# Figma Converter for Unity

## for developers

1.0.7

D.A. Assets

# Contents

# 3. Introduction

Before starting to read the next part of this guide, I will tell you about **some of the features** in working with this asset.

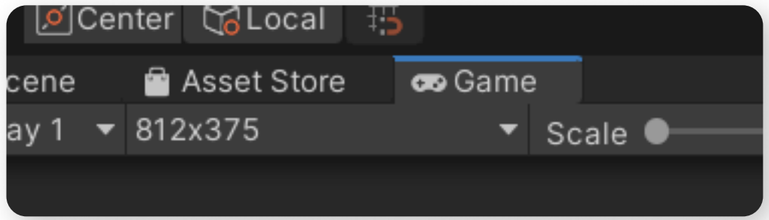
It has a lot of customization options, but if you want to quickly try out its functionality, you can skip straight to section "**Auth**".

If your project contains **many components** related to UI components, and are not frames, such project will **not be suitable for import** due to the large weight of the json file that figma gives - the file size can reach 100 megabytes and its download and parsing can take a long time.
I recommend transferring the frames you need to a **separate project** and importing it.

This asset can work in conjunction with other graphics assets using their capabilities.
There is currently support for the following assets: **True Shadow - UI Soft Shadow and Glow, Modern Procedural UI Kit, Procedural UI Image and I2Localization.**
In order to work with these assets, you need to **buy** them from the **Asset Store** and **import** them **into** your **project**.
After you have done this, carefully **read** both **manuals** - for developers and for designers.
In the contents of these manuals, you will find **page numbers** for information on the **use** of these **assets** and their corresponding **tags**, if any.

# 4. Scene Setup

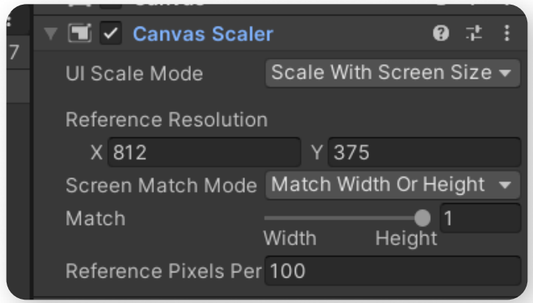
1. Set the screen size in the "**Game**" tab according to the size of your frame.




2. Specify the same size in the "**Reference Resolution**" in the "**Canvas Scaler**" script (it is on the canvas).

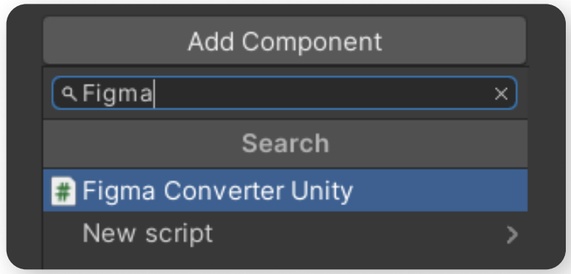
3. Set “**UI Scale Mode**” to “**Scale With Screen Size**”.

4. Set the “**Match**” parameter to the maximum, or at your discretion.

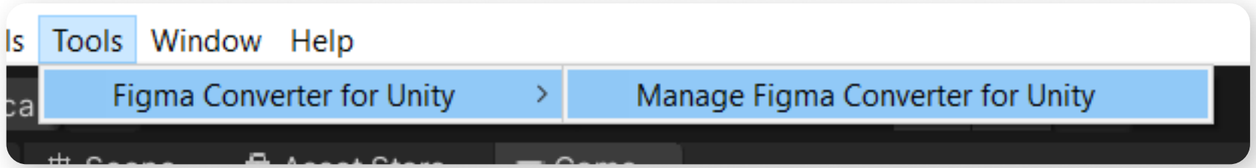


# 5. Options for adding a script to the scene

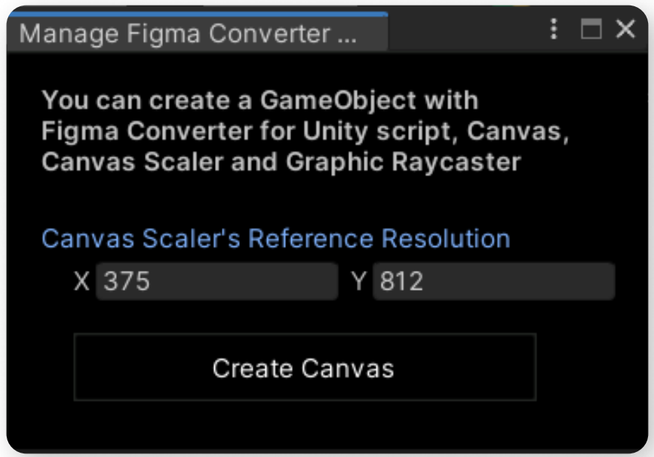
1. Add “**FigmaConverterUnity**” script to your **Canvas.**




2. You can also create a GameObject with a Canvas and all the necessary scripts through the **Tools** menu.




3. Specify the **reference resolution** and create a canvas with the “**FigmaConverterUnity**” script already added.



# 6. Installing Json.NET

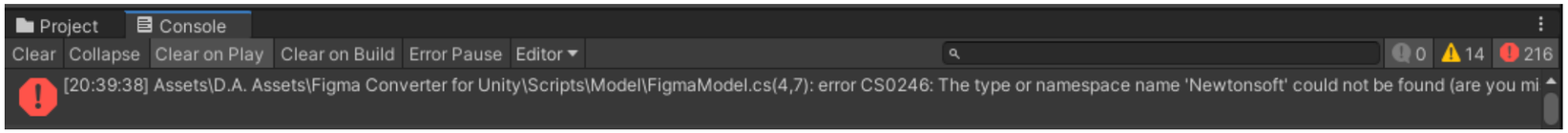
The asset has only one required dependency, and that is "**Json.NET**". Without this library, the functionality of the asset is not available in the inspector.

