SGPatcher

Configuration Guide

Introduction

SGPatcher is a tool designed to streamline the process of managing and deploying assets, scenes, and resources in Unity projects. This guide will walk you through the configuration steps to effectively use SGPatcher in your project.

Step 1: Project Preparation

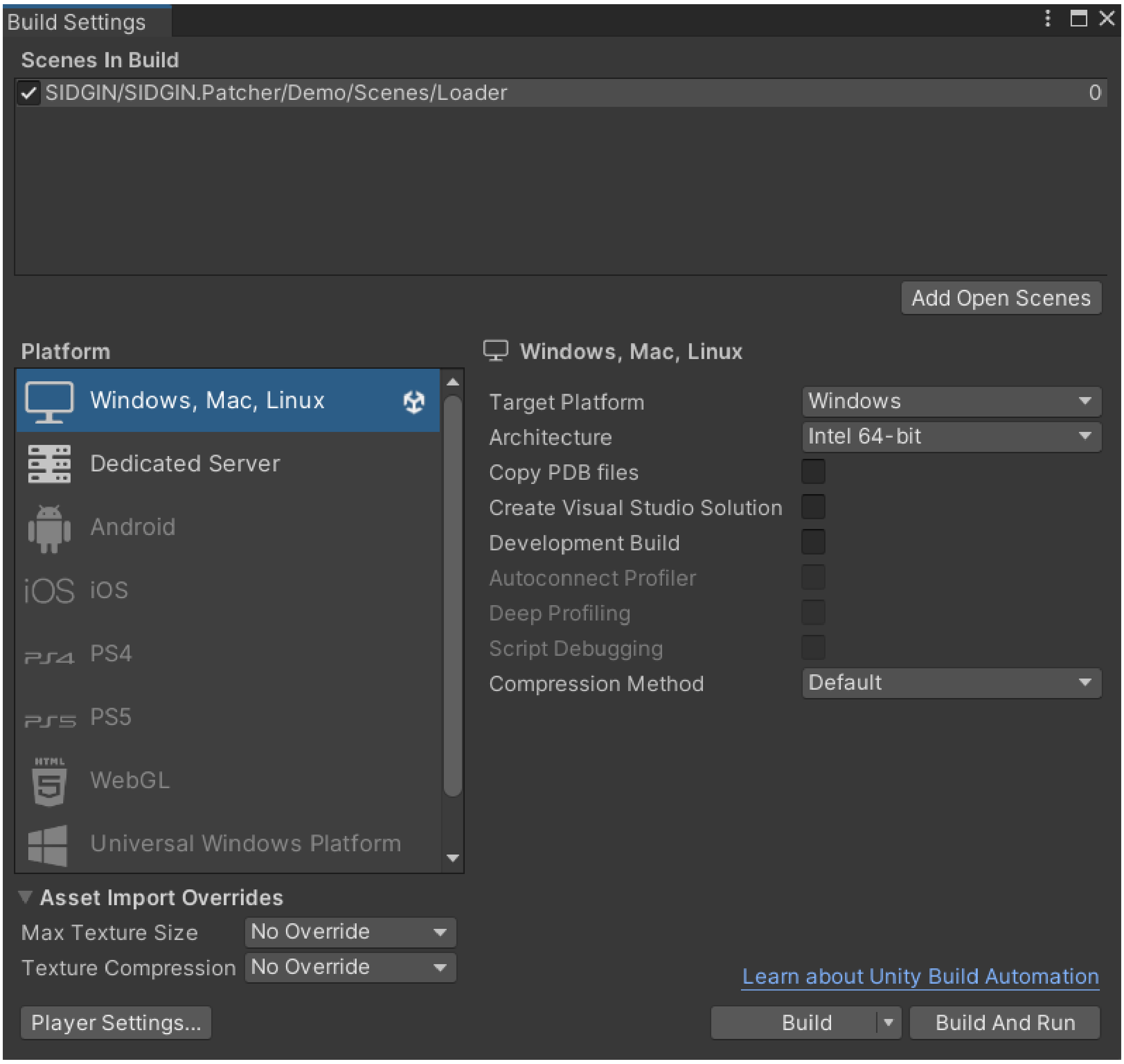
Before integrating SGPatcher into your project, follow these preparatory steps:

1.1 Rename Resource Folders

Rename all project folders named "Resources" that are specifically used by your code. Do not modify folders named "Resources" used by third-party plugins. This step ensures proper integration with SGPatcher.

1.2 Build Settings Configuration

- Open Unity's Build Settings.
- Remove all existing scenes. Take note of their order and paths; this information will be needed later.
- Add the loader scene:



Scenes In Build

☑ SIDGIN/SIDGIN.Patcher/Demo/Scenes/Loader 0

Windows, Mac, Linux

Target Platform	Windows
Architecture	Intel 64-bit
Copy PDB files	□
Create Visual Studio Solution	□
Development Build	□
Autoconnect Profiler	□
Deep Profiling	□
Script Debugging	□
Compression Method	Default

Asset Import Overrides

Max Texture Size	No Override
Texture Compression	No Override

In most cases, this will be the only scene configured in Build Settings.

