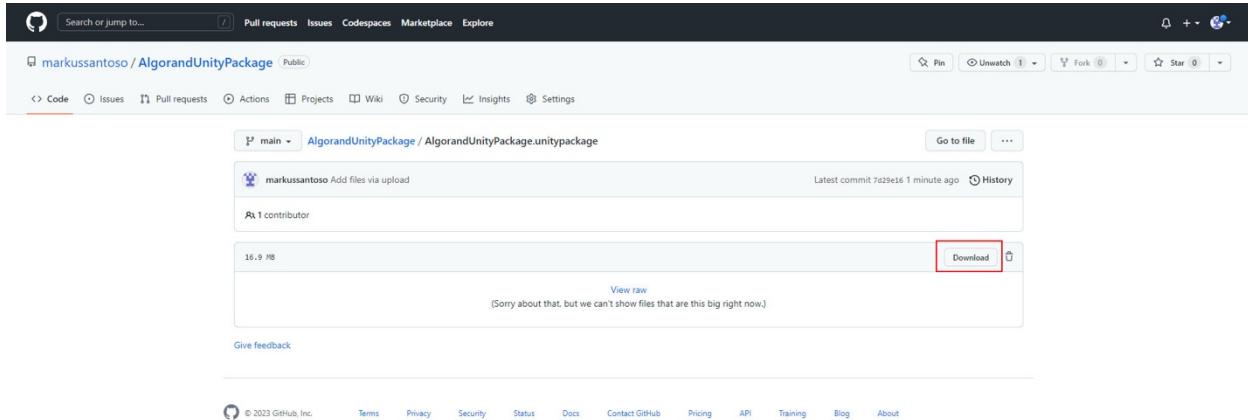
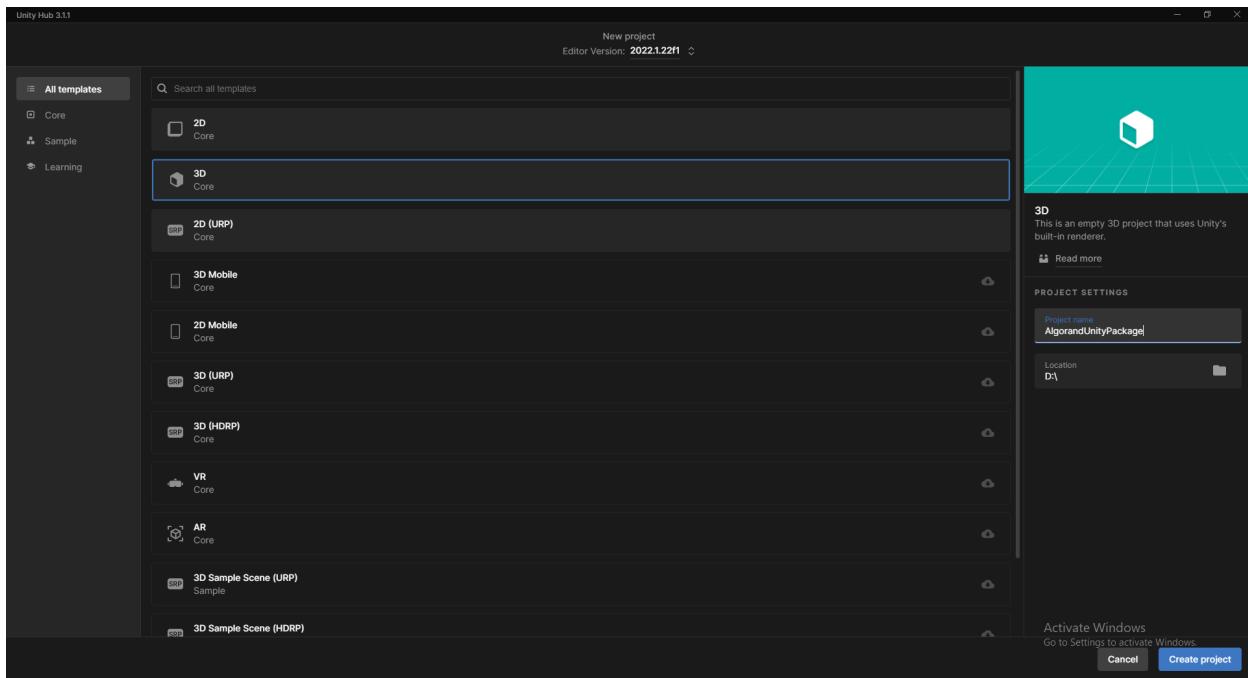