JSON.NET is automatically installed when the asset is installed through Package Manager, although it is not visible there.

1. To activate **JSON.NET**, set parameter "**JSON_NET_EXISTS**" to "**ENABLED**".


2. After installing "**JSON.NET**", the asset will be **ready** for further configuration and working.
A detailed description of installing other dependencies of the asset is in section "**Scripting Define Symbols**".

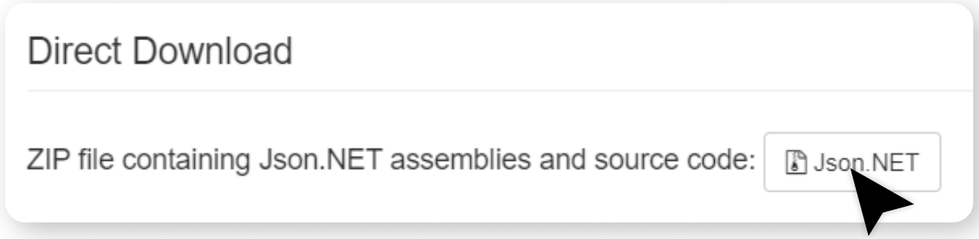
3. If for some reason you see an error "**Newtonsoft could not be found**", then follow the instructions on **slide 6.1.**



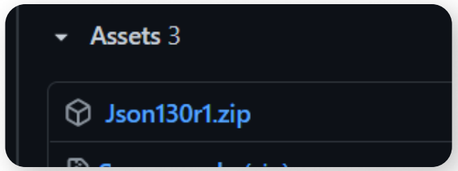
# 6.1. Install Json.NET

1. Go to the **json.net** site and press **download** button.

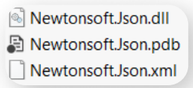

2. Select direct download and click “**Json.NET**” button.




3. Download and open the zip archive from the **latest** release.

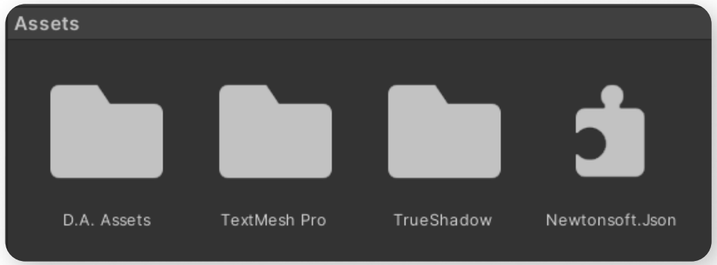



4. **Extract** dll "**Newtonsoft.Json.dll**" from the archive along the path "**/Bin/netstandard2.0/**".



# 6.2. Install Json.NET

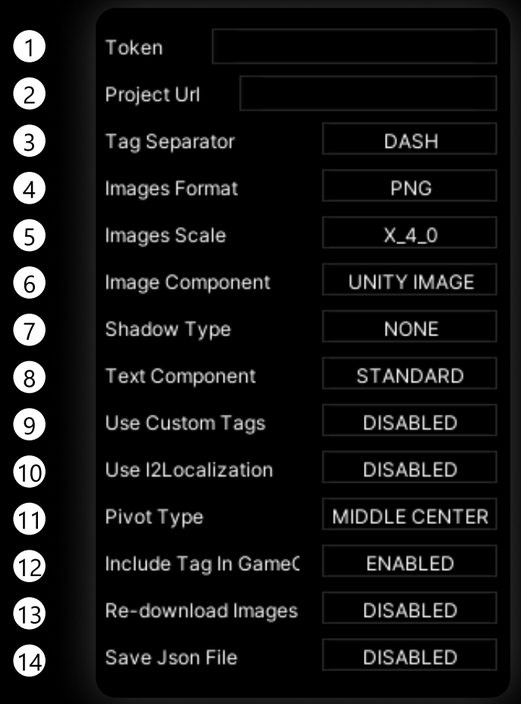
5 **Import** dll into your **Unity** project.




6 To activate **JSON.NET**, set parameter "**JSON_NET_EXISTS**" to "**ENABLED**".

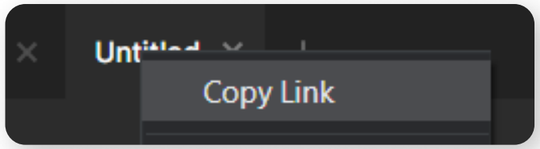

7 After installing "**JSON.NET**", the asset will be **ready** for further configuration and working.
A detailed description of installing other dependencies of the asset is in section "**Scripting Define Symbols**".

# 7. Main Settings




1. Your figma api key. You can get it by clicking on the "**Auth With Browser**" button, the key will be received automatically. A detailed explanation is in the "**Auth**" section of this manual.

2. Link to your project in figma. It can be obtained by **right-clicking on the tab** with an open project.
**Example link**: https://www.figma.com/file/XXXXXXXXXXXXXXXXXXXXXXXXXXXXX...





3. A delimiter type that **separates** the component **tag** that the asset uses to **determine** its **type from** the **rest** of the component name.
You can use "**DASH**" - "**-**", or "**SLASH**" - "**/**". **Examples** are described **in** the **designer's manual**.