***Note:** If your project uses plugins that load scenes based on Build Settings, you may need to modify those plugins. Alternatively, leave scenes used by the plugin in the Build Settings. In this case, SGPatcher won't control these scenes.*

1.3 Code Adjustments

- Replace instances of **Resources.Load** and similar methods with **SGResources.Load**. Note that the method signatures for **UnityEngine.Resources** and **SGResources** are identical, so you can make this change seamlessly.
- Replace all occurrences of **SceneManager.Load** with **SGSceneManager.Load**. This adjustment is necessary to load scenes through SGPatcher.

Step 2: SGPatcher Configuration

Now let's configure SGPatcher to suit your project needs. Follow these steps:

2.1 Open SGPatcher Window

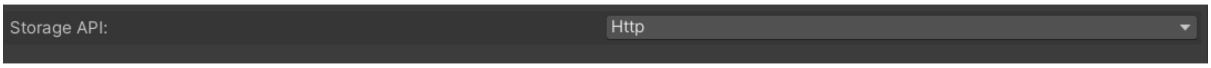
Access the SGPatcher window by navigating to Tools -> SGPatcher.

2.2 Navigate to Settings

In the SGPatcher window, switch to the "Settings" tab.

2.3 Choose API Type

Select the desired API type. For example, choose HTTP.



Storage API: Http

FTP Settings (HTTP mode):

2.3.1 FTP Protocol Selection

- In the "**FTP Protocol**" section, choose the protocol type (FTP or SFTP) for the connection.

2.3.2 FTP Server Configuration

- **FTP Domain**: Enter the FTP server domain.
- **FTP Port**: Specify the FTP server port.
- **FTP User**: Provide the username for connection.
- **FTP Password**: Enter the password for the user.
- **Enable SSL**: Check this option if SSL is required.
- **FTP Home Directory**: Specify the home directory on the FTP server.

2.3.3 Additional Settings for SFTP

- **Open SSH key**: Path to the private key file.
- **Key file passphrase**: Passphrase for the private key file (if applicable).

2.3.4 **Download Base Link**

Download base link: This is the base link for downloading files. It should point to the folder where files are uploaded via FTP ("**FTP Home Directory**").

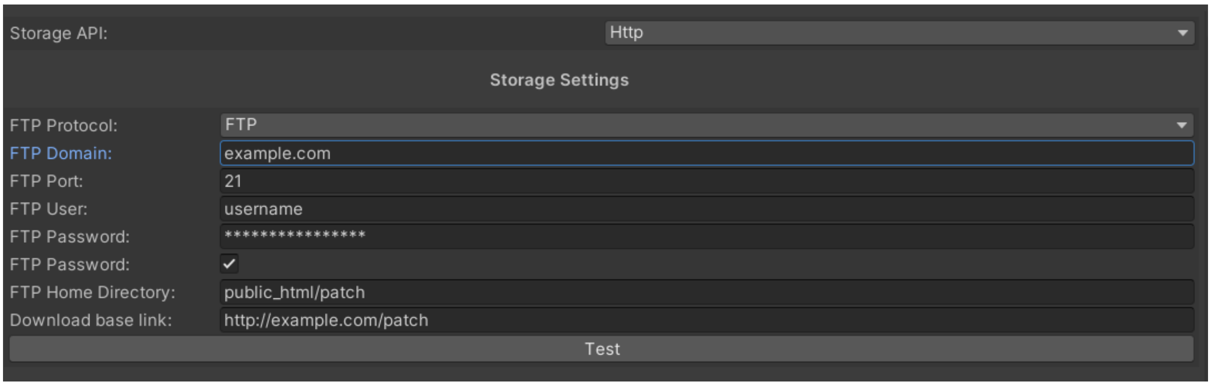
Example:

My FTP host is exampleftp.com.

For instance, when connecting to this host using FileZilla, I see the following directories: Public_html, App, and so on.

Typically, all website files are located in the public_html folder. Therefore, for the correct setup, in the "FTP Home Directory" field, I would specify "public_html/patches" (indicating that the patches folder will be used by SGPatcher).

The "Download base link" would be "http://example.com/patches" (note that the link should have the website's domain, not the FTP connection, if they differ). Also, make sure that files in the folder you specified, in our case, "patches," are accessible directly via an http link.



Storage API:	Http

Storage Settings

FTP Protocol:	FTP
FTP Domain:	example.com
FTP Port:	21
FTP User:	username
FTP Password:	***************
FTP Password:	✓
FTP Home Directory:	public_html/patch
Download base link:	http://example.com/patch

Sometimes, FTP on hosting is configured in a way that upon connection, you directly enter the website's directory. In this case, the "FTP Home Directory" path would be "patches."

I use a simple hosting service at https://www.hetzner.com/webhosting/level-1 (This is not an advertisement 🙂).