## 1. Go to:

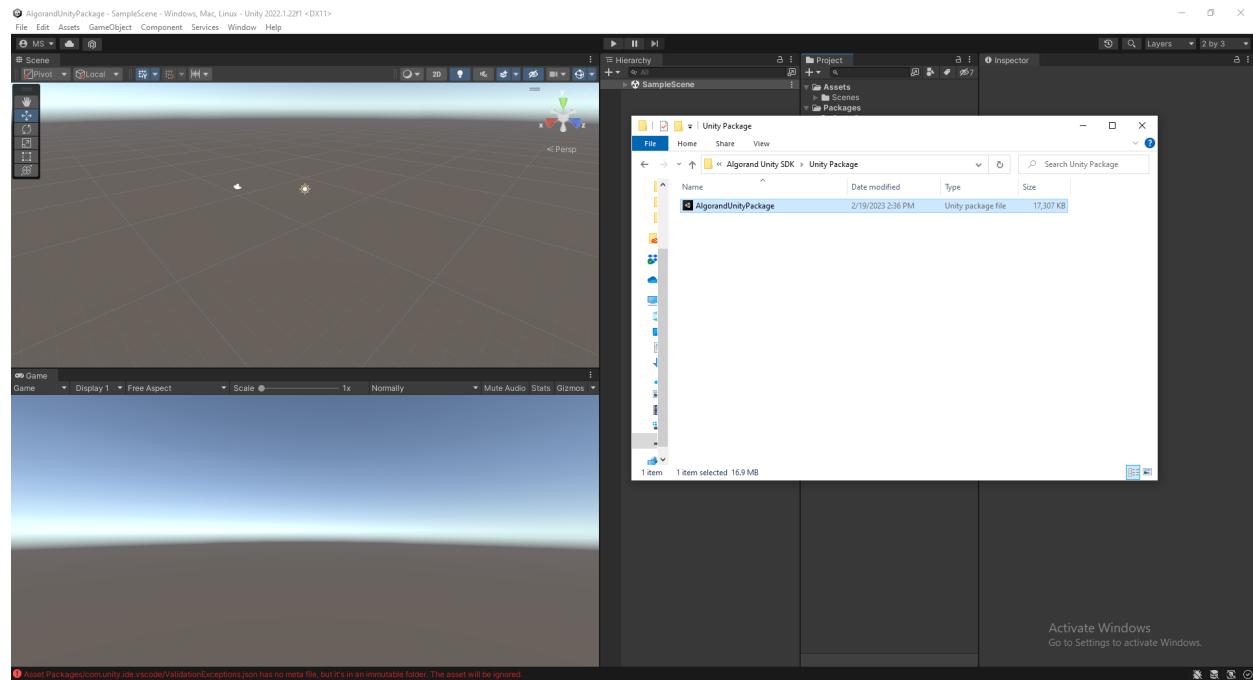
<https://github.com/markussantoso/AlgorandUnityPackage/blob/main/AlgorandUnityPackage.unitypackage> and download the Algorand Unity package:



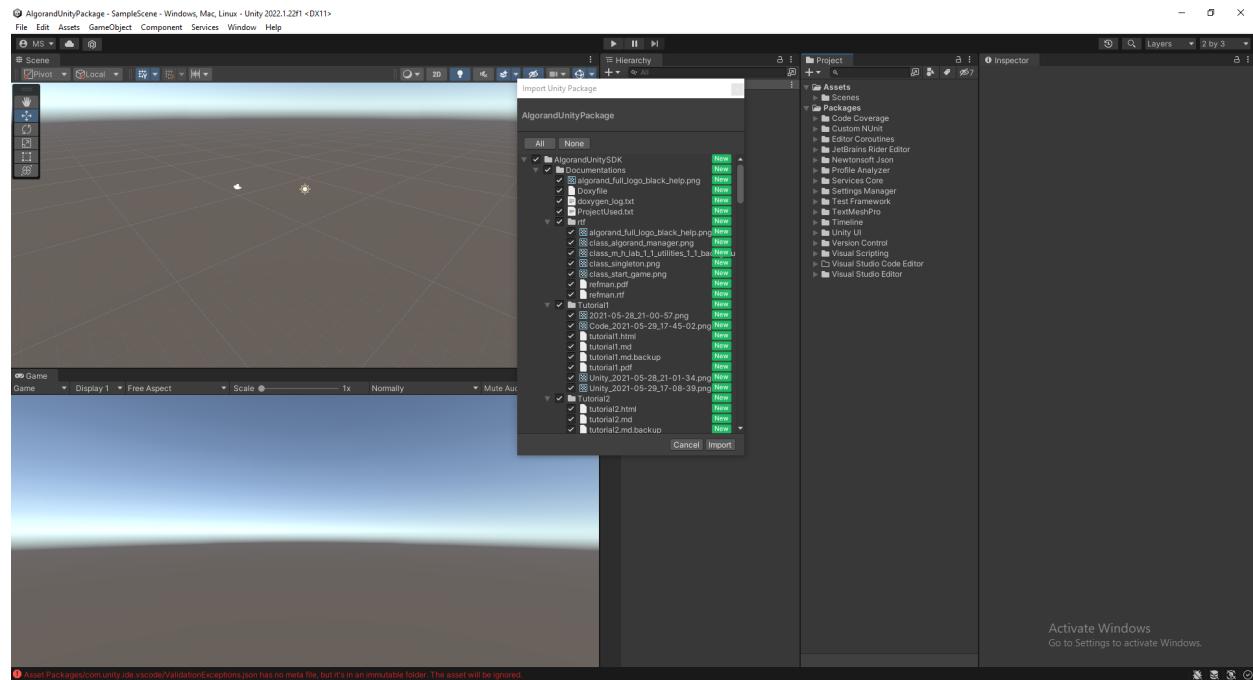
## 2. Create New Unity project



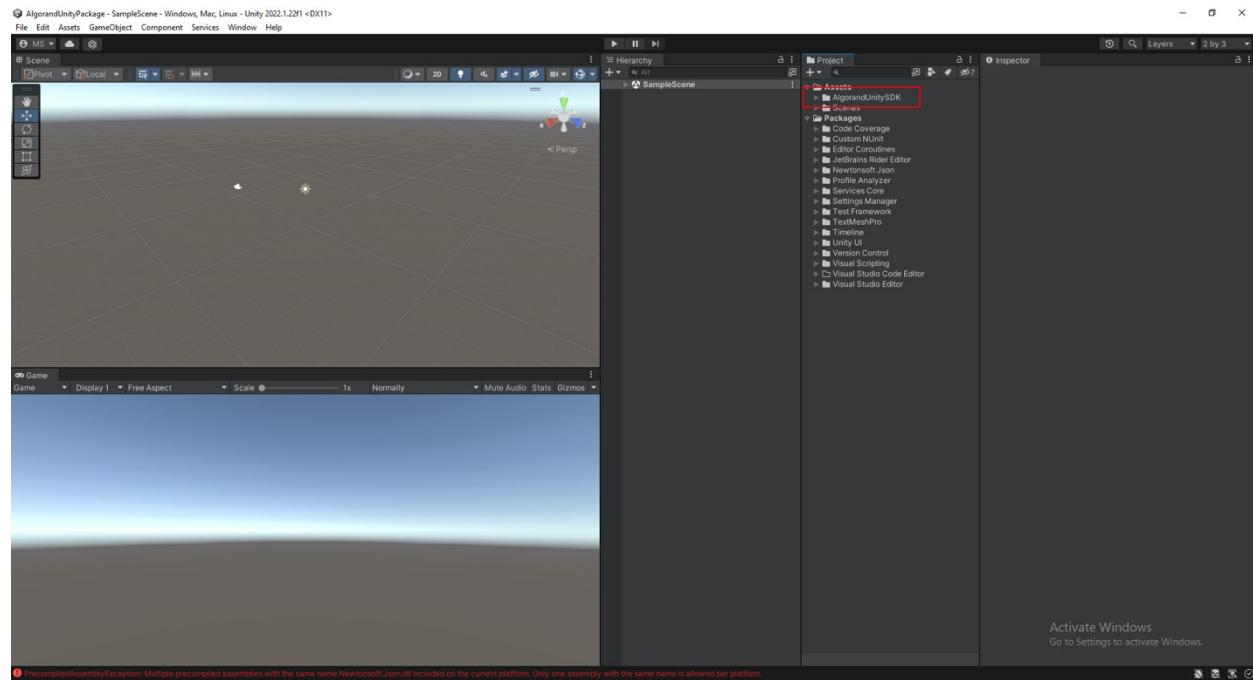
### 3. Drag n Drop AlgorandUnity package to the project



