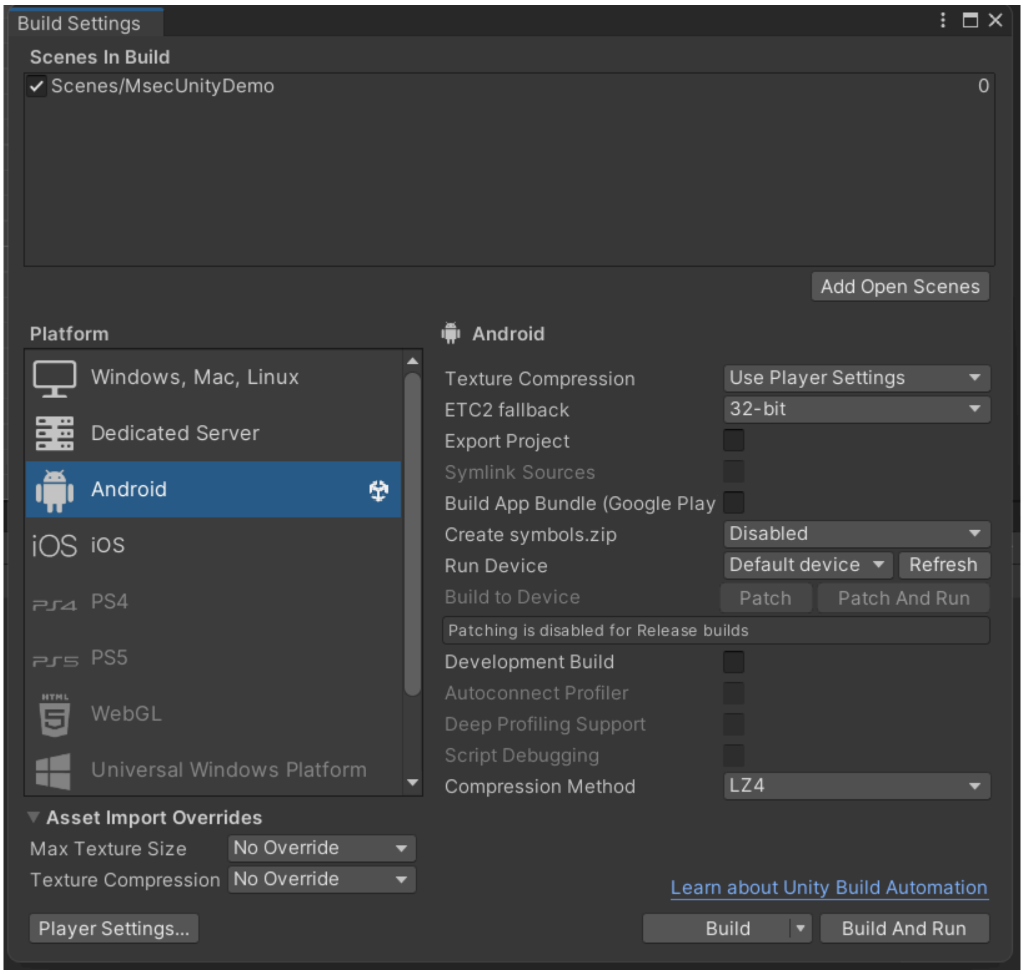
GUIDE FOR RUNNING & BUILDING MSEC UNITY DEMO

1. Prepare the Unity Project:

- To download the project, use the command:
  git clone https://github.com/msecsys-dev/msecunitydemo.git
- Ensure that your project is thoroughly tested and ready to run.



