Custom Font Creation Workflows for Unity

This guide covers two workflows for creating and testing custom fonts for Unity game projects. It is designed for users with little or no prior experience with font design software.

Workflow 1 – Designing a Font in Glyphr Studio and Testing in Unity

This workflow explains how to design a simple vector font using Glyphr Studio and import it into Unity for testing with TextMesh Pro.

1. Open Glyphr Studio at https://www.glyphrstudio.com/ (it runs in your browser, no installation needed).
2. Create a new project and choose the basic Latin character set for simplicity.
3. If you already have a PNG image or tilemap containing your letters, open it in a vector graphics editor such as Inkscape.
4. In Inkscape: trace each character to convert it to vector outlines (Path → Trace Bitmap). Clean up the outlines as needed.
5. Export each traced letter as an individual SVG file. Name them logically (e.g., A.svg, B.svg, etc.).
6. In Glyphr Studio, open the project and for each character, click ‘Import SVG’ to bring in your pre-made letter shapes.
7. Adjust size and alignment as needed. Use the side panel to fine-tune spacing and baseline alignment.
8. When finished, go to File → Export Font → choose 'TTF' and save the font file.
9. In Unity, open your project and import the exported .ttf file by dragging it into the Assets folder.
10. Right-click the imported font and select ‘Create → TextMeshPro → Font Asset’.
11. Add a TextMeshPro Text object to your scene (GameObject → UI → Text - TextMeshPro).
12. Assign your new font asset to the TMP Text object’s Font Asset field and type some sample text to test it.

Tips:

You can use Inkscape’s 'Trace Bitmap' tool to turn a single image or tilemap of letters into editable SVGs. Glyphr Studio only supports vector input, so this step avoids redrawing every character manually.

Workflow 2 – Preparing Detailed or Shaded Glyphs for Unity

For more detailed or shaded fonts, you’ll need to use bitmap or layered color workflows. This allows you to preserve artistic shading, depth, or texture that SVG-based fonts can’t represent.

1. Design your glyphs (characters) in an art program such as Photoshop, GIMP, or Krita. Each letter should be a separate tile or image with transparent background.
2. Keep all glyphs the same size and alignment. Arrange them in a grid layout if possible for easier import later.
3. Export your entire tilemap as a single PNG image, ensuring all glyphs are evenly spaced.

4. Use an online bitmap font generator such as Littera (https://snowb.org/littera/) to map your characters.
5. In Littera: upload your PNG, define the grid size (tile width/height), and enter the character mapping (e.g. ABCDEFG…).
6. Export as .fnt (AngelCode BMFont format) and .png texture atlas.
7. In Unity, import both the .fnt and .png files into your project's Assets folder.
8. Open the TextMesh Pro Font Asset Creator (Window → TextMeshPro → Font Asset Creator) and generate a new TMP Font Asset using your .fnt and texture.
9. Test the font in a simple TextMeshPro object. It should retain your shading and detail from the original art.
10. For advanced users, you can combine this with materials or shaders to enhance lighting or color effects.

Tips:

Bitmap fonts are ideal for stylized, hand-drawn, or pixel-art fonts where you want to preserve texture and shading. They do not scale cleanly, so use them at their designed resolution.

Summary

• Use Glyphr Studio for clean, scalable vector fonts (you can import traced letters from PNG via Inkscape). • Use Littera or similar bitmap font tools for detailed, shaded, or artistic fonts. • Test all fonts in Unity using TextMesh Pro to ensure correct spacing, alignment, and visual quality.