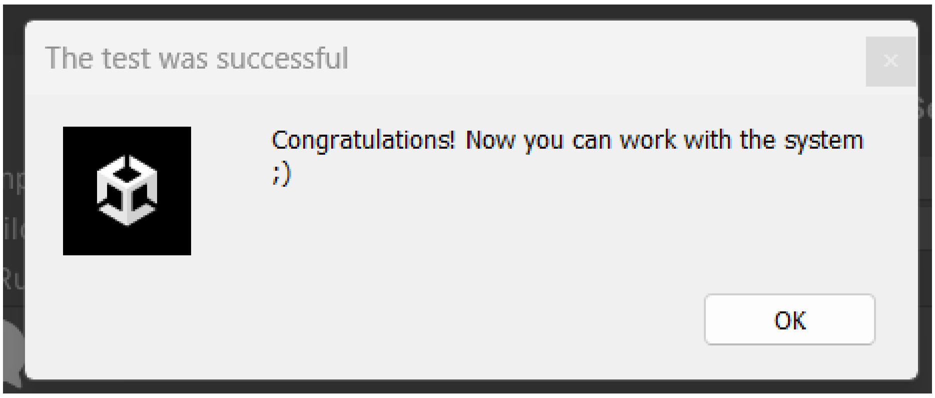
Note: Ensure that the files in the specified folder are accessible directly via HTTP. On common hosting platforms, this is configured in the .htaccess file. Add the following lines to .htaccess:

```
<Files *>
   Header set Content-Disposition 'inline'
   Header set Content-Type 'application/octet-stream'
</Files>
AddType application/octet-stream .json .fullpackage .patchpackage
```

Before proceeding, confirm that your hosting setup allows direct HTTP access to files. If not, consult your hosting support.

2.3.5 Test Connection

- Press "Test" to conduct a test connection and download a file to verify your settings.
- If successful, a dialog will confirm with "The test was successful." For errors, details will be available in the Console window.



The test was successful

Congratulations! Now you can work with the system ;)

Amazon S3 Settings (Amazon mode):

For Amazon S3 integration in SGPatcher:

2.3.6 Amazon S3 Details

- Bucket Name: Specify your Amazon S3 Bucket name.

- Access Key ID: Enter your Access Key ID (you can import it from a CSV file by clicking "Import Credentials File").
- Secret Access Key: Provide your Secret Access Key ID.
- Region Endpoint: Choose the region for Amazon S3.

Optionally, input a Root Folder if you need to specify a root folder in your Amazon S3 bucket.

Press "Test" to check the connection to Amazon S3.

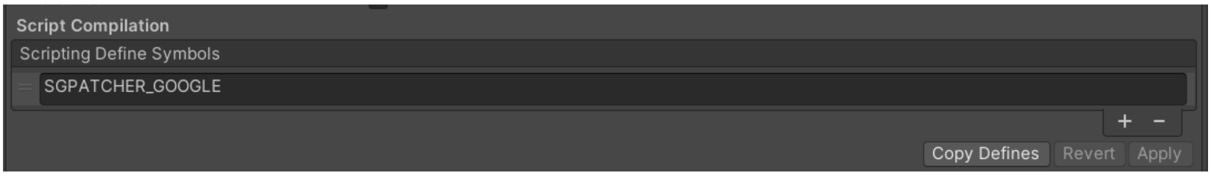
If the settings are correct, a success message will appear. Otherwise, check the "Download base link" and ensure the file named "Test_Connection.json" is accessible via a direct link: "Download base link" + /SGPatcher_test/Test_Connection.json.

If you encounter issues with direct file access, reach out to Amazon support or consult tutorials on configuring Amazon S3 access.

Congratulations! You have successfully configured SGPatcher for your project, whether using FTP or Amazon S3.

Google Drive Settings (Google mode):

For GoogleDrive integration in SGPatcher:

- Import Assets/SIDGIN/GoogleModule.unitypackage
- Add define symbol “SGPATCHER_GOOGLE” into Scripting Define Symbols in PlayerSettings (Edit->ProjectSettings->Player->Other Settings section).
- 

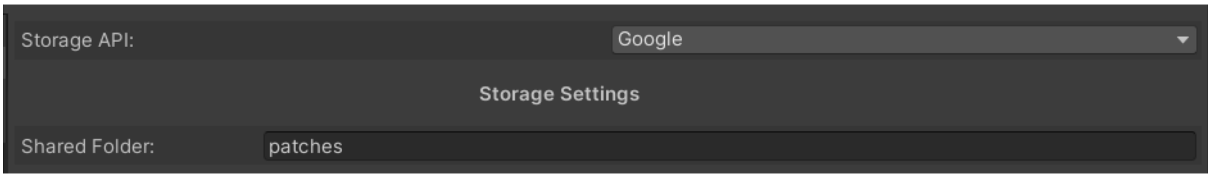
  Script Compilation

  Scripting Define Symbols

  SGPATCHER_GOOGLE

2.3.8 Google Drive Details

- **Shared Folder** is a folder on your Google Drive that will be used by SGPatcher.