4. The format of the downloaded images. It can be **png** or **jpg**.

5. The size of the exported images. This option is identical to the same option when exporting from Figma.

6. You can choose a component that will be created on the stage as an element of your UI. This can be the standard "**UnityEngine.UI**", as well as "**MPUIKIT.MPImage**" and "**UnityEngine.UI.ProceduralImage**" if you have the appropriate **assets installed** - you can learn **more about** this on **page 8 and 9**.

7. The type of shadow for the object. Shadow options are exported from figma. Disabled by default, but you can use the "**Tai's True Shadow**" asset. You can read **more about** this on **page 7**.

8. The component to be used when exporting the figma text components. You can choose the standard **Unity text**, or **TextMeshPro**.

# 7.1. Main Settings

9 With this function, you can set **your own tags** and **bind existing objects** to them in the form of prefabs or scene components, after which, **based on** the **layouts** in Figma, they will be **drawn on** your **scene.**
You can learn **more about** this on **page 10**.

10 This function automatically **adds** a **localization script** from “**I2Localization**” asset and **generates** a **key** for them **based on** the **name** of the text component in **Figma**, after which it **saves** the current **text** and its **key** as an english localization in an automatically **generated** localization **file**.
This function only **works if** you have “**I2Localization**” asset **installed.**
You can learn **more about** this on **page 12**.

11 The **pivot** of the objects exported to the scene. A detailed explanation is in the "**Pivot**" section of this manual.

12 **Adds** a special **tag** to the names of your game objects to indicate the type of component.
**Optional**. Enabled by default.

13 Check this option if the figma layout elements have **changed since the export** and you need to **overwrite** them when you **re-export**. Also, you can delete the changed images manually and start exporting from scratch with the option disabled, in this case, the existing images will not be overwritten, but the images that have been deleted will be downloaded if they are still in the figma layout.

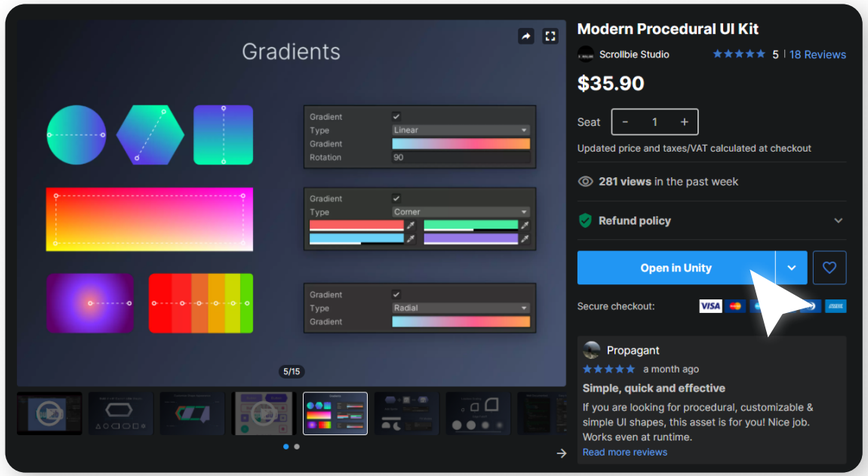
14 Option for developers. Saves a json file with a response from the figma api.

# 8. Modern Procedural UI Kit

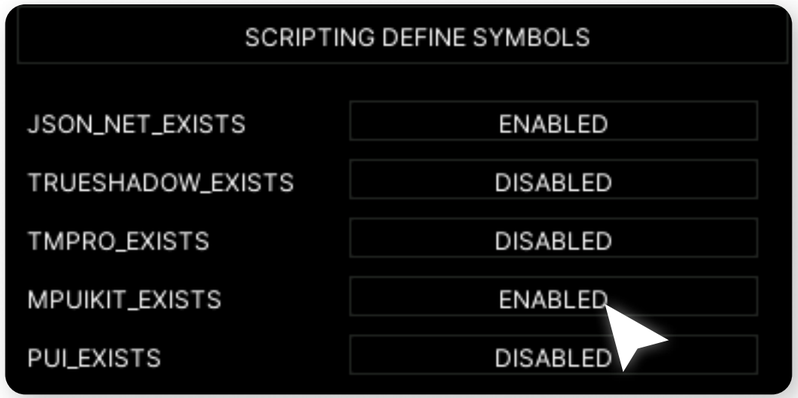
You can purchase **Modern Procedural UI Kit** asset and use it in conjunction with **Figma Converter for Unity**.

This asset supports **linear gradients** as well as transfers **rounding corners** of images.

1. Add the purchased asset to your project.




2. Switch "**MPUIKIT_EXISTS**" to "**ENABLED**".



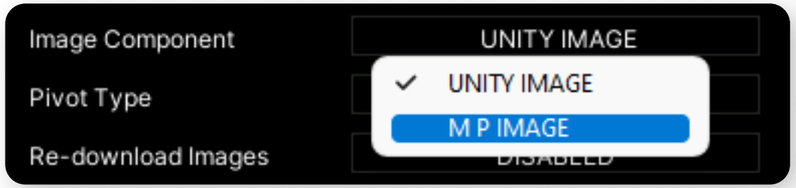

D.A. Assets

# 8.1. Modern Procedural UI Kit

Components with **supported gradients** and **fills** are **recreated** in Unity using an **MPUIKit** asset, after which the downloaded **images** related to them are **removed** from the sprite folder.
Components with **unsupported gradients** are imported as **regular sprites** inside component **MPImage**, while their images **remain** in the **sprite folder** in a subfolder with the name of the frame on which they are located, or in the folder with shared frame elements - "**Mutual**".



