LAYERED

USER MANUAL

CONTENTS

INTRODUCTION

LAYERED is a Unity Editor tool that makes it easy to create orthographic 2.5D scenes out of PSD files. Layers, Groups, and their layout transfer into Unity seamlessly. Once in Unity, padding can easily be added between layers to make room for particle effects and 3D models. Raster layers without a Vector Mask are imported as Sprites, and ones with Vector Masks are imported with a Mesh. Type Tool text is imported as TextMeshPro text.

Features

- Imports art, mask, vector mask, layout, and hierarchy of a PSD file into Unity.
- Layers are imported as PNGs.
- Images are padded to the nearest multiple of 4 so they meet compression requirements.
- Layers with a Shape or Vector Mask automatically export as an image/mesh/material combination.
- Layers without a Vector Mask automatically export as a sprite.
- A layer can be overridden to become a: sprite, mesh/material, text, and even a transform.
- Vector Mask smoothness can be adjusted in Unity after import.
- Grayscale masks automatically merge with the layer.
- Only saves images that have changed.
- Adjusting the depth between layers is dynamic and done easily from inside the Editor.
- Properties shared between Photoshop and Unity can be modified and reverted at any time.
- Uses Pixels Per Unit to size the document in Unity.
- Layer and Group pivots can be adjusted inside the Unity Editor.
- The source PSD file can live anywhere.

Compatibility and Technical Details

- Not compatible with any Unity UI systems.
- Only works with orthographic cameras and orthographic sorting.
- Uses Z positions to sort layers, not *Sorting Layers*.
- Only supports importing from the *Editor*.
- See Compatibility for PSD related compatibility.

QUICK START

Scene Preparation

1. Start a new Unity project.
2. Select the Camera and change the *Projection* mode to *Orthographic*.
3. Set the Camera *Size* field to 0.005 times the height of the PSD. If the height of the PSD is 1080, then that comes out to *0.005 * 1080*, which is *5.4*.
4. Set the *Near Clipping Plane* to -10, and the *Far* to 10.

Create a MetaDocument asset

5. Create a folder the same name as the PSD in an appropriate location in the Unity Assets folder.
6. Right click on the folder and select: *Create-> Layered ->MetaDocument*
7. The *ScriptableObject* can be any name, but we'll name it after the PSD for this example.
8. With the *MetaDocument* selected, click the "..." button and navigate to the PSD document you want to import. If you don't have one, there's a sample PSD called "Vector" located under Samples.
9. Leave *Pixels Per Unit* at 100.

Create a LayoutDocument GameObject

10. Right click on the *Camera* in the *Hierarchy* and select *Layered->LayoutDocument*.
11. The *GameObject* can be any name, but we'll name it after the PSD for this example.
12. With the *LayoutDocument* selected, drag the *MetaDocument* created in step 6 into the *Meta Document* field.
13. Click the *Import* button.

PSD COMPATIBILITY

Document

Supports RGB/RGBA 8-bit color PSD files encoded with sRGB. Images stored internally must either be RAW or RLE (both are typical). Zip compression is currently unsupported but is uncommon anyhow.

NOTE: PSD files exported from other paint programs do not always support all PSD features.

Group

A Group is imported as a Transform.

Raster Layer

A rasterized layer or a Smart Object layer that does not contain a *Vector Mask* will automatically be imported as a Sprite PNG image.

Shape Layer

Shape layers will automatically be imported as Meshes and Materials. Solid colors, gradients, and patterns are all rasterized and saved as PNG images.

The rasterized image fits inside the vector contour. That means inside strokes will get captured as part of the image, mid strokes will be cut in half, and outside strokes be cut out completely.

NOTE: Once inside Unity, zooming in will reveal that the pixels that have been anti-aliased straddle the contour. It's not very noticeable when the game view resolution is high, but it's very noticeable when the resolution is low. If the image is meant to be fully opaque, and you don't mind aliasing, then switching the shader to an opaque one helps it to look better.

Vector Mask

Raster layers that contain a Vector Mask will automatically be imported as Meshes and Materials. Vector Masks can be used for several things, like cutting large portions of unused alpha, used as a Mesh to emit particles from, and used for stenciling.

Mask

A raster layer that has a Mask applied to it will automatically be merged with the underlying image on import. This goes for layers with or without a Vector Mask.

NOTE: A *Vector Mask* applied to *Type Tool* text is not supported. Text will need to be rasterized for the *Vector Mask* to work.

Type Tool

A Type Tool layer will be imported as TextMeshPro text. There are some differences between the two, so the conversion isn't exact, but still very close.