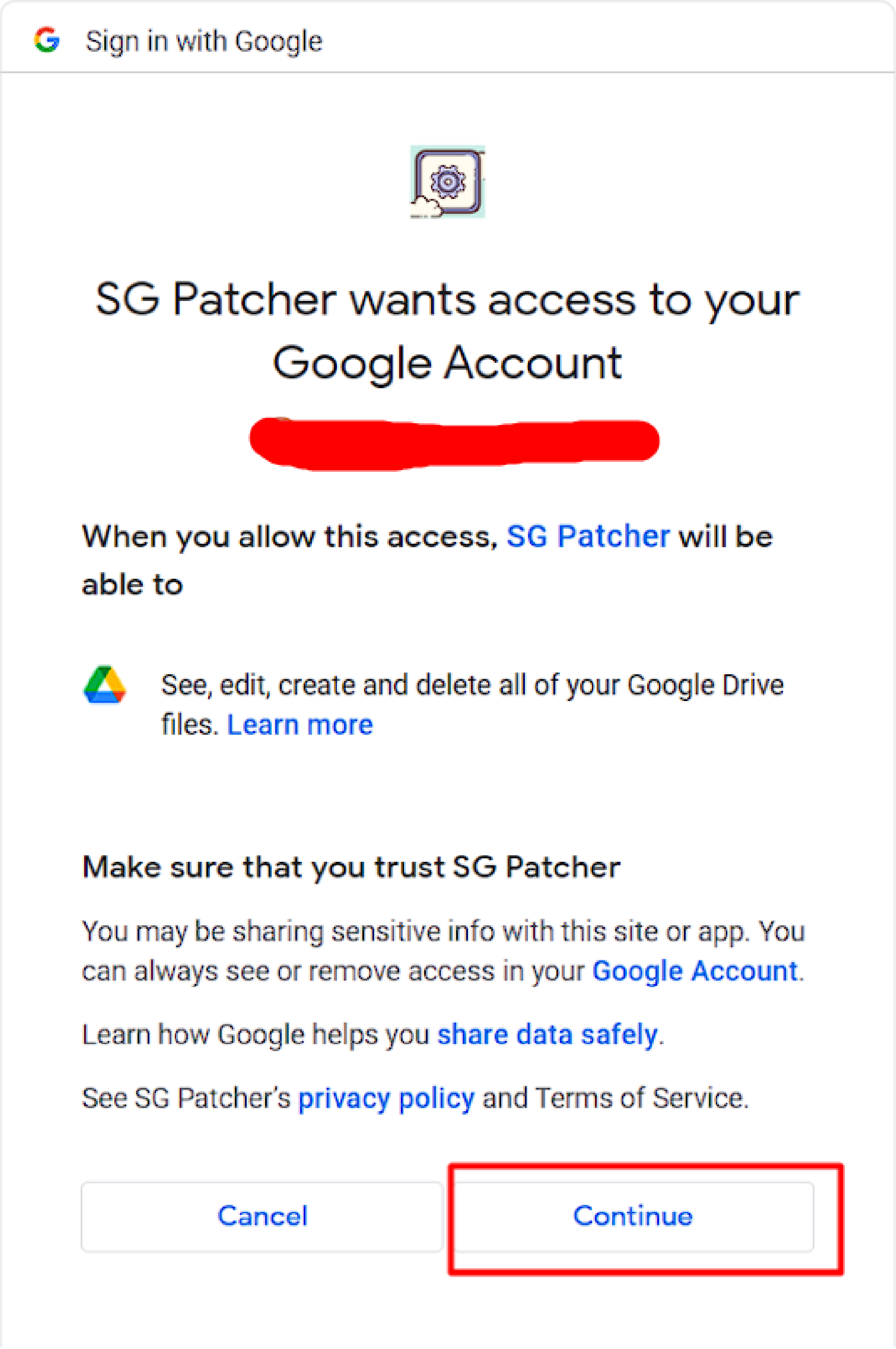
  

  Storage API: Google

  Storage Settings

  Shared Folder: patches

The first time SGPatcher accesses Google Drive, it will request permissions to your Google Drive. A browser page will open where you need to select the Google account you want to grant access to. Then, simply allow SGPatcher to use your drive.



Sign in with Google

SG Patcher wants access to your Google Account

[redacted]

When you allow this access, SG Patcher will be able to

See, edit, create and delete all of your Google Drive files. Learn more

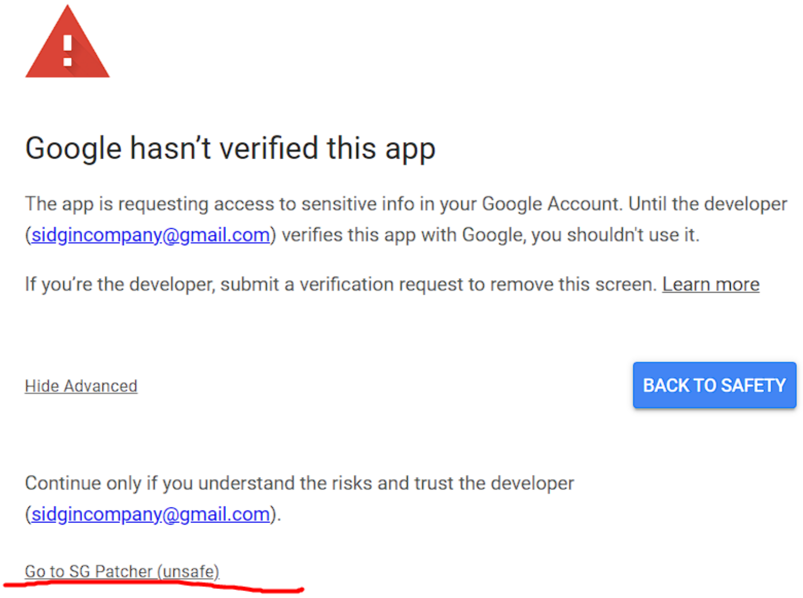
Make sure that you trust SG Patcher

You may be sharing sensitive info with this site or app. You can always see or remove access in your Google Account.

Learn how Google helps you share data safely.

See SG Patcher's privacy policy and Terms of Service.

Sometimes you may see the following screen, simply click on "Go to SGPatcher."