3 After loading the project after completing the previous paragraph, in the **Main Settings** of the asset you can **change** the **component** of the image, which is used to **draw images** on your scene.
Select "**MPImage**".




4 In order for your project to be **imported correctly** using this asset, you **need** to **read** additional information regarding working with this asset in the **Manual for designers**:
*Assets\D.A. Assets\Figma Converter for Unity\Figma Converter for Unity - Manual for designers.pdf*

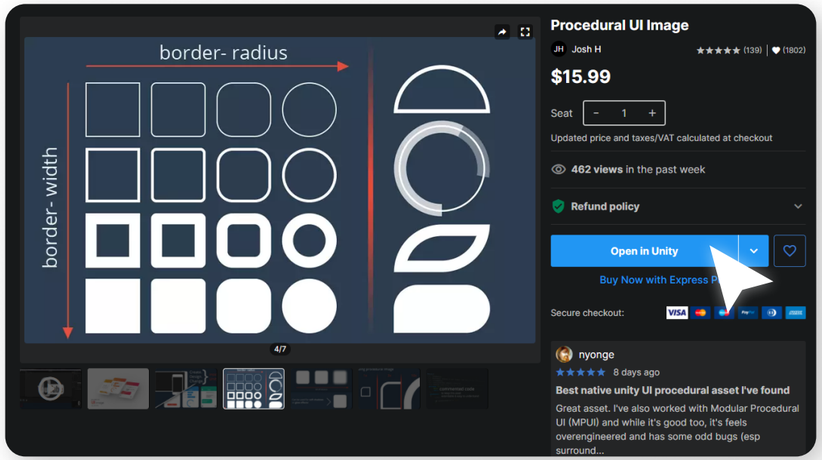
After that, you can **continue to import** your Figma project as usual, but instead of the standard image component in the scene, you will have "**MPImage**".

# 9. Procedural UI Image

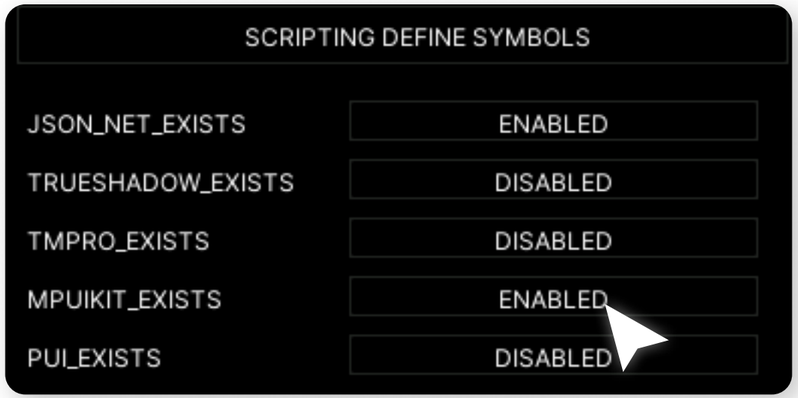
You can purchase **Procedural Image** asset and use it in conjunction with **Figma Converter for Unity**.

This asset **does not support gradients**, but it does **support** simple **fills** and **corner radius**. Images imported from Figma, consisting of a single-color **fill**, will be **removed** from the **Sprites folder**.

1. Add the purchased asset to your project.

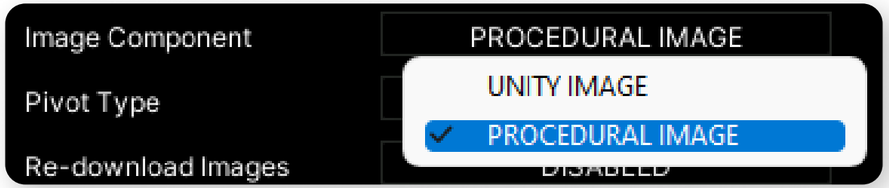



2. Switch "**MPUIKIT_EXISTS**" to "**ENABLED**".



# 9.1. Procedural UI Image

3. After loading the project after completing the previous paragraph, in the **Main Settings** of the asset you can **change** the **component** of the image, which is used to **draw images** on your scene.
Select “**Procedural Image**”.



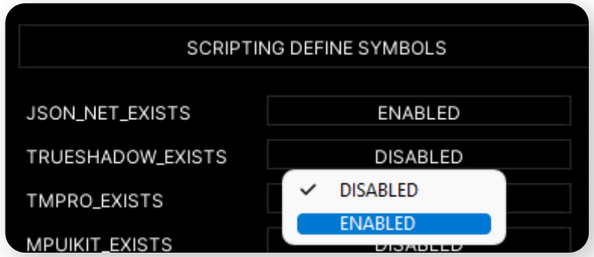

4. In order for your project to be **imported correctly** using this asset, you **need** to **read** additional information regarding working with this asset in the **Manual for designers**:
*Assets\D.A. Assets\Figma Converter for Unity\Figma Converter for Unity - Manual for designers.pdf*

After that, you can **continue to import** your Figma project as usual, but instead of the standard image component in the scene, you will have "**Procedural Image**".