Build Settings

Scenes In Build

✓ Scenes/MsecUnityDemo 0

Add Open Scenes

Platform

Windows, Mac, Linux
Dedicated Server
Android
iOS
PS4
PS5
WebGL
Universal Windows Platform

Android

Texture Compression	Use Player Settings
ETC2 fallback	32-bit
Export Project	☐
Symlink Sources	☐
Build App Bundle (Google Play	☐
Create symbols.zip	Disabled
Run Device	Default device / Refresh
Build to Device	Patch / Patch And Run
Development Build	☐
Autoconnect Profiler	☐
Deep Profiling Support	☐
Script Debugging	☐
Compression Method	LZ4

Patching is disabled for Release builds

Asset Import Overrides

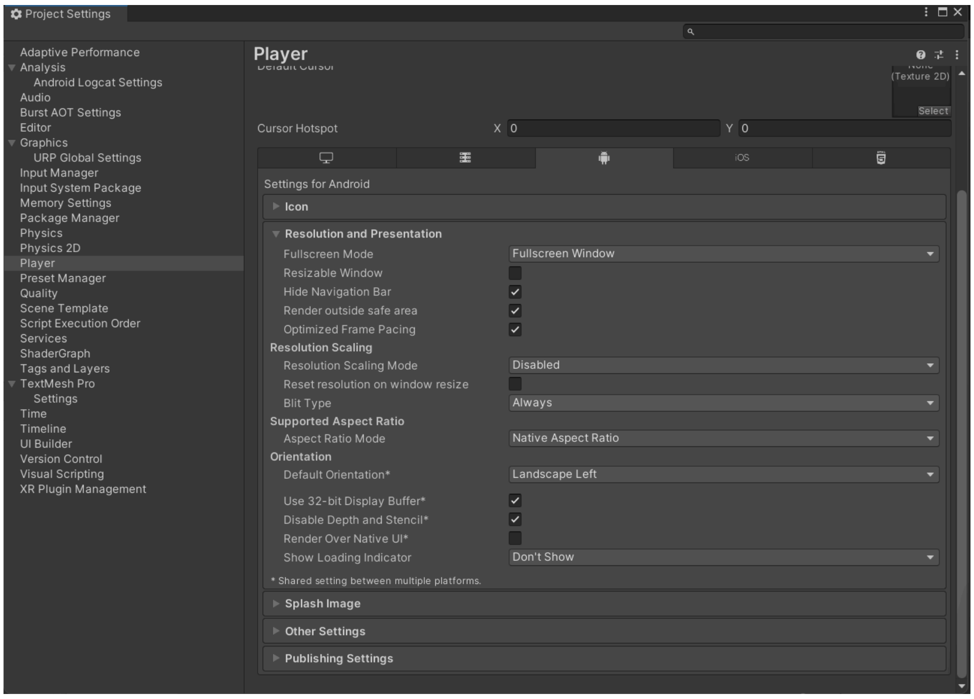
Max Texture Size	No Override
Texture Compression	No Override

Learn about Unity Build Automation

Player Settings... Build Build And Run

2. For Android

- Set Up and Configure Android Build:
  - In Unity, go to File > Build Settings.
  - Select Android and click Switch Platform if it hasn't been done yet.
  - Under Scenes In Build, add the following scene:
    `Scenes/MsecUnityDemo`
- Adjust the following settings according to your requirements in Player Settings and Resolution and Presentation:


Project Settings

Adaptive Performance
Analysis
Android Logcat Settings
Audio
Burst AOT Settings
Editor
Graphics
URP Global Settings
Input Manager
Input System Package
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
ShaderGraph
Tags and Layers
TextMesh Pro
Settings
Time
Timeline
UI Builder
Version Control
Visual Scripting
XR Plugin Management