Click 'Import'

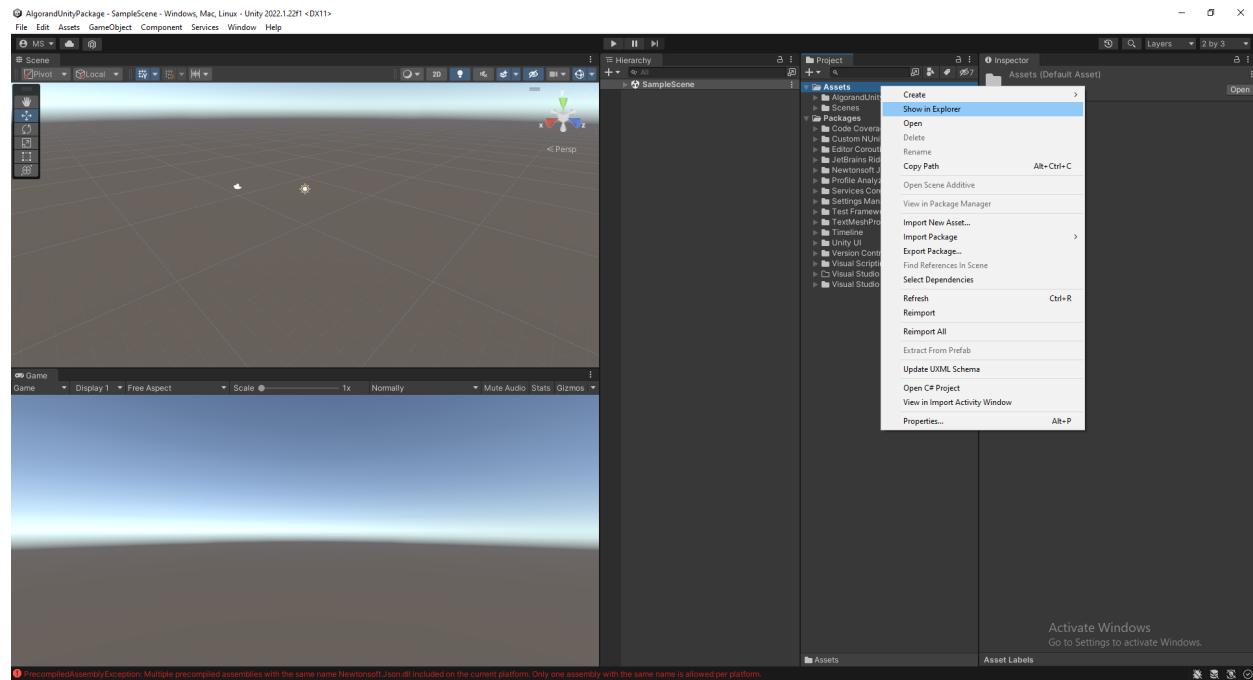


## AlgorandUnitySDK imported !!!

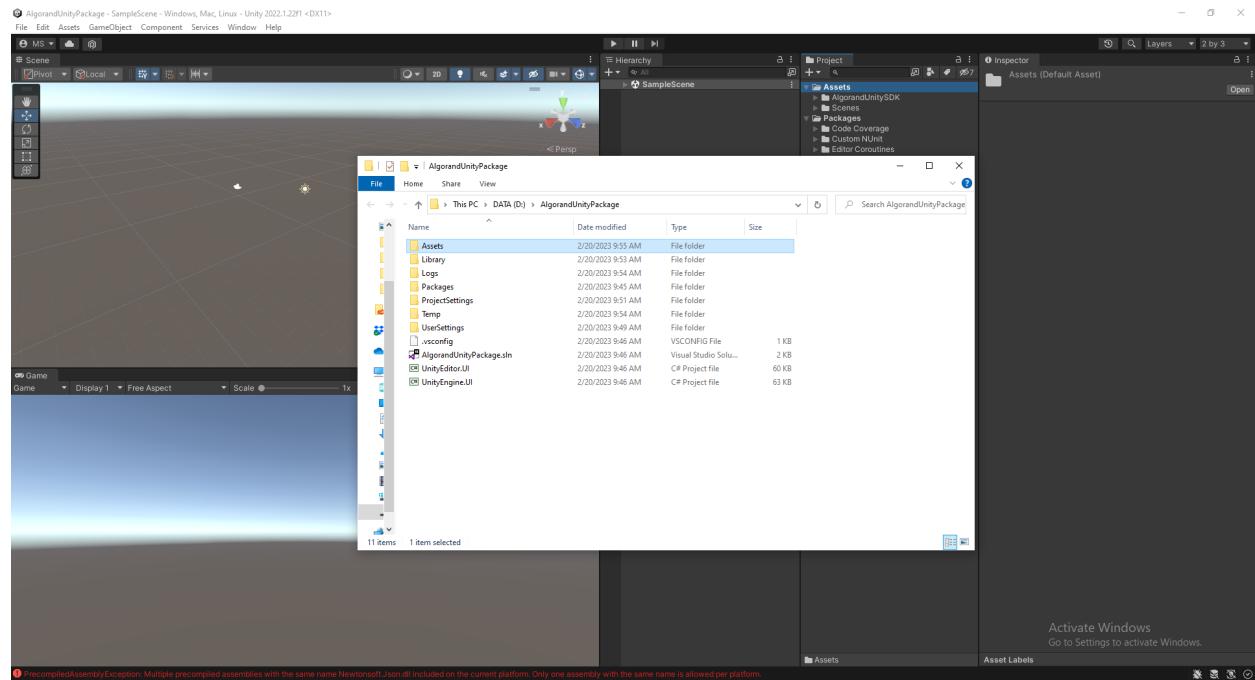


In this phase, you will see couple errors related with 'newtonsoft' and others.