Google hasn’t verified this app

The app is requesting access to sensitive info in your Google Account. Until the developer (sidgincompany@gmail.com) verifies this app with Google, you shouldn't use it.

If you’re the developer, submit a verification request to remove this screen. Learn more

Continue only if you understand the risks and trust the developer (sidgincompany@gmail.com).

To unlink SGPatcher from your current account, delete the "sgpatcher_token" folder located at the same level as the "Assets" folder.

Step 3: Configuring Definitions

What are Definitions?

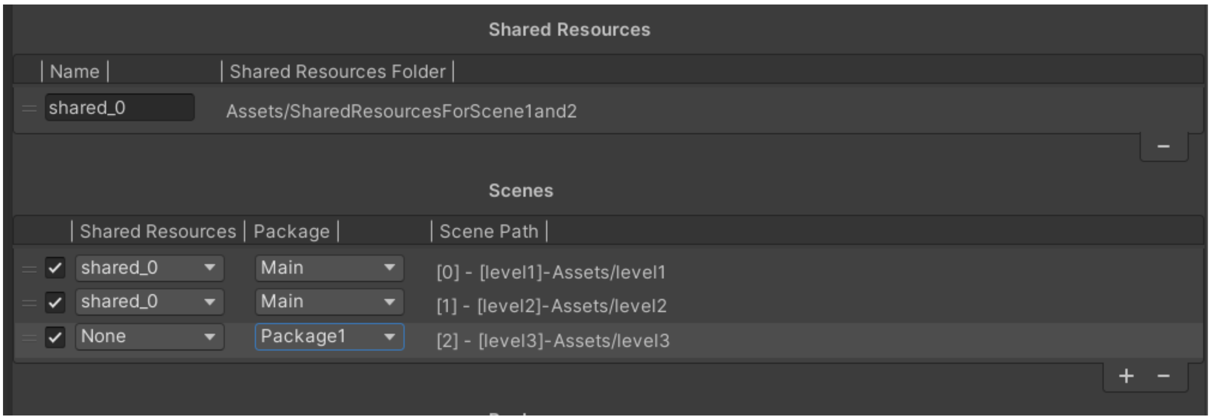
In the Definitions section, SGPatcher provides the ability to configure various build configurations for different platforms, such as iOS and Android. Definitions define the rules by which resources will be assembled in each specific configuration.

Keep in mind that you must create a separate Definition for each platform.

Key Configuration Points for Definitions:

1. Scene Configuration

Scenes: Scenes are references to game scenes in the project. When configuring Definitions, you specify the package in which each scene is located. Additionally, for each scene, you can specify which Shared Resources will be used.



Shared Resources

Name	Shared Resources Folder
shared_0	Assets/SharedResourcesForScene1and2

Scenes

	Shared Resources	Package	Scene Path
☑	shared_0	Main	[0] - [level1]-Assets/level1
☑	shared_0	Main	[1] - [level2]-Assets/level2
☑	None	Package1	[2] - [level3]-Assets/level3

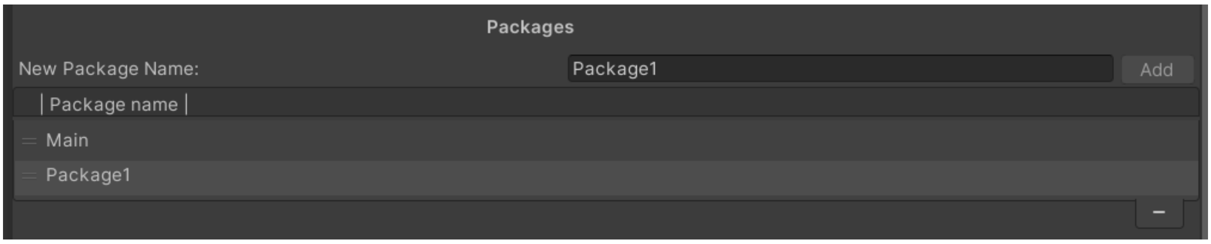
2. Shared Resources

Shared Resources: Shared Resources are common assets that can be used by multiple scenes. You define a list of folders where these shared resources will reside. This is crucial for optimizing disk space usage.

Shared Bundle Scenes: When specifying shared resources for a scene, a special "shared bundle" is created. This bundle contains common resources used by multiple scenes, reducing the build size. Shared Resources are loaded before the scene and unloaded when all scenes using them are unloaded.

3. Packages

Here, you can add packages.



