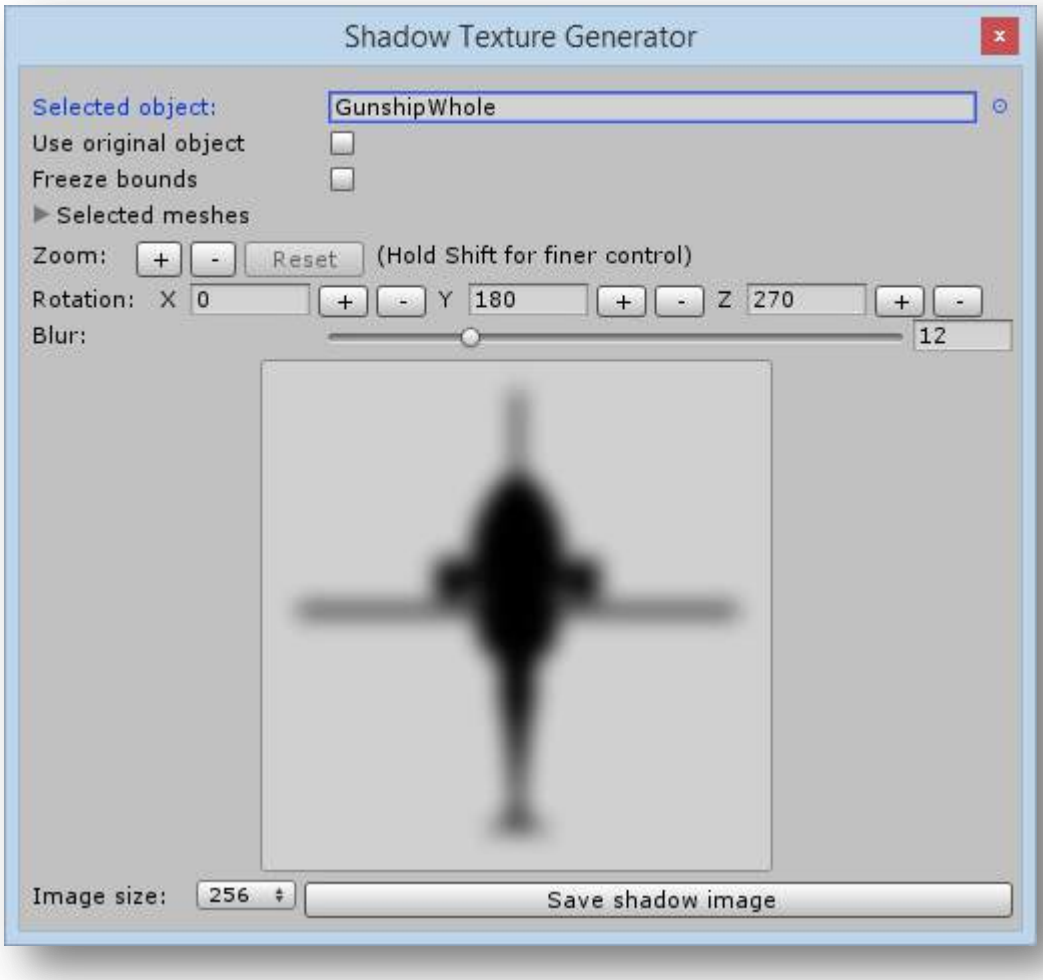
By default, the shadow is not calculated if it cannot be seen, to avoid wasting CPU resources. However, this check has some overhead by itself. Turn on this option if you know your shadow is always in camera's sight to get a small performance boost.

SHADOW LAYER

By default, shadow is drawn on the same layer as the `GameObject`. You can override this behavior and set the layer manually. This may help in certain situations.

Shadow texture generator

For objects with complex shape, a fuzzy circle shadow is not always the best choice. For that case, *Swift Shadows* includes a shadow texture generator that can save you a lot of time with trying to make a nice looking shadow texture manually.



To open the generator window, use this item in the main Unity menu:

Tools → Swift Shadows → Shadow Texture Generator

It is very simple to use: pick an object you want to generate the shadow for, select the object meshes you need to have in the shadow, rotate the object to "view" it from the above, adjust the blur amount to your taste, and save the resulting image!

You can then use it with your custom shadow material to make shadows that exactly match the shape of your objects.

Let's take a closer look on the generator interface.

SELECTED OBJECT

`GameObject` to generate shadow for. You can select from prefabs or from current scene objects.

USE ORIGINAL OBJECT

When an object from the scene is currently selected and this options is enabled, the shadow image will be captured from the current state of selected `GameObject` instead of making a copy and capturing that. This is useful, for example, if you want to create shadow texture from an animated skinned mesh mid-animation.

FREEZE BOUNDS

Usually, the bounds of all combined Renderers of the `GameObject` are used to calculate the object dimensions on the resulting shadow texture. However, in situations when bounds are changing (for example, when trying to capture an image of animated texture), this may be undesirable. While this options is enabled, object bounds won't be recalculated, and you can safely scale the view to your needs.