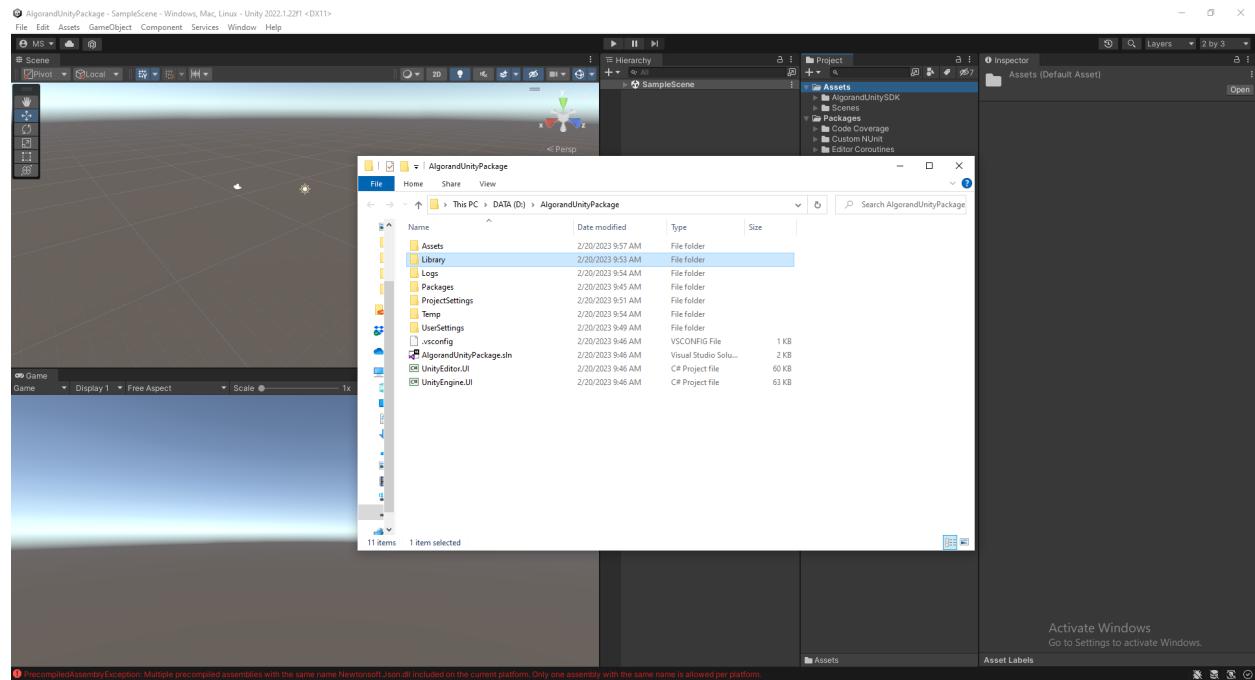
4. To solve these errors, right-click Assets then select 'Show in Explorer' (WIN) or 'Reveal in Finder' (MAC):



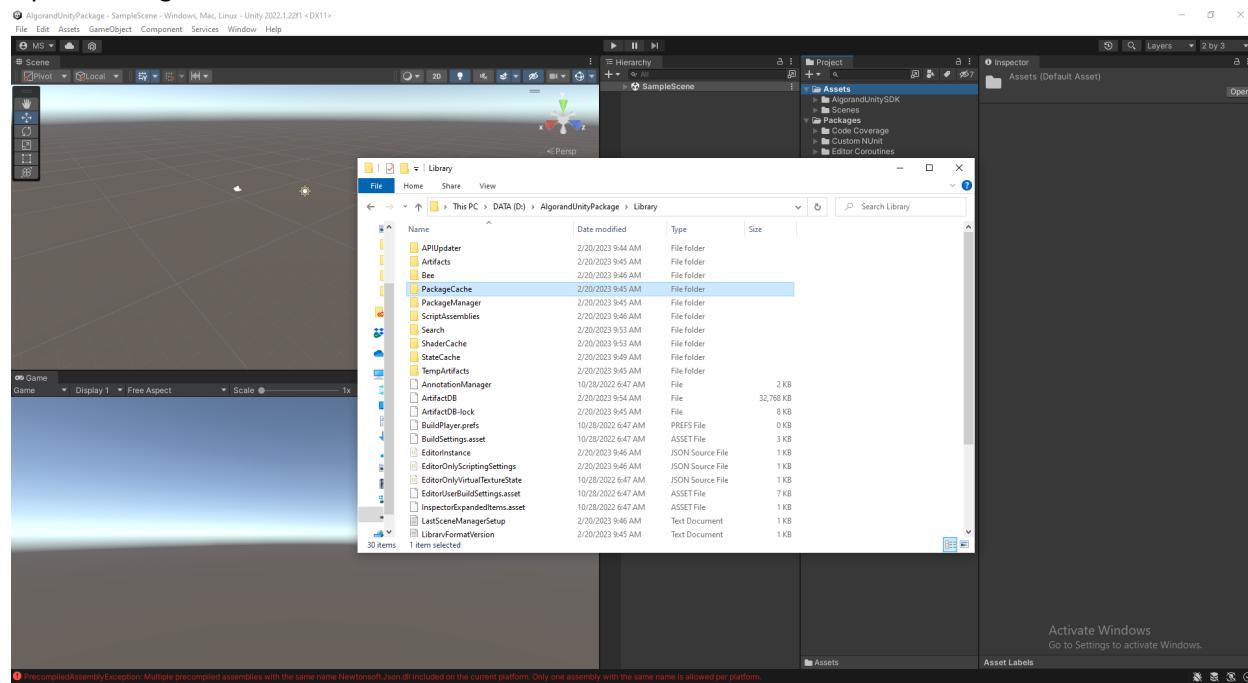
And this window will open:



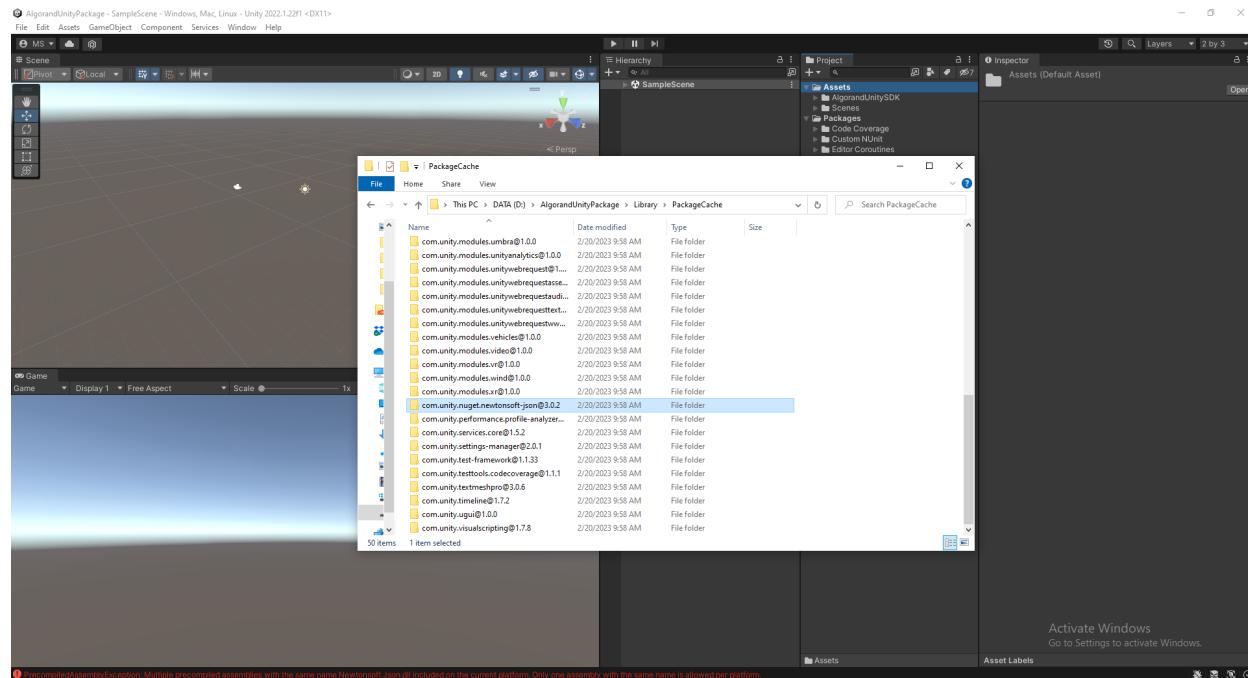
Now go to 'Library'



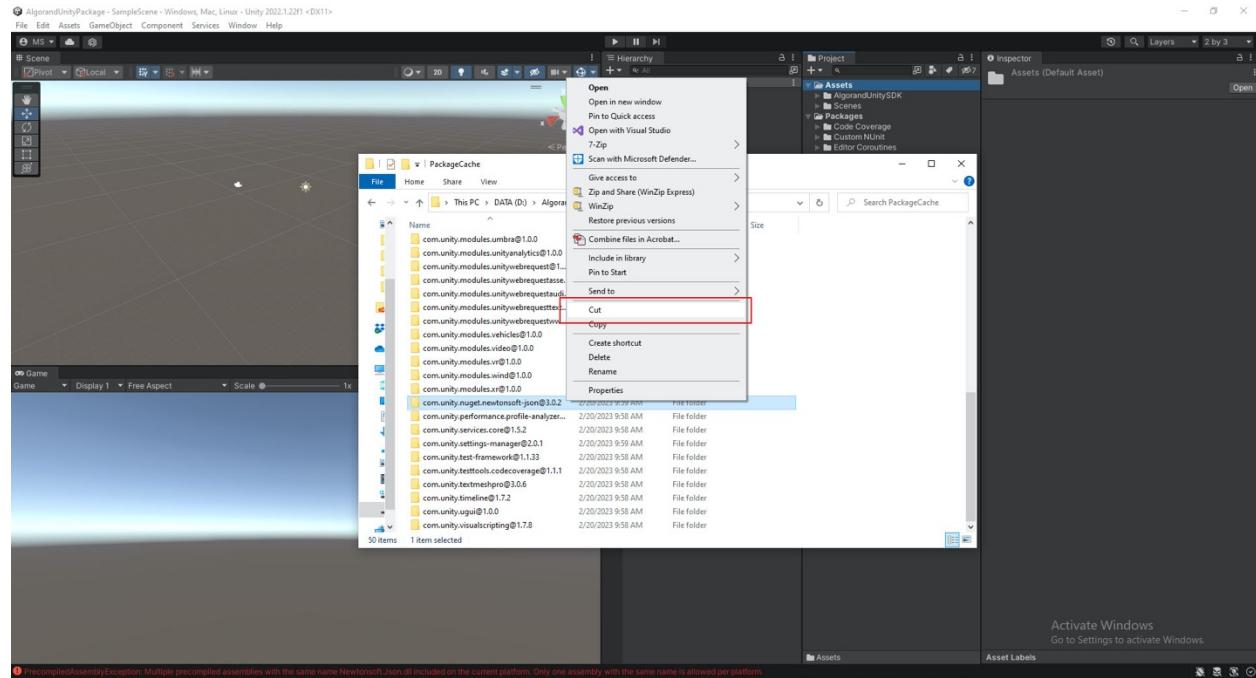
## Open 'Package Cache'