Text features that are generally shared between Photoshop and TextMeshPro are:

- Color (only accurate when the project uses *Gamma Color Space*, otherwise the color is desaturated)
- Left, middle, and right justification
- Character spacing
- Line spacing
- Vertical scale (limited to single lines)
- Horizontal scale

NOTE:

- To get the closest possible match between Photoshop and TextMeshPro, the same font must be used in both places.
- All text will use the style applied to the very first run of characters.

Layer Opacity

Supports Layer Opacity

Normal Blend Mode

Supports the "Normal" blending mode.

Unsupported

If a feature hasn't been mentioned above, it is unsupported.

OVERRIDES

Most of the time letting LAYERED autodetect what kind of assets to import works just fine, but sometimes it's necessary to override the default behavior.

Overriding is possible by adding a "|" (pipe) symbol followed by a *Substrate Override* and/or optional *Occlude* flags to the layer/group name.

The format is: Name|SubstrateOverride -Occlude -Occlude -Occlude

NOTE: If you use a *SubstrateOverride* to force a Shape Layer or a layer with a Vector Mask to become a *Sprite*, a mesh will not be generated. Instead, LAYERED lets Unity generate the mesh.

Name

The name of the layer or group. The GameObject and Asset will inherit this name.

IMPORTANT: Names with **invalid file name characters** will be ignored. Also, siblings with **duplicate names** will cause the import to abort with an error notifying the reason.

Substrate Override

Layers:

- **mesh**: creates a *Mesh and Material,* and is assigned a shader that renders transparent, or opaque, depending on if the image has alpha or not.
- **sprite**: creates a *Sprite* that uses the default sprite material.
- **text**: creates a *TextMeshPro* text area the size of the art boundary of the layer.
- **transform**: creates a *Transform* with its position set to the center of the art boundary.

Groups:

- **normalize**: causes a group to crop each child layer to the combined size of all immediate child layers.

Occlude

One or more of the following options can be used for layers to keep certain objects from being created:

- -**m**: short for "mesh" and will keep a mesh from being created.
- -**i**: short for "image" and will keep an image from being created.
- -**s**: short for "substrate" and will keep a *GameObject* from being created.

Shortcuts

The following shortcuts may be used in place of *Substrate Override* and *Occlude* combinations:

- **meshonly** (mesh -i -s): only creates a *Mesh* asset and not a *GameObject* or *Image*.
- **imageonly** (mesh -m -s): only creates an *Image* asset and no *GameObject* or *Mesh*.
- **spriteonly** (sprite -s): only creates a *Sprite* asset and no *GameObject*.
- **blankmesh** (mesh -i): creates a *GameObject* with a *MeshRenderer* and *Material* but no *Image*.
- **blanksprite** (sprite -i): creates a *GameObject* with a *SpriteRenderer* but no *Image*.

SCRIPTS

LAYERED utilizes four user-facing scripts. A *MetaDocument* that points to a PSD, a *LayoutDocument* that does the importing, and *LayoutLayer*/*LayoutGroup* that are attached to layers and groups, respectively.

MetaDocument

Links to a PSD file by relative path. It also stores meta information about the PSD file along with formatted data pertaining specifically to LAYERED. Its data updates on import.

LayoutDocument

This script is responsible for importing a PSD file. It takes a reference to a *MetaDocument*, and contains a property that sets *Z Padding*. Z Padding adds depth between layers along Z so that layers draw in the correct order.

LayoutLayer

This script is placed on GameObjects for every layer imported from a PSD. Its properties will vary depending on the underlying *Substrate* the layer was identified as. Here's the list of properties:

- **Pivot:** Pivot adjustment for all layer types. Values range from 0 to 1. For x, 0 is the leftmost of the art, and 1 is rightmost. For y, 0 is the topmost of the art, and 1 is bottommost.
- **Resource:** Tells assets if they should live in a Resources folder or not.
- **Contour:** A set of properties that modify the smoothing of a Vector Mask.
- **Padding:** *Hold Z*, *Front* and *Back* padding.
- **Revert:** Context related buttons that revert values to their PSD equivalent values.

LayoutGroup

Like *LayoutLayer*, but for groups.

- **Pivot:** Pivot adjustment for all layer types. Values range from 0 to 1. For x, 0 is the leftmost of the document, and 1 is rightmost. For y, 0 is the topmost of the document, and 1 is bottommost.
- **Padding:** *Front* and *Back* padding.
- **Revert:** Context related buttons that revert values to their PSD equivalent values.

PIVOT

Group and layer pivots can be adjusted once inside Unity. The pivot is normalized, which means the values range from 0 to 1 in both x and y directions.

- **Layer:** (0, 0) is the bottom left corner of the *layer*, and (1,1) is the top right corner of the *layer*.
- **Group:** (0, 0) is the bottom left corner of the *document*, and (1, 1) is the top right corner of the *document*.