SELECTED MESHES

Allows you to selectively disable meshes that you don't need to appear on generated shadow texture. This is typically used for generating separate shadows for different parts of object. For example, for a helicopter you may want to generate separate images for the hull and for the blades, to allow animating them separately.

ZOOM

Allows you to zoom on the center of the object. Holding Shift while pressing the button allows for more precise zooming.

ROTATION

Allows you to rotate the object. In most cases, you'd want to produce a silhouette of an object as viewed from above. This is useful if your model has non-standard rotation.

BLUR

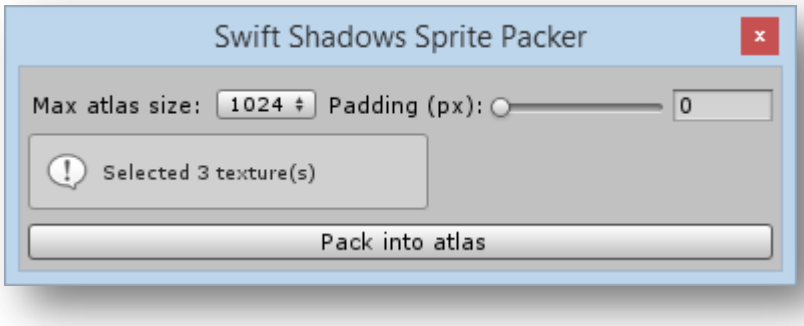
The amount of blur to apply to a shadow texture. Value of 0 means no blur, value of 50 stands for maximum available blur.

IMAGE SIZE

The resolution of shadow texture to generate when saving. Texture will always be saved as a square texture with this side, with RGBA32 texture format.

Sprite Packer

Swift Shadows also includes a simple texture packer that generates an atlas from multiple textures and automatically splits it into sprites. This is extremely useful if you want to assign sprites to your shadows, but don't want to create atlas manually or via an external solution. It can be used as a simple texture packer for other purposes just as well!



Just select all textures that you want to pack into a single texture from the Project view and hit *"Pack into atlas"* button, then select where to save the atlas, and it will be saved at that path. That's it!

Available options:

MAX ATLAS SIZE

Specifies the maximum size of the generated atlas texture. Input textures will be shrinked to fit this size if it is not possible to fit them at their original dimensions.

PADDING

Specifies the amount of pixels to leave empty between textures. This can be used to prevent textures "bleeding" one into another when atlas is scaled.

Troubleshooting

I see no shadows at all!

1. Check the **PROJECTION DISTANCE** value. Shadows will not appear beyond this distance.
2. Make sure the object you want shadow to project on has `Collider` attached. This is required for raycasting to work.

The shadow is twitching all around or appears in mid-air!
Make sure the RAYCAST LAYER MASK does not include the shadow caster itself, otherwise the shadow may cast on it as well. This usually happens when RAYCAST LAYER MASK is set to "Everything". It's usually best to put shadow casting objects on a separate layer to avoid this kind of problems too.

My shadows won't batch!
Shadows only batch if their materials and layers are the same. Also, static shadows are always rendered in a separate draw call.

In case of any troubles, enable Gizmos and select the problematic object. You should see the shadow frustum and the line of raycast. This may help you determine the source of problem.

Sometimes I'm getting an error "Component MonoBehaviour could not be loaded when loading game object. Cleaning up!"
Don't worry, this message is harmless and doesn't affects anything.

Shadows sometimes blink at the moment Play mode is entered or paused.
This is normal and happens only in the Editor. This artifact won't be present in the build of your project.

Contact

For any questions or concerns about this asset, feel free to contact me at:

Unity forums thread: http://bit.ly/1ClHBTJ
e-mail: contact@lostpolygon.com
Skype: serhij.yolkin

Changelog

1.1.1:

- Fixed an error when using “set to static if not moving” option.
- Disabled normal generation for the shadow mesh, since they aren’t used anyway.
- Fixed an error when there were no visible shadows.
- Minor UI improvements.

1.1.0:

- Fully compatible with Unity 5, WebGL, and IL2CPP.
- Shadows now work in edit mode.
- Added option of shadow size scaling depending on distance to the ground.
- Added smooth shadow rotation option.
- Added support for assigning sprites as a shadow texture. Sprites must not be packed, and the materials’ main texture must be the same as the sprite texture.
- Added a simple shadow atlas texture generator that also automatically creates sprites. Useful for assigning sprites as shadow texture.
- Rewritten shadow texture generator, handles uncommon cases much better.
- A lot of performance and GC allocation improvements.
- Having cameras that don’t render shadows doesn’t affects the performance now. Attach new `NoShadowsCamera` component on `Camera` if you want to be extra sure.
- Removed “Frame Skip” option since it wasn’t working correctly and seemingly no one used it anyway.
- Improved inspector UI.
- Dropped support for Unity 4.2 and older.
- Dropped support for Flash platform.

- Moved the code into `LostPolygon.SwiftShadows` namespace.
- Code cleanup.

1.0.0:
- Initial release.