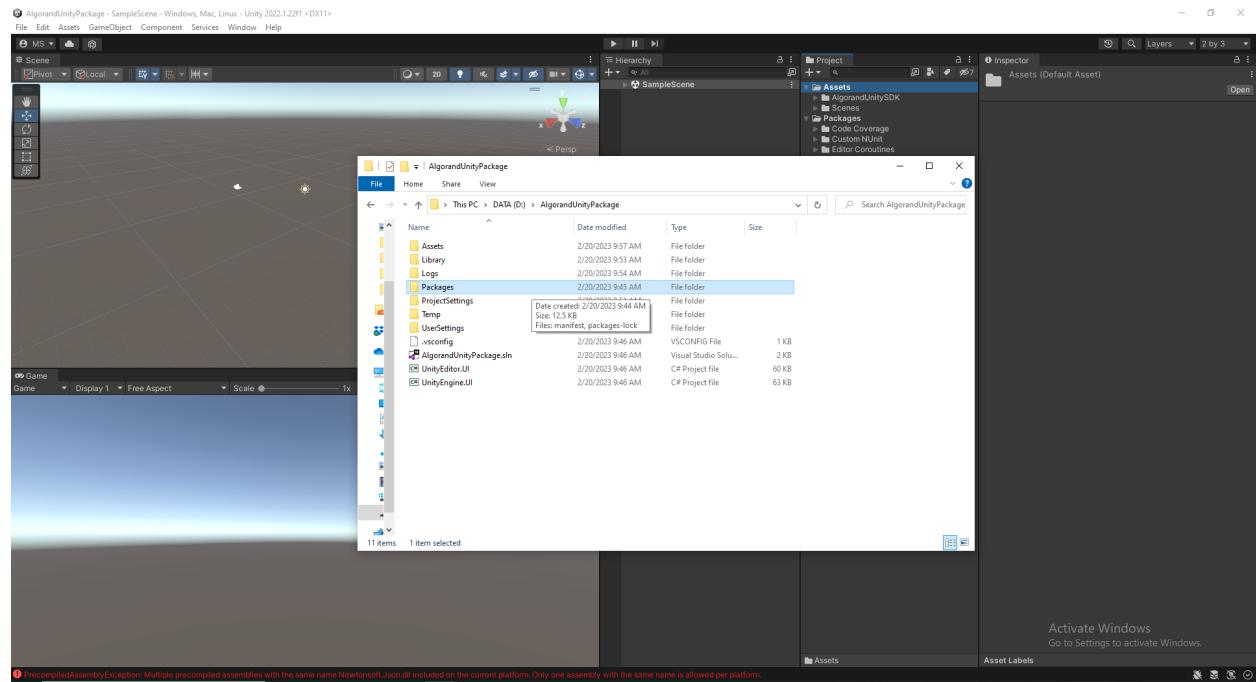
and find 'com.unity.nuget.newtonsoft-json' folder