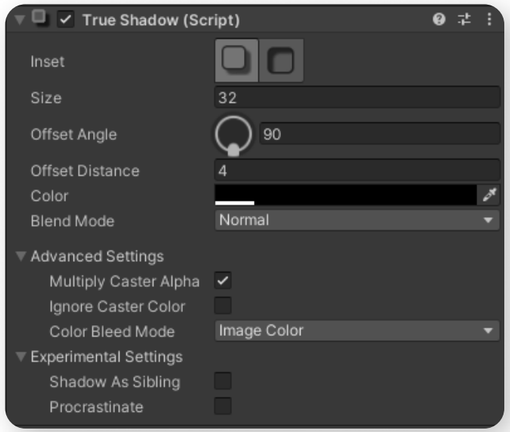
# 10. True Shadow

**TrueShadow** asset most
**closely** simulates
the **shadows** from **Figma**.

1. **Import** asset into your **project** and then switch "**TRUESHADOW_EXISTS**" to "**ENABLED**" in "**SCRIPTING DEFINE SYMBOLS**" section.

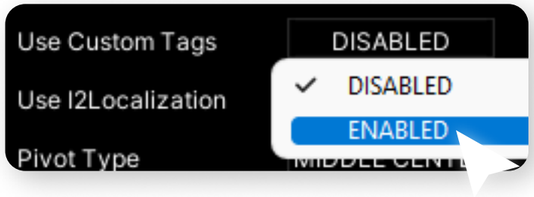



2. After import, **all** your **components** that have a **shadow** in the Figma layout **will have** a **shadow** script from "**TrueShadow**" asset.
**To use** this functionality properly, **read** the section on this asset in the **designer's manual**.

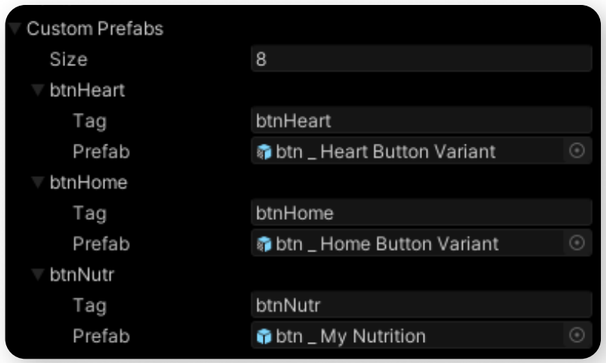


# 11. Custom Tags

1. To activate this function, find item “**Use Custom Tags**” in the **main settings** and switch it to “**Enabled”**.



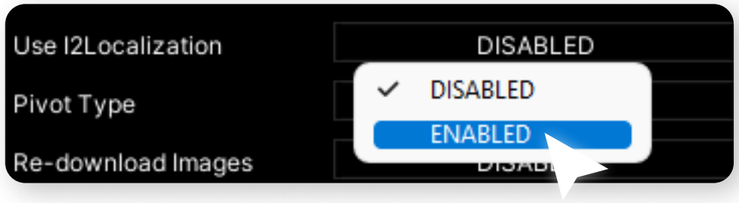

2. After activating the function, an array will appear at the bottom of the asset, in which you can **add prefabs** and **set tags** for them.




3. Once you've **fill** your **array**, you can **use** your **tags** in **Figma** just **like** any **other tags**.
**After importing** the project using the asset, your **prefabs** will be **drawed in** the **scene** according to the tag in the Figma layout, and their size will correspond to the component of the Figma layout that is assigned the corresponding tag.

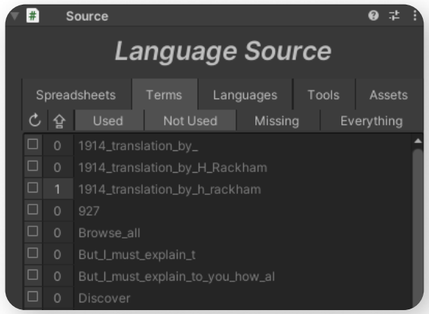
# 12. Localization

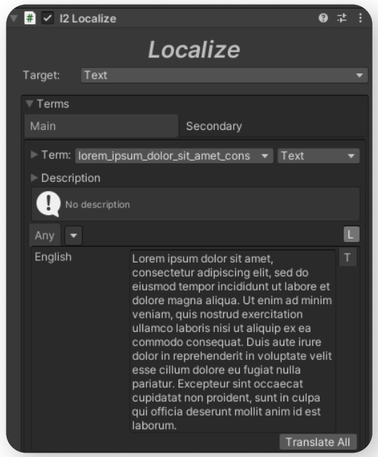
1. To activate this function, find item “**Use I2Localization**” in the **main settings** and switch it to “**Enabled”**.

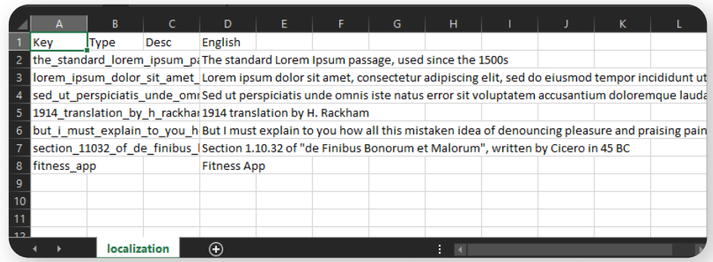



2. Import your layout as you normally would.

3. **After import**, script “**I2Localize**” will be **added to all text components**, their text will be written to the localization file "**localization.csv**" (**separator** - **semicolon**) as an **english** localization, the localization corresponding to the text component will be **selected** in the script.
You can open the localization file in **Excel**.
All further **instructions** are detailed in the **manual** for asset **I2Localization**.