Player

Cursor Hotspot X 0 Y 0

Settings for Android

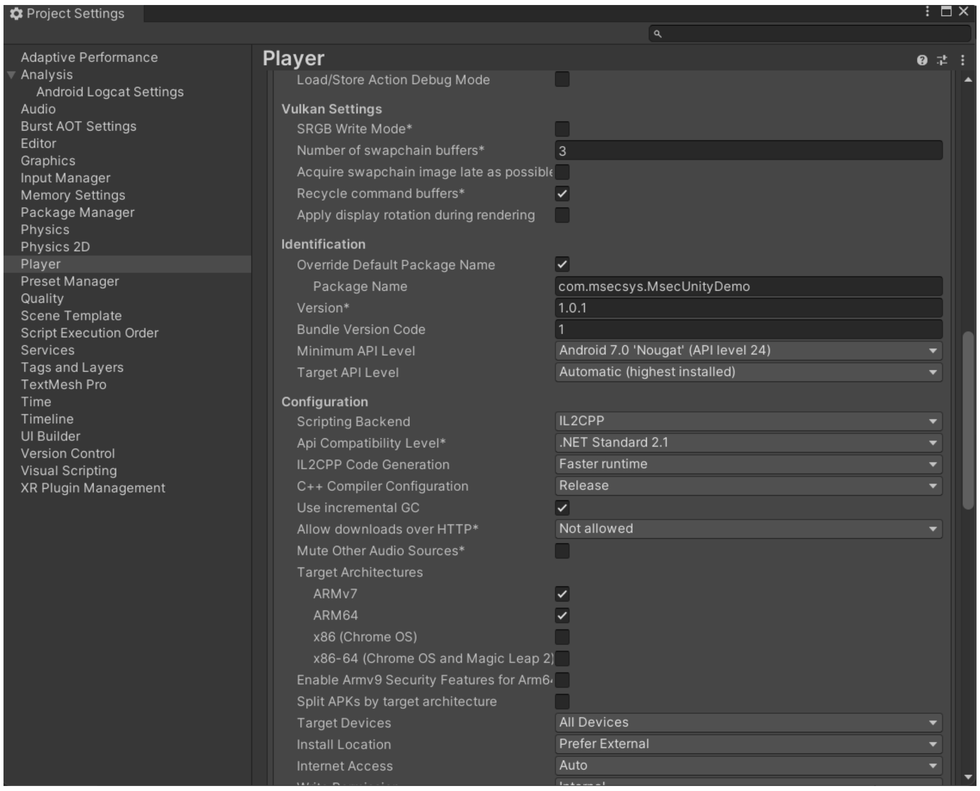
Icon

Resolution and Presentation

Fullscreen Mode	Fullscreen Window
Resizable Window	
Hide Navigation Bar	✓
Render outside safe area	✓
Optimized Frame Pacing	✓
Resolution Scaling	
Resolution Scaling Mode	Disabled
Reset resolution on window resize	
Blit Type	Always
Supported Aspect Ratio	
Aspect Ratio Mode	Native Aspect Ratio
Orientation	
Default Orientation*	Landscape Left
Use 32-bit Display Buffer*	✓
Disable Depth and Stencil*	✓
Render Over Native UI*	
Show Loading Indicator	Don't Show

* Shared setting between multiple platforms.

Splash Image
Other Settings
Publishing Settings


Project Settings

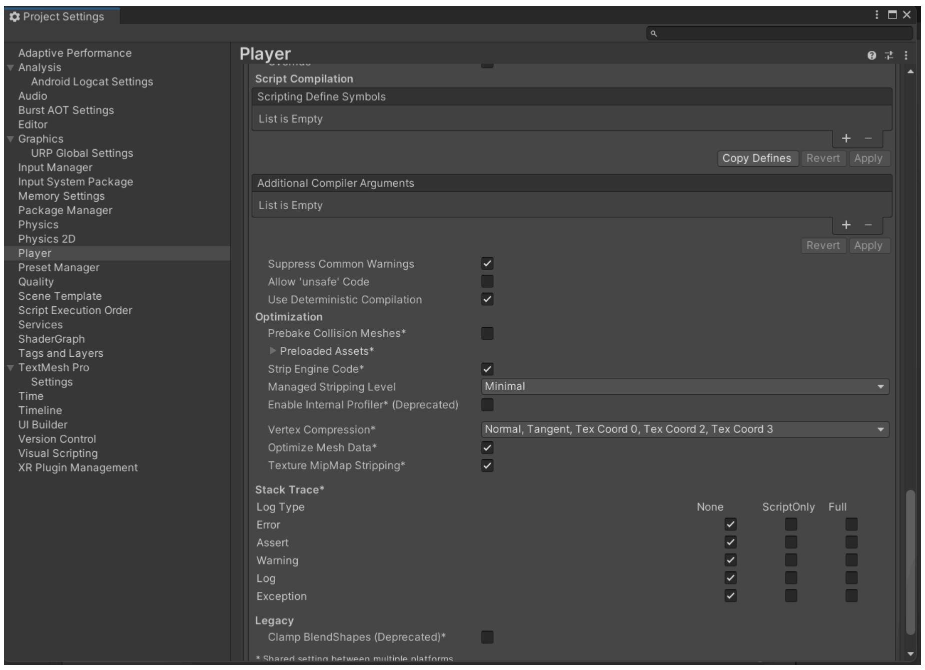
Adaptive Performance
Analysis
Android Logcat Settings
Audio
Burst AOT Settings
Editor
Graphics
Input Manager
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Tags and Layers
TextMesh Pro
Time
Timeline
UI Builder
Version Control
Visual Scripting
XR Plugin Management

