TOXIC STOCK // PLUGIN DOCUMENTATION // V1.0 // UE 5.6

UMG SVG

Version 1.0 | Unreal Engine 5.6-5.7
by TOXIC STOCK

SVG Import · Vector UI · UMG Widget · Real-Time Preview · Resvg

TOXIC STOCK toxic.stock@toxicdev.ru

// TABLE OF CONTENTS

// 01 ACCESSING THE PLUGIN

UMG SVG integrates into Unreal Engine as an SVG asset type and a UMG widget. It provides a complete pipeline for importing, storing, and rendering scalable vector graphics directly inside the UMG UI system using the resvg rasterizer.

> Installation

- Copy the UMG_SVG folder into your project's Plugins/ directory.
- Enable via Edit → Plugins → UI / UMG → UMG_SVG.
- Restart the editor.

> Content Browser

- Import .svg files by drag & drop or via the Import button.
- Assets appear as SVG (USVGAsset) in the Content Browser.
- If SVG assets are not visible in asset pickers, enable Show Plugin Content or place assets under /Game.

// 02 SVG ASSET WORKFLOW

Each imported .svg file is stored internally as a USVGAsset. The asset holds the raw SVG text and the preferred display size derived from the SVG's intrinsic dimensions.

> Storage

SvgText	The raw SVG markup stored as a string inside the asset.
PreferredSize	Derived from the SVG viewBox, width, or height attributes.

> Reimport

Right-click a USVGAsset in the Content Browser and choose Reimport to refresh the SVG text from the source file on disk. The preferred size is recalculated automatically on reimport.

> Asset Pipeline

- Drag .svg files into the Content Browser to trigger the import dialog.
- Bulk import is supported — select multiple .svg files at once.
- Assets can be placed under any content folder including plugin content.

// 03 UMG SVG IMAGE WIDGET

The SVG Image widget renders a USVGAsset inside a UMG layout. It rasterizes the vector data to the widget's current pixel size and updates automatically whenever the widget is resized or its SVG asset changes.

> Adding the Widget

- Open a Widget Blueprint in the UMG editor.
- Search for SVG Image in the widget palette and drag it into the canvas.
- Assign a USVGAsset to the Svg property in the Details panel.

> Key Properties

Property	**Type**	**Description**
Svg	USVGAsset*	The SVG asset to render. Assign from the Details panel or via Blueprint.
Preserve Aspect Ratio	bool	When enabled, the original SVG proportions are maintained regardless of widget size.
Color And Opacity	FLinearColor	Final tint and opacity applied in Slate on top of the rasterized image.

> Runtime Behavior

The widget rasterizes to the current allocated size. Rasterization is triggered when the widget size changes or when a new SVG asset is assigned. The result is cached as a Slate brush until the next invalidation.

- Size changes (layout resize) trigger a re-rasterize.
- Changing the Svg property triggers a re-rasterize.
- Color And Opacity changes do not trigger re-rasterization — they are applied as a tint.

> Blueprint API

SetSvg(USVGAsset*)	Replace the displayed SVG asset at runtime.
SetColorAndOpacity(FLinearColor)	Change the tint color and opacity.
SetPreserveAspectRatio(bool)	Toggle aspect ratio preservation.

// 04 PREVIEW & THUMBNAILS

Both editor thumbnails and UMG design-time previews use the same resvg rasterizer as the runtime rendering path, ensuring visual consistency between editor and packaged builds.

> Content Browser Thumbnails

Thumbnails are generated automatically when a USVGAsset is imported or when the Content Browser requests a refresh. The thumbnail renderer respects the SVG's natural aspect ratio and renders at the standard UE thumbnail resolution.

- Thumbnails are rendered using the resvg C-API.
- Aspect ratio is preserved — no stretching in the thumbnail grid.
- If thumbnails appear blank, verify that resvg.dll is present in the plugin Binaries folder.

> UMG Design-Time Preview

The UMG editor preview renders SVG Image widgets using the same SSVGImage Slate widget and FResvgRasterizer pipeline as runtime. What you see in the UMG editor is pixel-accurate to the final packaged result.

// 05 RUNTIME & PACKAGING

UMG SVG uses the resvg C-API (resvg.dll / libresvg.so) for all parsing and rasterization operations. The binary is bundled with the plugin and staged automatically during packaging.

> Rasterizer

The FResvgRasterizer class wraps the resvg C-API. It accepts SVG text and a target size, then returns a CPU-side RGBA8 pixel buffer that is uploaded to a UTexture2D for Slate rendering.

> Packaging

- The resvg.dll is staged automatically for Win64 builds.
- No manual setup is required for shipping on Windows.
- Additional platforms require matching resvg pre-built binaries placed in the plugin's Binaries folder.

> Platform Support

Platform	Status	Notes
Win64	Full support	resvg.dll staged automatically. All features available.

Mac	Manual setup	Requires libresvg.dylib in plugin Binaries/Mac.
Linux	Manual setup	Requires libresvg.so in plugin Binaries/Linux.
Android	Not tested	Requires resvg cross-compiled for ARM64/ARMv7.
iOS	Not tested	Requires resvg cross-compiled as a static library.

// 06 PERFORMANCE NOTES

SVG rasterization is performed on the game thread in response to size or data changes. Understanding when rasterization occurs helps avoid unnecessary CPU cost, particularly in complex UIs.