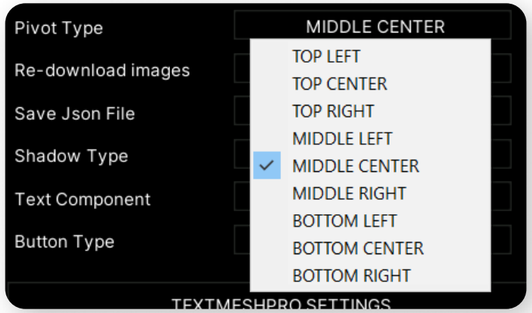


# 13. Pivot

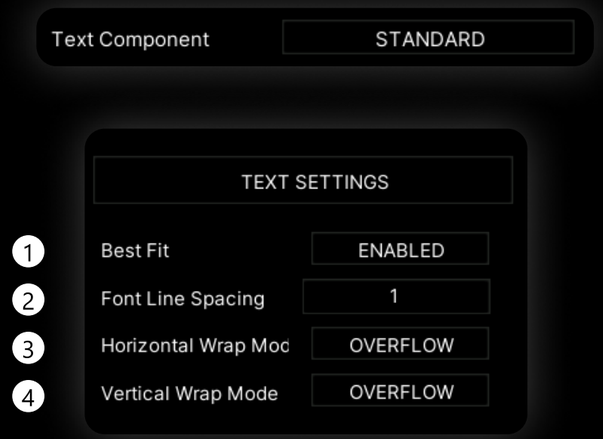
1. **Pivot** is the offset of the local center of the object relative to the world center. With this parameter you can set the **Pivot** for all imported objects. **MIDDLE CENTER** is a default **Pivot** value. What pivots are supported in **Unity**:




2. Table with the ratio of **string names** of **pivots** and **numerical values**:

| Name | Value 1 | Value 2 |
|---|---|---|
| TOP LEFT | 0 | 1 |
| TOP CENTER | 0.5 | 1 |
| TOP RIGHT | 1 | 1 |
| MIDDLE LEFT | 0 | 0.5 |
| MIDDLE CENTER | 0.5 | 0.5 |
| MIDDLE RIGHT | 1 | 0.5 |
| BOTTOM LEFT | 0 | 0 |
| BOTTOM CENTER | 0.5 | 0 |
| BOTTOM RIGHT | 1 | 0 |

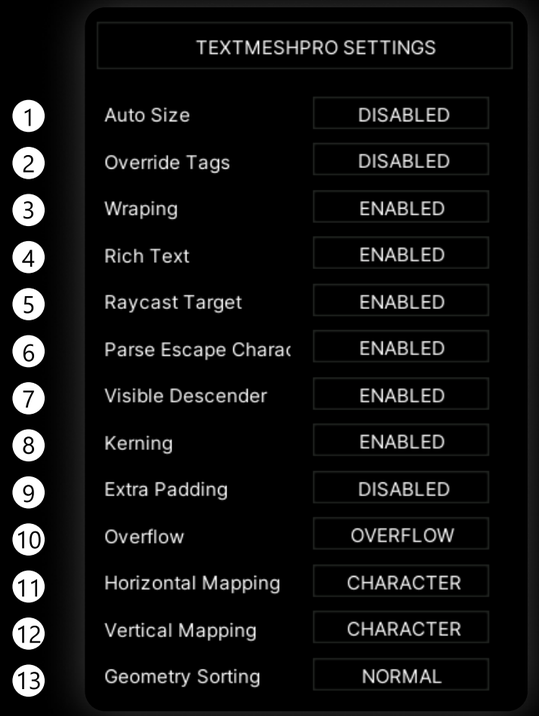
# 14. Standard Text Settings




You can choose which text component to use when importing the figma layout. These are the settings for **standard** (Unity.UI) text.

1. Should Unity ignore the size properties and simply try to fit the text to the control's rectangle?
2. The vertical separation between lines of text. This value is **unique** for each font, so the imported text may look **different** from Figma, you will need to select this value **manually**.
3. The method used to handle the situation where the text is too wide to fit in the rectangle. The options are Wrap and Overflow.
4. The method used to handle the situation where wrapped text is too tall to fit in the rectangle. The options are Truncate and Overflow.

# 15. TextMeshPro Settings




1. Auto sizes the text to fit the available space.
2. Whether the color settings override the <color> tag.
3. Wraps text to the next line when reaching the edge of the container.
4. Enables the use of rich text tags such as <color> and <font>.
5. If selected, the UI element will block the raycast (i.e. your mouse click). If you don't have it checked, then your mouse will not work on that button, and the next thing below/behind it will catch the raycast.
6. Whether to display strings such as \"\\n\" as is or replace them by the character they represent.
7. Compute descender values from visible characters only. Used to adjust layout behavior when hiding and revealing characters dynamically.
8. Enables character specific spacing between pairs of characters.
9. Adds some padding between the characters and the edge of the text mesh. Can reduce graphical errors when displaying small text.
10. How to display text which goes past the edge of the container.
11. Horizontal UV mapping when using a shader with a texture face option.