There can be up to two buttons on the right of the pivot. 'D' stands for *Center of **D**ocument*, and 'L' stands for *Center of **L**ayer*, or ***L**ayers* if it's a group.

Changing these values will cause the pivot to preview in the Scene View.

Once the pivot has been set, press the *Apply* button to apply changes, or *Revert* to revert them.

CONTOUR

The contour of a Mesh that came from a *Shape Layer* or *Vector Mask* can be adjusted from *LayoutLayer*.

- **Subdivisions:** the number of subdivisions between control points.
- **Angle Deviation:** the amount the angle can deviate before a control point is added to a curve.
- **Optimize Straights:** keeps straight lines from being subdivided.

Changing these values will cause the curve to preview in the Scene View. The line color alternates between each subdivision so it's easy to tell where the curve has been subdivided. The colors of the lines can be altered under *Edit->Preferences->LDG->Layered*.

Once properties have been set, press the *Apply* button to apply changes, or *Revert* to revert them.

NOTE: Applying these properties directly affects the Mesh asset itself, so any update here will affect all other *LayerDocuments* that point to the same *MetaDocument* and Mesh.

PADDING

Padding can be added between layers along z so that objects with depth can fit in between. There are three types of padding: Z Padding, Padding Back (side opposite of camera), and Padding Front (side closest to camera).

Z Padding

Z Padding is set in the *LayoutDocument* and gets applied to every *LayoutLayer* that is imported. This is the primary way that layers are sorted and drawn. If sorting by z is not desired, then set this value to 0 and manually set Sorting Layers instead.

Hold Z

Keeps the *Z Padding* value located in *LayoutDocument* from being applied to a *LayoutLayer*.

NOTE: There isn't a *Hold Z* for *LayoutGroup* since they are not relevant for groups.

Padding Back

Adds padding to the back of a *LayoutLayer*/*LayoutGroup*. Back is considered the side opposite the camera.

Padding Front

Adds padding to the front of a *LayoutLayer*/*LayoutGroup*. Front is considered the side facing the camera.

ASSETS

Assets are categorized into three types: Image, Mesh, and Material. Each type of asset is saved into a hierarchy of folders that mimic the corresponding groups in a PSD. All files and folders are created relative to the *MetaDocument*.

Example - Without Group

- If there's a layer called "Mountainscape" with a *Vector Mask* applied to it, it would be saved to the following locations relative to the *MetaDocument*:
    - "Images\MountainScape.png"
    - "Meshes\Mountainscape.mesh"
    - "Materials\Mountainscape.mat"

Example - With Group

- If there's a group called "Animals" with a layer called "Cat", and a *Vector Mask* applied to it, it would be saved to the following locations relative to the *MetaDocument*:
    - "Images\Animals\Cat.png"
    - "Meshes\Animals\Cat.mesh"
    - "Materials\Animals\Cat.mat"

Duplicate Names

Duplicate names under the same group or root will cause the importer to abort.

Invalid File Name

Group and layer names must follow file naming conventions. This is because assets and folders inherit layer and group names.

Predefined Default Values

When LAYERED saves an asset, it is saved with predefined default values. These predefined default values can be altered from *Preferences->LDG->Layered*. Preferences are stored with the user and not with the project.

Q&A

How do I get the contour of a vector to show?

- Contours only display when their properties are being adjusted.
- If a property has changed, and it's still not showing, be sure to enable Gizmos.

How do I get the Pivot point to show?

- Pivots only display when their position is being adjusted.
- Click and drag from the ring that is part of the transform.
- If the position has changed, and it's still not showing, be sure to enable Gizmos.

Is there a way to adjust or turn off the Hierarchy Icons?

- Yes. Navigate to **Edit->Preferences->Layered->Visuals->Hierarchy**. From there the position of the icon and visibility can be adjusted.

Why do text colors look incorrect?

- Text color looks most accurate when the project is set up to use *Gamma Color Space*.

Unfortunately, as of the time of this writing, the 3D version of TextMeshPro makes no effort to convert *Linear* colors to *Gamma* when the project is using a *Linear Color Space*. Unity just assumes the color needs to be linear just like the lighting applied to meshes.

Why is text missing?

- TextMeshPro is required. If it is not automatically installed, then install it from the package manager. Go to **Package Manager->Unity Registry->TextMeshPro**, or **Package Manager->Unity Registry->uGUI** for newer versions of Unity.

SUPPORT, CONTACT, AND LEGAL INFORMATION

For additional assistance visit the contact page at www.littledreamergames.com/contact, or email help/questions to support@littledreamergames.com.