> When Rasterization Occurs

- The widget's allocated size changes (layout resize).
- A new USVGAsset is assigned via SetSvg or the Details panel.
- The widget is first added to the viewport.

Rasterization does NOT occur when Color And Opacity changes, as tinting is applied as a Slate brush modifier without re-decoding the SVG.

> Recommendations

- Use stable, non-animated layout sizes where possible.
- Avoid driving widget size with Lerp or tick-based animation every frame.
- For animated SVG scaling, prefer discrete size steps or animate via transform instead.
- Very large or highly complex SVG files will have longer rasterization times.
- Simpler SVGs (flat paths, no filters, no complex gradients) perform best.

> Memory

Each SVG Image widget allocates one UTexture2D matching its current raster size. Multiple widgets sharing the same USVGAsset each maintain their own texture, since they may be sized differently. Pooling is not currently implemented.

// 07 LIMITATIONS

The following are known limitations in the current version. Support is determined by the capabilities of the underlying resvg library.

> SVG Feature Support

- Support depends on resvg's SVG 1.1 implementation. Some advanced SVG 2.0 features may not render.
- CSS animations and SMIL animations inside SVG are not rendered (static only).
- JavaScript inside SVG is not executed.
- External resource references (e.g. linked images inside SVG) require the assets to be available.
- Very complex SVGs with many paths, filters, or gradients may be slower to rasterize.

> Platform

- Currently targeted for Win64 out of the box.
- Additional platforms require separately compiled resvg binaries.
- Mobile platforms are untested in this version.

> Editor

- SVG assets stored under plugin content may require Show Plugin Content to be enabled in the asset picker.
- Reimport does not watch files on disk automatically — manual reimport is required.

// 08 TROUBLESHOOTING

> SVG assets not visible in asset picker

By default, the UE asset picker filters out plugin content.

- Enable Show Plugin Content in the asset picker filter options.
- Alternatively, move SVG assets into a /Game content folder.

> Thumbnails missing or blank

Blank thumbnails typically indicate a missing or inaccessible resvg binary.

- Verify that resvg.dll is present in Plugins/UMG_SVG/Binaries/Win64/.
- Confirm the plugin is enabled in Edit → Plugins.
- Re-save the asset or use Asset Actions → Reload to force a thumbnail refresh.

> Widget renders blank at runtime

If the SVG Image widget is blank in a packaged or PIE build:

- Check the Output Log for errors from FResvgRasterizer.
- Verify the resvg.dll is staged in the packaged build's Binaries folder.
- Ensure the USVGAsset is referenced correctly and not null.
- Confirm the widget's allocated size is non-zero (zero-size widgets produce no output).

> Plugin not listed in Edit → Plugins

- Confirm the UMG_SVG folder is placed directly inside <ProjectRoot>/Plugins/.
- Check that UMG_SVG.uplugin is present and valid.
- Recompile if working from source (requires UE source build or a compatible binary plugin).

// 09 TECHNICAL INFORMATION

> Architecture Overview

The plugin is structured as a single Runtime module. The core rendering pipeline flows from USVGAsset (data storage) through USVGImage (UMG widget) to SSVGImage (Slate widget) and finally FResvgRasterizer (native rasterizer bridge).

> Core Classes

Class	Type	Role
USVGAsset	UObject / Asset	Stores the SVG markup text and the natural preferred size.
USVGImage	UWidget (UMG)	UMG widget that wraps SSVGImage and exposes properties to Blueprints and the Details panel.
SSVGImage	SLeafWidget (Slate)	Slate widget that owns the rasterized texture and triggers FResvgRasterizer on invalidation.
FResvgRasterizer	Native C++ struct	Loads the resvg DLL at startup, parses SVG text, and rasterizes to a target RGBA8 buffer.

> Rendering Pipeline

- USVGAsset holds SvgText (the raw SVG string) and PreferredSize (FVector2D).
- SSVGImage receives the desired render size from the Slate layout pass.
- If the size differs from the cached size, SSVGImage calls FResvgRasterizer::Rasterize.
- FResvgRasterizer calls resvg_render() via the C-API and fills an RGBA8 buffer.
- The buffer is uploaded to a UTexture2D, which is used as the Slate brush resource.
- USVGImage exposes Color And Opacity as a tint applied over the Slate brush.

> Module Information

Field	**Value**
Module name	UMG_SVG
Module type	Runtime
Load phase	Default
Supported engine	Unreal Engine 5.6
Third-party lib	resvg (C-API, dynamic library)

> resvg Integration

The plugin dynamically loads the resvg C shared library at module startup using FPlatformProcess::GetDllHandle. Function pointers for resvg_parse, resvg_render, resvg_tree_destroy, and related utilities are resolved at load time. If the DLL cannot be found, rasterization silently returns an empty buffer and the widget renders blank.

// 10 SUPPORT & RESOURCES

> Contact

Email toxic.stock@toxicdev.ru

Plugin Version 1.0 | Unreal Engine 5.6
Module Type Runtime

> Reporting Issues

When reporting a bug or requesting support, please include:

- Unreal Engine version and platform.
- Plugin version (found in UMG_SVG.uplugin).
- A minimal repro case or the SVG file that causes the issue.
- Output Log output (filter by "UMGSVG" or "Resvg").

Use subject line [UMG SVG] when contacting by email.

> License & Distribution

TOXIC STOCK // UMG SVG // V1.0