# 15.1. TextMeshPro Settings

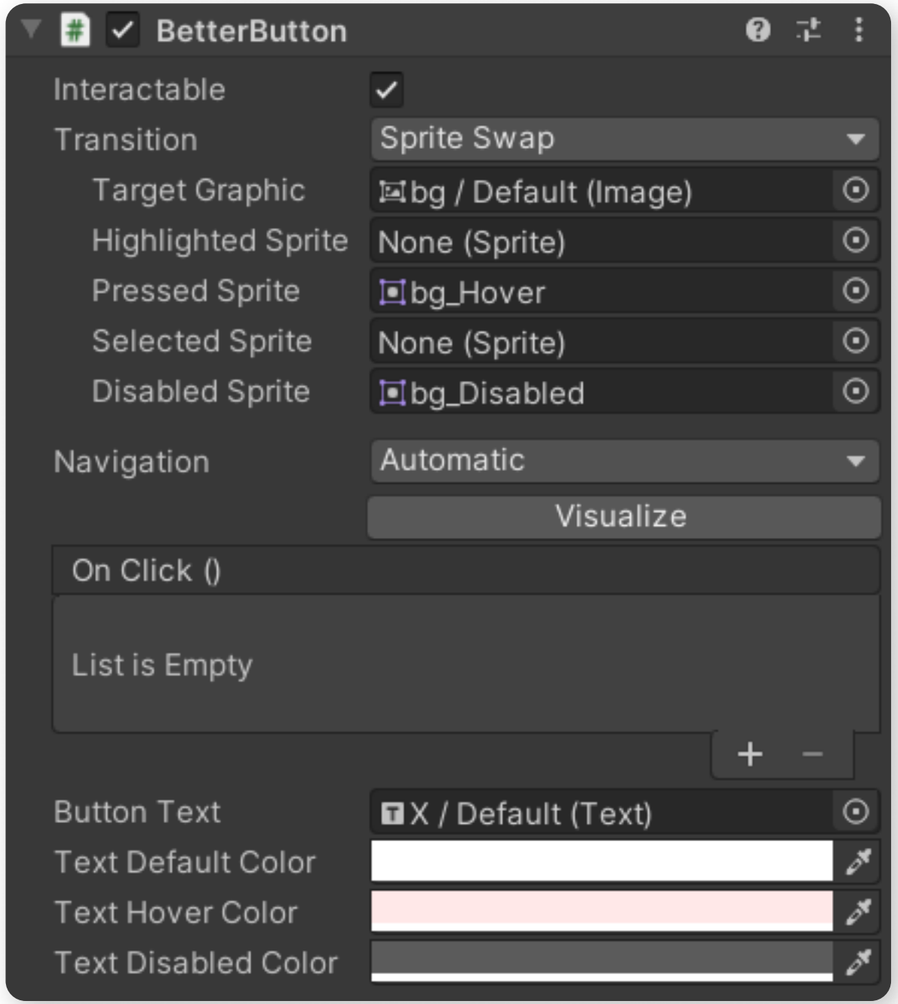
You can choose which text component to use when importing the figma layout. These are the settings for "", which you can find in the package manager and set to the project so that the settings for it can be displayed after switching the text component from standard to "".

12 Vertical UV mapping when using a shader with a texture face option.

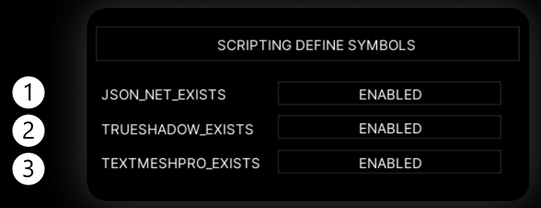
13 The order in which text geometry is sorted. Used to adjust the way overlapping characters are displayed.

# 16. BetterButton

This component is similar in every way to a button, but it supports changing the color of the text in the "**Sprite Swap**" mode. For example, if you click on such a button and your button sprite changes, the text color will also change. This component is used instead of a button automatically if the button in the figma layout has a special structure - see the **designers guide** for more details.



# 17. Scripting Define Symbols

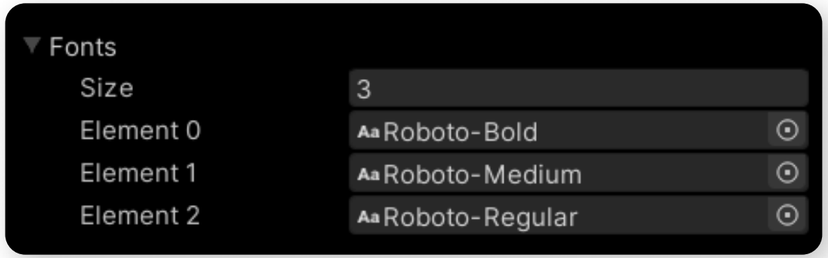



To let an asset know if a specific asset that is a dependency exists in the project, import that asset into the project and switch the parameter "**DISABLED**" corresponding to the name of the asset to "**ENABLED**", and wait for the project to update the resources, this will take about 10 seconds. If there is no dependency, or you have removed it, switch the corresponding dependency parameter to position "**DISABLED**".

1. Switch to "**ENABLED**" if asset "**Json.NET**" is imported. Switch to "**DISABLED**" if the asset is not imported.
2. Switch to "**ENABLED**" if asset "**Tai's True Shadow**" is imported. Switch to "**DISABLED**" if the asset is not imported.
3. Switch to "**ENABLED**" if asset "**TextMeshPro**" is imported. Switch to "**DISABLED**" if the asset is not imported.

# 18. Fonts

1. You can **import** into the project the **fonts** that you use in the **figma** layout, then they will be used when importing. **Otherwise**, the built-in font **Arial** will be used. **Only** works with **standard** text component.