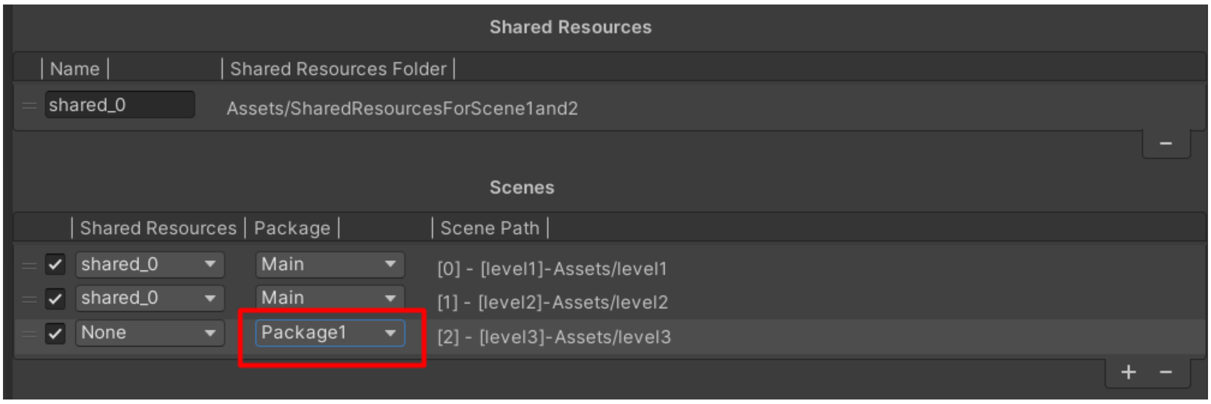
Packages

New Package Name: Package1

Package name
Main
Package1

Main Package: This package is loaded along with the build. It contains essential resources required to run the application.
Other packages you add can be loaded on demand, optimizing resource usage and speeding up application loading.

To specify which package a scene will use, indicate the package in the scene list window.



Shared Resources

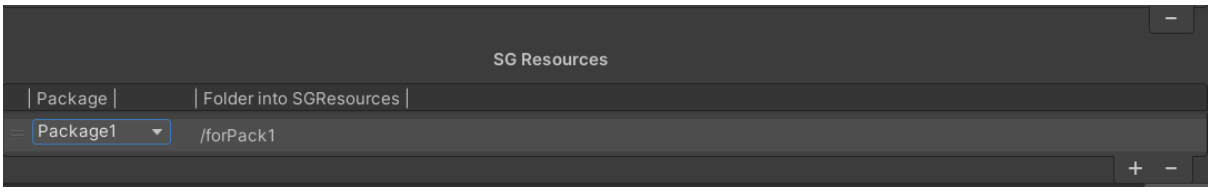
Name	Shared Resources Folder
shared_0	Assets/SharedResourcesForScene1and2

Scenes

	Shared Resources	Package	Scene Path
☑	shared_0	Main	[0] - [level1]-Assets/level1
☑	shared_0	Main	[1] - [level2]-Assets/level2
☑	None	Package1	[2] - [level3]-Assets/level3

4. SGResources

SGResources: All folders in the project named SGResources, by default, include all resources in these folders in the Main Package. However, in the SGResources section, you can specify which specific folders within SGResources will not be included in the main build. You also indicate which package these folders should belong to.



SG Resources

Package	Folder into SGResources
Package1	/forPack1

In this example, the folder "forPack1" is inside the SGResources folder and will not be added to the Main Package but will be included in Package1.

Step 4: Publishing a Version

Publishing involves building AssetBundles and automatically sending all packages to the server.

- Navigate to the **Versions** window.

- Select the **Definition** in the top-right corner. This indicates that SGpatcher will build packages for the chosen platform.

- Check "Full version" if you are creating the initial version or want to delete all patches and create a new full package.

  When you choose "**Full version**" SGpatcher creates packages as they are and uploads them to the server.

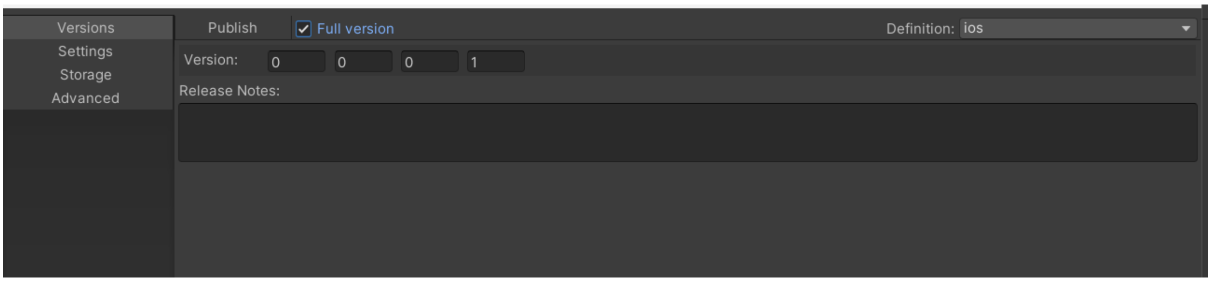
  When you don't select this flag SGpatcher creates a package and then computes the changed part sending a patch to the server.

  Usually, the patch is much smaller in size than the full version.

  Patches created by SGPatcher are always applied sequentially, with hash checksum verification, providing a reliable method for updating resources and saving on internet traffic.

  If any issues arise with resources or versions during your game updates, a reliable way to fix this is to build a **full version**. Additionally, if your clients don't have potential issues with internet speed, you can always build a **full version**. It might be slower to download from the server, but it reduces the update application time and project build time.

To publish a version, click the "**Publish**" button.