Player

Load/Store Action Debug Mode	
Vulkan Settings	
SRGB Write Mode*	
Number of swapchain buffers*	3
Acquire swapchain image late as possible	
Recycle command buffers*	✓
Apply display rotation during rendering	
Identification	
Override Default Package Name	✓
Package Name	com.msecsys.MsecUnityDemo
Version*	1.0.1
Bundle Version Code	1
Minimum API Level	Android 7.0 'Nougat' (API level 24)
Target API Level	Automatic (highest installed)
Configuration	
Scripting Backend	IL2CPP
Api Compatibility Level*	.NET Standard 2.1
IL2CPP Code Generation	Faster runtime
C++ Compiler Configuration	Release
Use incremental GC	✓
Allow downloads over HTTP*	Not allowed
Mute Other Audio Sources*	
Target Architectures	
ARMv7	✓
ARM64	✓
x86 (Chrome OS)	
x86-64 (Chrome OS and Magic Leap 2)	
Enable Armv9 Security Features for Arm6	
Split APKs by target architecture	
Target Devices	All Devices
Install Location	Prefer External
Internet Access	Auto

- Resolution and Presentation:
- Identification:
  - Package Name: Set the package name for your app in the format, e.g., com.msecsys.MsecUnityDemo.
  - Version: Set the app version.
  - Bundle Version Code: This is the internal version number of the app. It must be incremented each time the app is updated.

- Minimum API Level: Select the minimum API level that your app supports (usually Android 7.0 'Nougat' API level 24 or higher).
- Target API Level: Typically, this is set to Automatic (highest installed) to ensure support for the latest Android version.
- Configuration:
- Scripting Backend: Choose the scripting backend, usually IL2CPP for best performance.
- API Compatibility Level: Generally set to .NET Standard 2.1.


Project Settings

Adaptive Performance
Analysis
Android Logcat Settings
Audio
Burst AOT Settings
Editor
Graphics
URP Global Settings
Input Manager
Input System Package
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
ShaderGraph
Tags and Layers
TextMesh Pro
Settings
Time
Timeline
UI Builder
Version Control
Visual Scripting
XR Plugin Management

Player

Script Compilation
Scripting Define Symbols
List is Empty
Copy Defines Revert Apply
Additional Compiler Arguments
List is Empty
Revert Apply

Suppress Common Warnings	✓
Allow 'unsafe' Code	
Use Deterministic Compilation	✓
Optimization	
Prebake Collision Meshes*	
Preloaded Assets*	
Strip Engine Code*	✓
Managed Stripping Level	Minimal
Enable Internal Profiler* (Deprecated)	
Vertex Compression*	Normal, Tangent, Tex Coord 0, Tex Coord 2, Tex Coord 3
Optimize Mesh Data*	✓
Texture MipMap Stripping*	✓