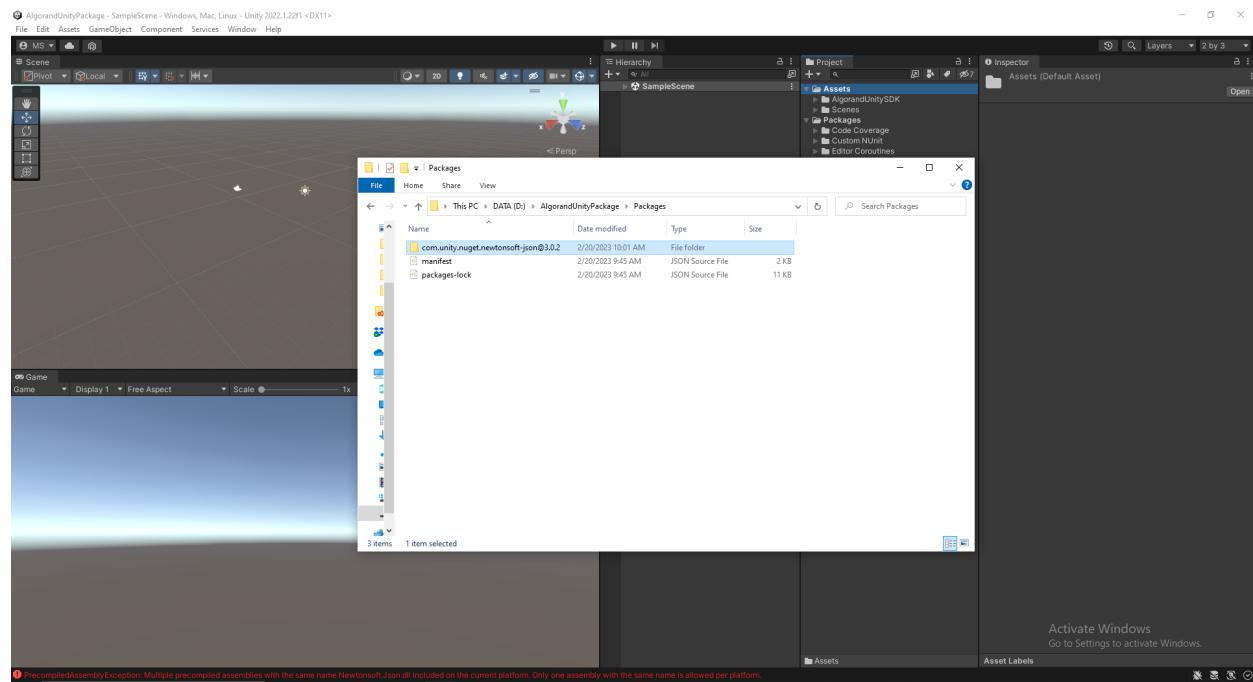
then 'Cut' (ctrl X) the whole folder



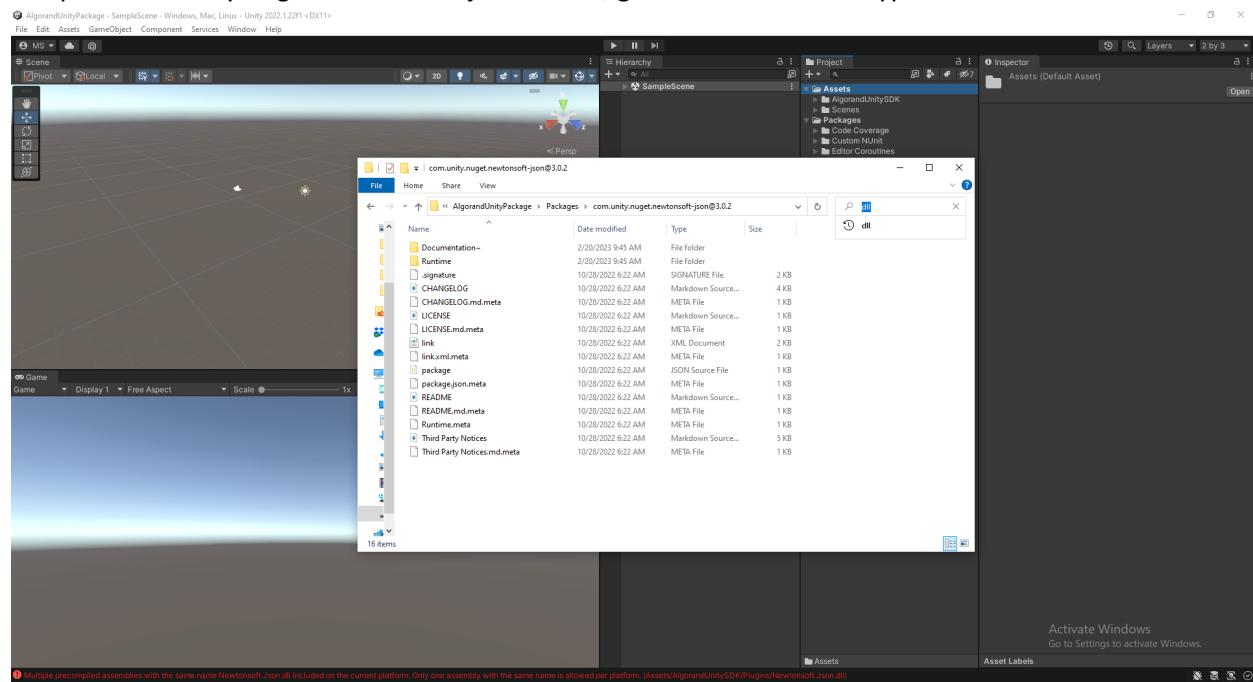
## 5. back to main folder, and find 'Packages', open it.....



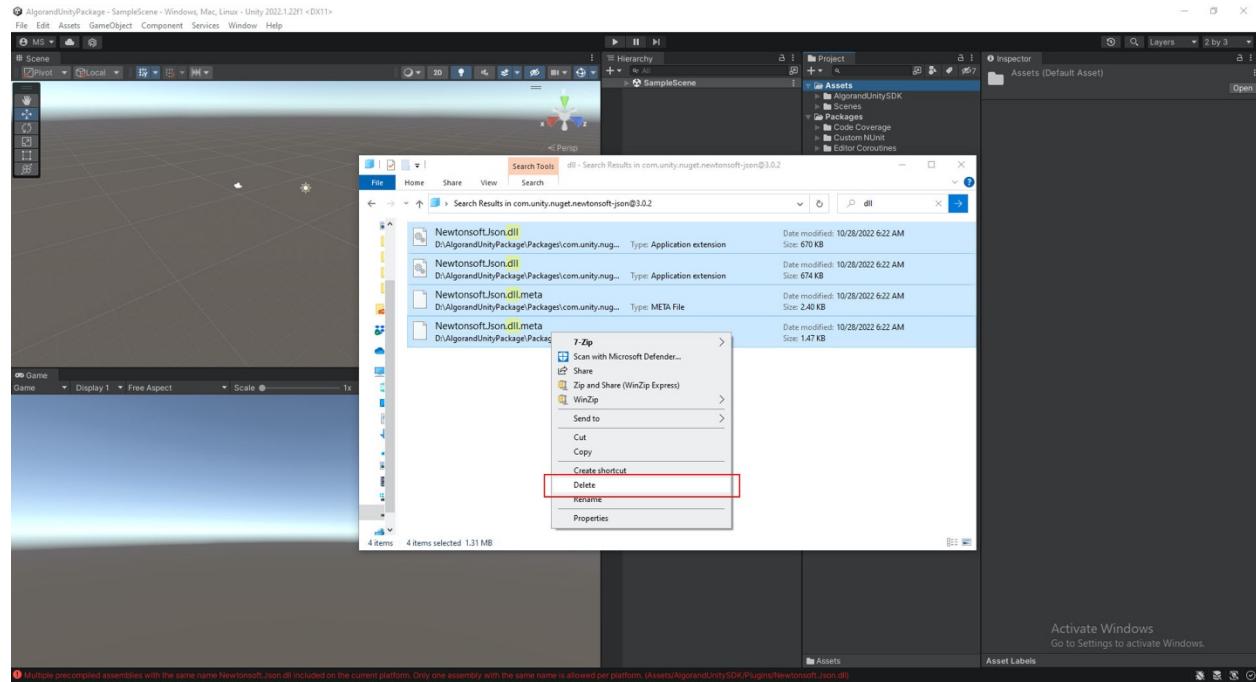
paste 'com.unity.nuget.newtonsoft-json' folder here.