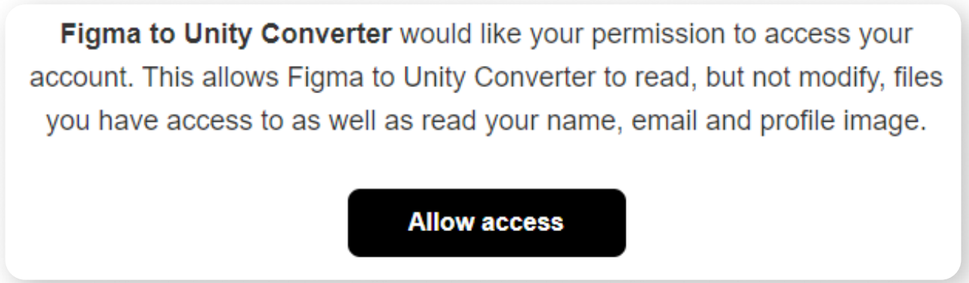


# 19. Auth

1. In order to get the **API-KEY** required to work with the asset, you need to authenticate using the browser. In order to do this, you need to be **logged** into the **figma account** in your default **browser**.

2. Click on the "**Auth With Browser**" button in the asset.


3. In the browser that opens, click on the "**Allow access**" button.

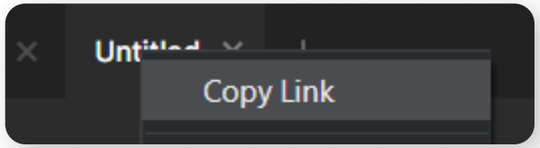



4. After a short time, the key will automatically appear in the "**Api Key**" field of the asset settings.

# 20. Start Importing

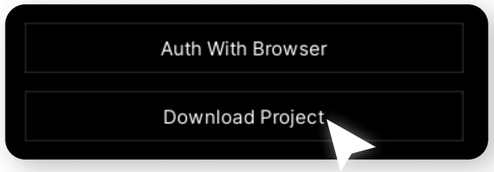
After you have received the api key (section “**Auth**”) and inserted the link to your figma project in the "**Project Link**" field, you can start converting the project from figma to Unity.

2 Open the figma project you are about to import and get a link to it. It can be obtained by **right-clicking on the tab** with an open project.
**Example link**: https://www.figma.com/file/XXXXXXXXXXXXXXXXXXXXXXXXXXXXX...

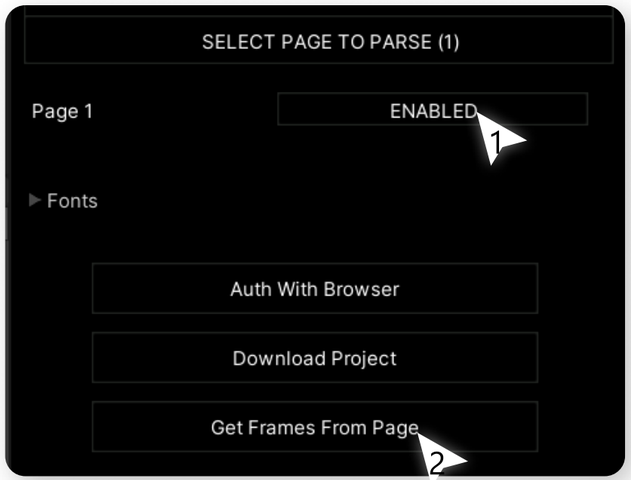




1 After you have logged in and inserted a link to your figma project in the appropriate field, you can start converting the project from figma to unit.

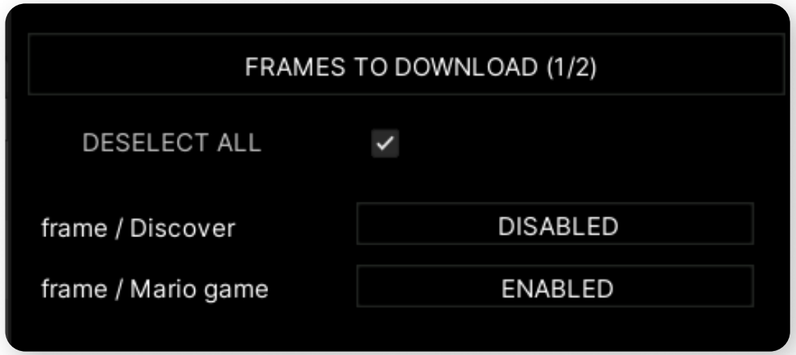



2 You can select **only one page** from which frames will be exported. After you have switched the page you need to "**ENABLED**", press button "**Get Frames From Page**".



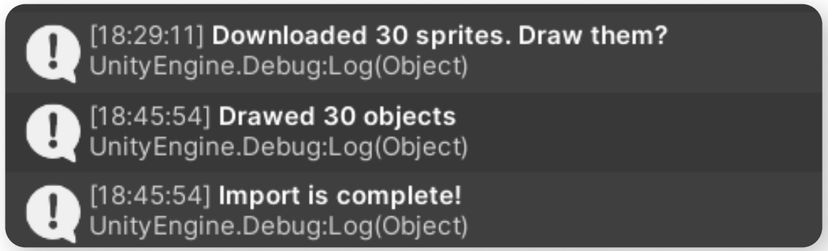
# 20.1. Start Importing

1. Select the frames to import. If there are identical elements in the frames selected for one import, their sprites, in order to avoid duplication, are downloaded to the "**Mutual**" folder. All other sprites will be downloaded to the folder corresponding to the frame name.




2. Once you have selected the frames you want, press the "**Download Frames**" button.

3. After the **message** about the completion of loading the frames and the readiness to render them is displayed in the **console**, press the button "**Draw Frames**".



# 21. Done

1. The import is complete. The frame sprites are located in folder "**/Assets/Sprites/{frameName}**".