Stack Trace*

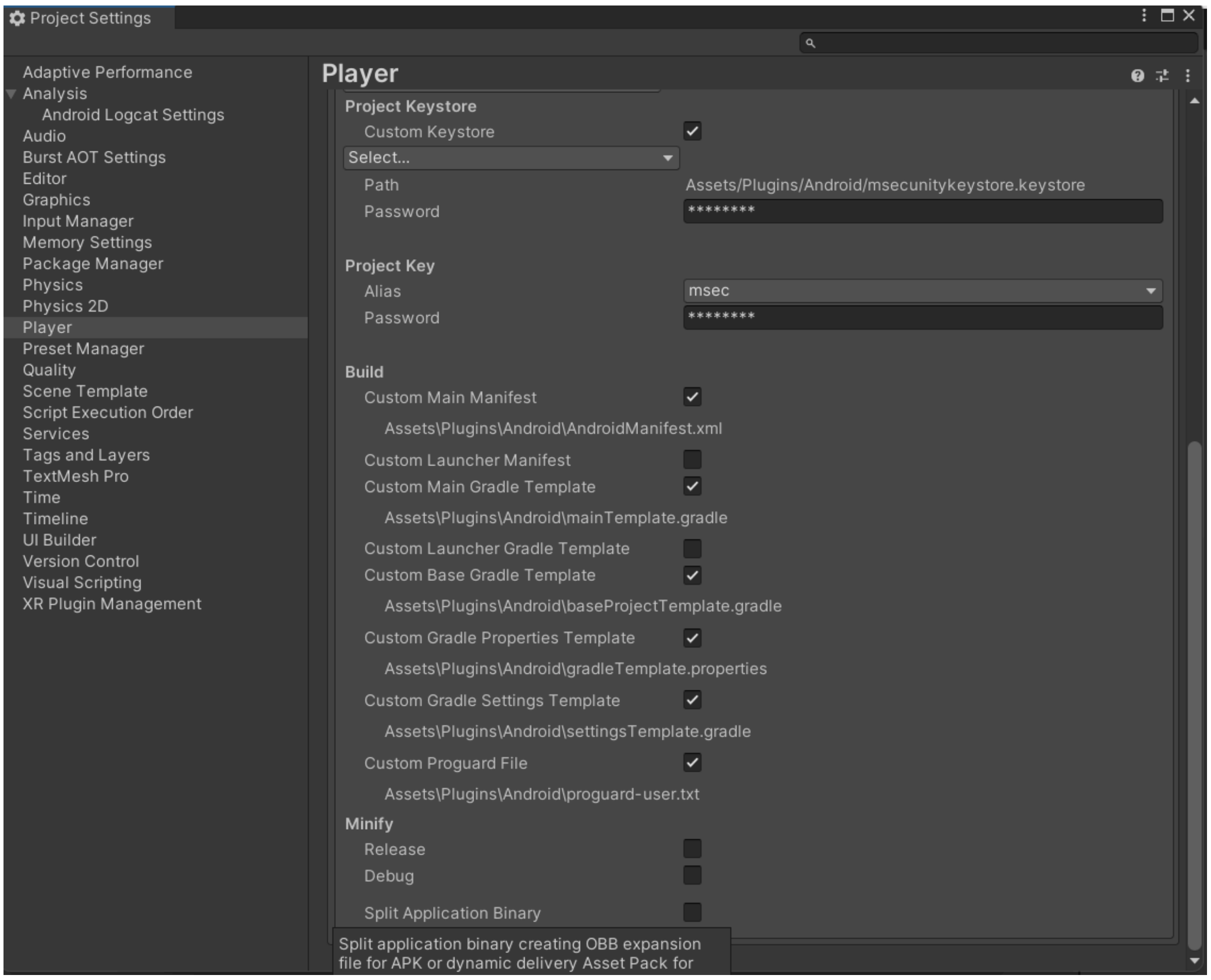
Log Type	None	ScriptOnly	Full
Error	✓		
Assert	✓		
Warning	✓		
Log	✓		
Exception	✓		

Legacy
Clamp BlendShapes (Deprecated)*

* Shared setting between multiple platforms.