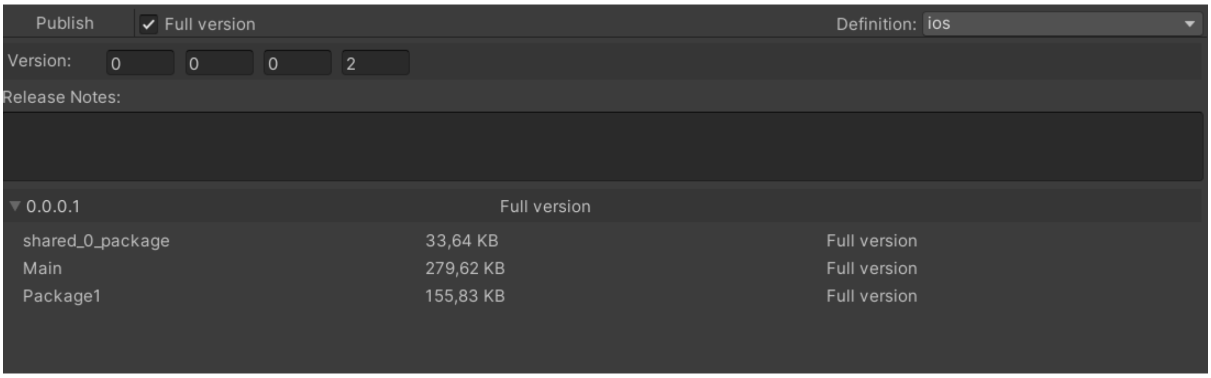
☑ Full version

Definition: ios

Version: 0 0 0 1

Release Notes:

After the build process, you will receive information about which packages and resources were uploaded to the server, the package type, and its size.

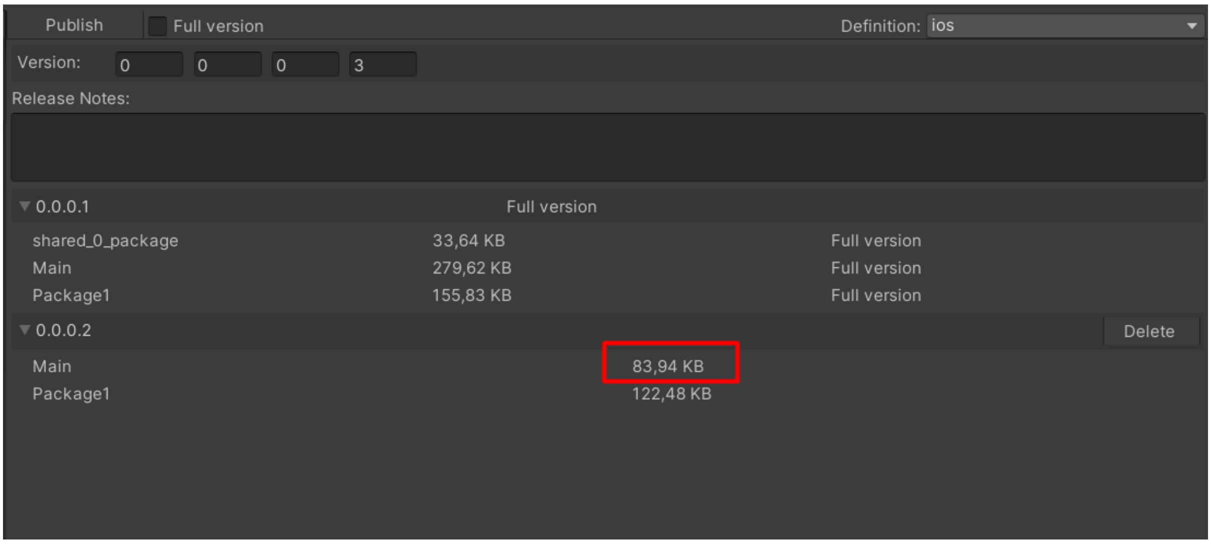


Version: 0 0 0 2

Release Notes:

▼ 0.0.0.1	Full version	
shared_0_package	33,64 KB	Full version
Main	279,62 KB	Full version
Package1	155,83 KB	Full version

To create a patch, simply click the "**Publish**" button again. You can see that SGPatcher will create a binary diff patch, which will be smaller in size than the original version.



Version: 0 0 0 3

Release Notes:

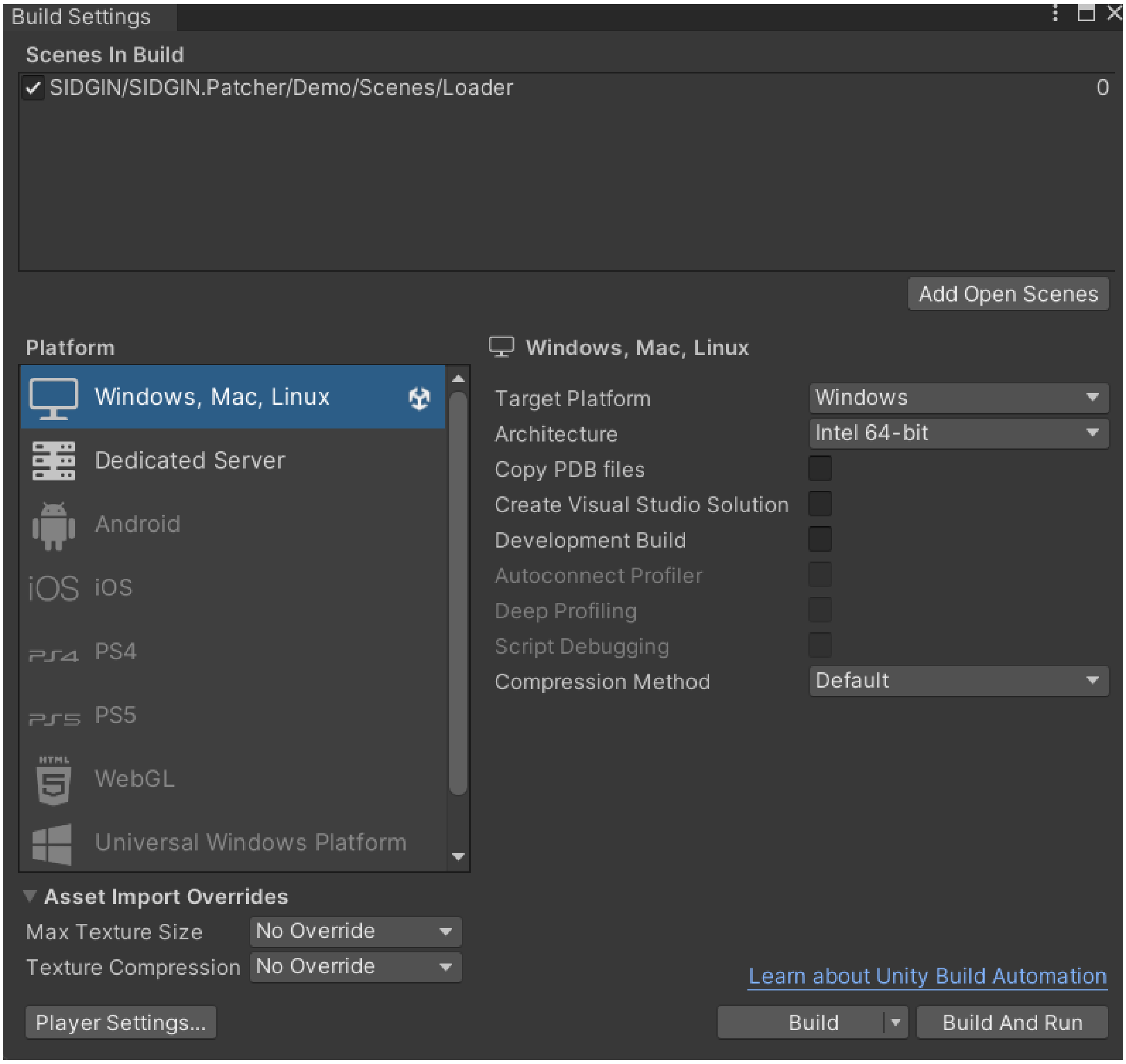
▼ 0.0.0.1	Full version	
shared_0_package	33,64 KB	Full version
Main	279,62 KB	Full version
Package1	155,83 KB	Full version
▼ 0.0.0.2		
Main	83,94 KB	
Package1	122,48 KB	

Tip: If you find yourself essentially replacing the current content with another, for example, removing 200MB of resources and adding 300MB, and the original application is approximately 300MB, it is optimal to create a Full version.

Step 5: Building the Client

The client is a Unity build that you will distribute (App Store, Google Play, Steam...).

Ensure that the loader scene is the first in Build Settings:



Build Settings

Scenes In Build

✓ SIDGIN/SIDGIN.Patcher/Demo/Scenes/Loader 0

Platform: Windows, Mac, Linux

Target Platform: Windows

Architecture: Intel 64-bit

Copy PDB files: □

Create Visual Studio Solution: □

Development Build: □

Autoconnect Profiler: □

Deep Profiling: □

Script Debugging: □

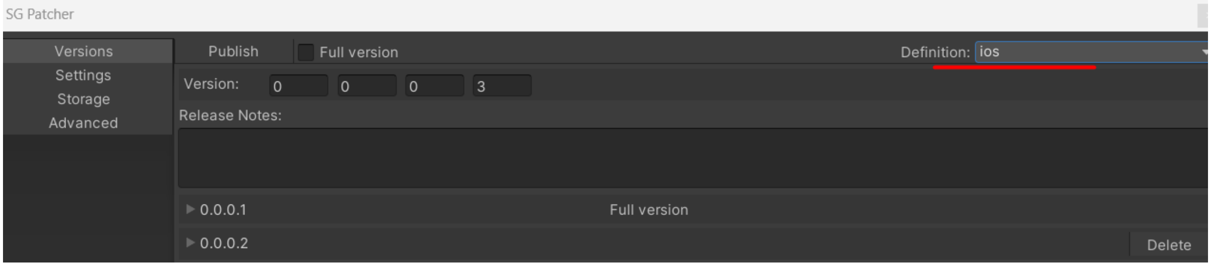
Compression Method: Default

Asset Import Overrides

Max Texture Size: No Override

Texture Compression: No Override

If you have more than one Definition, set the appropriate Definition before building:



SG Patcher

Publish □ Full version Definition: ios

Version: 0 0 0 3

Release Notes:

0.0.0.1 Full version

0.0.0.2

Keep in mind that you need to create a separate Definition for each platform.

Ready! Now you can check the functionality of SGPatcher.

We're here to help you every step of the way! If you have questions, feedback, or need assistance, feel free to reach out.

Support Contacts:

- Email: support@sidgin.com
- Discord: https://discord.com/invite/KcyQ3gc

If you find SGPatcher beneficial, kindly consider leaving a positive review on the Asset Store. Your feedback motivates us to enhance our product and assists others in making informed decisions.

Thank you for being part of the SGPatcher community and contributing to its success!