- Optimization:
  - Preload Shaders: If you have essential shaders, you can preload them to enhance performance.
  - Strip Engine Code: Remove unnecessary code to reduce app size.
  - Managed Stripping Level: Choose the stripping level, typically Medium or High.
- Create or Import a Keystore:
- Before publishing, you need a keystore to sign your app.
  - Go to Player Settings > Publishing Settings.
  - Create a new keystore or import an existing one.
  - Sign your app with this keystore.
  - Project Keystore:
    - Select Custom Keystore
    - Path: Assets/Plugins/Android/msecunitykeystore.keystore
    - Password: msec2024
  - Project Key:
    - Alias: Select msec
    - Password: msec2024
  - In the Build settings, check the following:
    - Custom Main Manifest
    - Custom Main Gradle Template

- Custom Base Gradle Template
- Custom Gradle Properties Template
- Custom Gradle Settings Template
- Custom Proguard File



Player

Project Keystore

Custom Keystore ✓

Select...

Path Assets/Plugins/Android/msecunitykeystore.keystore

Password ********

Project Key

Alias msec

Password ********

Build

Custom Main Manifest ✓

Assets\Plugins\Android\AndroidManifest.xml

Custom Launcher Manifest

Custom Main Gradle Template ✓

Assets\Plugins\Android\mainTemplate.gradle

Custom Launcher Gradle Template

Custom Base Gradle Template ✓

Assets\Plugins\Android\baseProjectTemplate.gradle

Custom Gradle Properties Template ✓

Assets\Plugins\Android\gradleTemplate.properties

Custom Gradle Settings Template ✓

Assets\Plugins\Android\settingsTemplate.gradle

Custom Proguard File ✓

Assets\Plugins\Android\proguard-user.txt

Minify

Release

Debug

Split Application Binary

Split application binary creating OBB expansion file for APK or dynamic delivery Asset Pack for

- Return to Build Settings and proceed with the build process.

3. For iOS

- Prepare the Unity Project for iOS:
    - Open Unity and navigate to File > Build Settings.
    - Select iOS as the target platform and click Switch Platform if not already selected.
    - Add necessary scenes under Scenes In Build (e.g., Scenes/MsecUnityDemo).
- Configure iOS Project Settings in Unity:
    - Open Player Settings and go to iOS Settings.
    - Set up the following configurations:
        - Bundle Identifier: Set the unique bundle identifier, e.g., com.mbf.MsecUnityDemo.
        - Version: Define the app version (e.g., 1.0).
        - Build Number: Increment the build number for each new release.
    - Configuration:
        - Scripting Backend: Use IL2CPP for better performance.
        - API Compatibility Level: Set to .NET Standard 2.1.
        - Architecture: Choose ARM64 for 64-bit iOS devices.
    - Optimization:
        - Strip Engine Code: Enable to reduce app size.
        - Managed Stripping Level: Set to Medium or High for further optimization.
- Other Settings:
    - Set the Target minimum iOS version to at least iOS 13.0.
    - Set the Graphics API to Metal for optimal iOS performance.
- Export the Project to Xcode:
    - In Unity, go to File > Build Settings.
    - Click Build and select a folder to save the Xcode project.
    - Unity will export your project to this folder. Open the generated .xcodeproj file in Xcode.
- Set Up CocoaPods:
    - Open Terminal and navigate to the directory where your Xcode project was exported by Unity.
    - Run the following command to initialize CocoaPods: pod init
    - in Terminal, run the following command to install the specified dependencies: pod install

CocoaPods will create a Unity-iPhone.xcworkspace file. Always open the project using Unity-iPhone.xcworkspace instead of Unity-iPhone.xcodeproj to ensure that CocoaPods dependencies are properly included.

- Configure the Xcode Project:
    - Open Unity-iPhone.xcworkspace in Xcode.
    - Ensure that all dependencies are properly linked by checking Build Phases > Link Binary With Libraries.
    - Go to Signing & Capabilities and:
        - Select your Apple Development Team.

        - Ensure the correct provisioning profile is selected for both Debug and Release builds.
- Enable any necessary capabilities (e.g., Push Notifications, In-App Purchases) based on the app's requirements.
- Build and Run on Device:
    - Connect your iOS device to your computer.
    - In Xcode, select your device as the target.
    - Clean the build folder by clicking Product > Clean Build Folder.
    - Build the project by selecting Product > Build.
    - Once built, select Product > Run to install and run the app on your connected device.