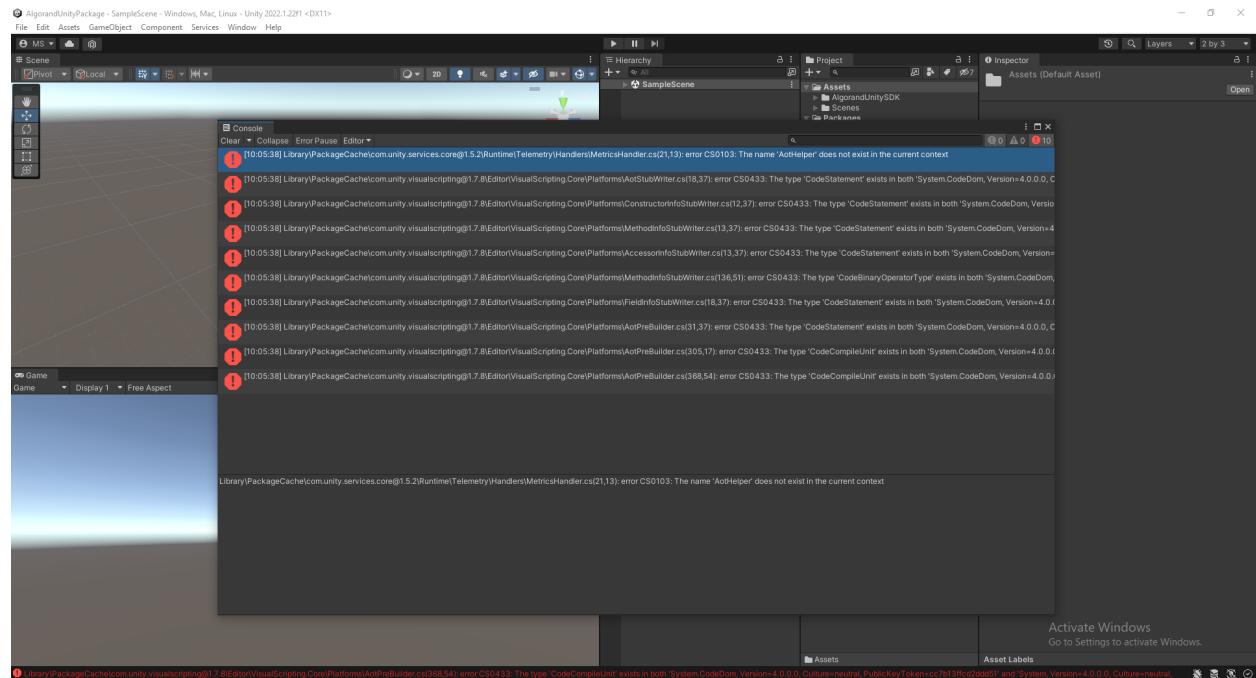
## 6. Open 'com.unity.nuget.newtonsoft-json' folder, go to search bar and type 'dll'



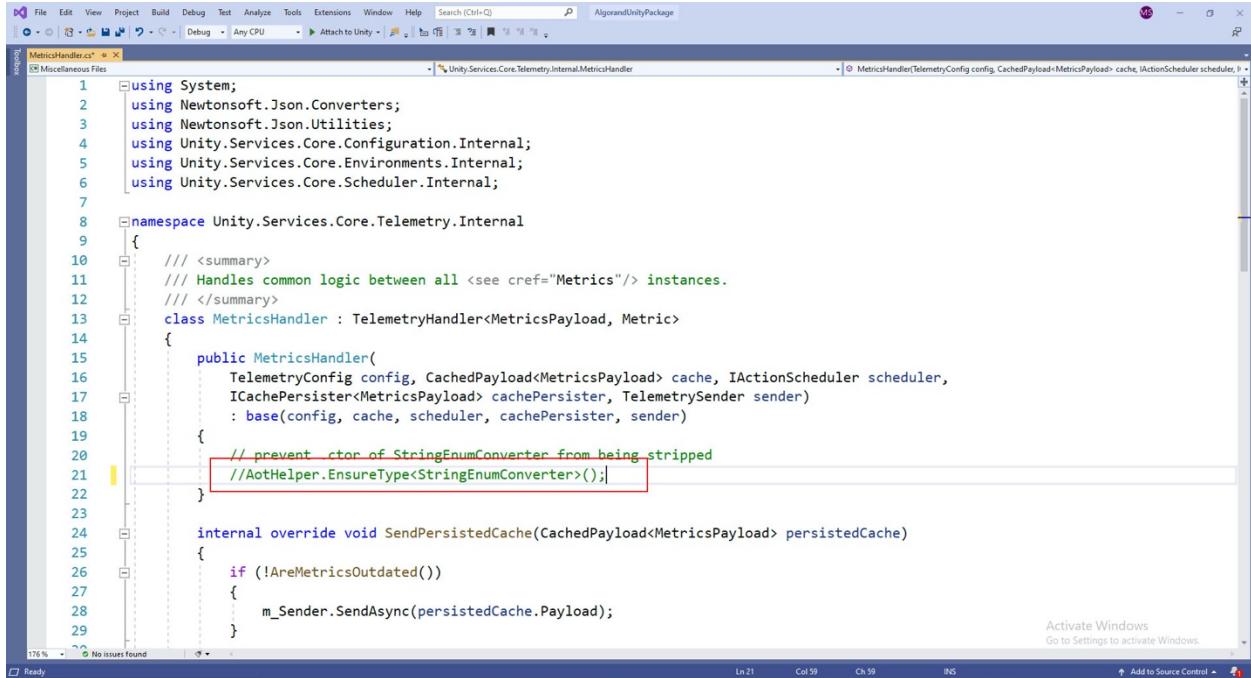
Select all these files and delete all



7. back to Unity, all newtonsoft-related errors should disappear. But now, we got couple other errors for i.e. 'AotHelper' and 'Visual Scripting' errors. Let's working on it.....



8. Follow these steps to remove 'AotHelper'-related errors. Open Console, double click the 1<sup>st</sup> AotHelper error. Once Visual Studio opened, disable this line by adding '//' in front of the code.

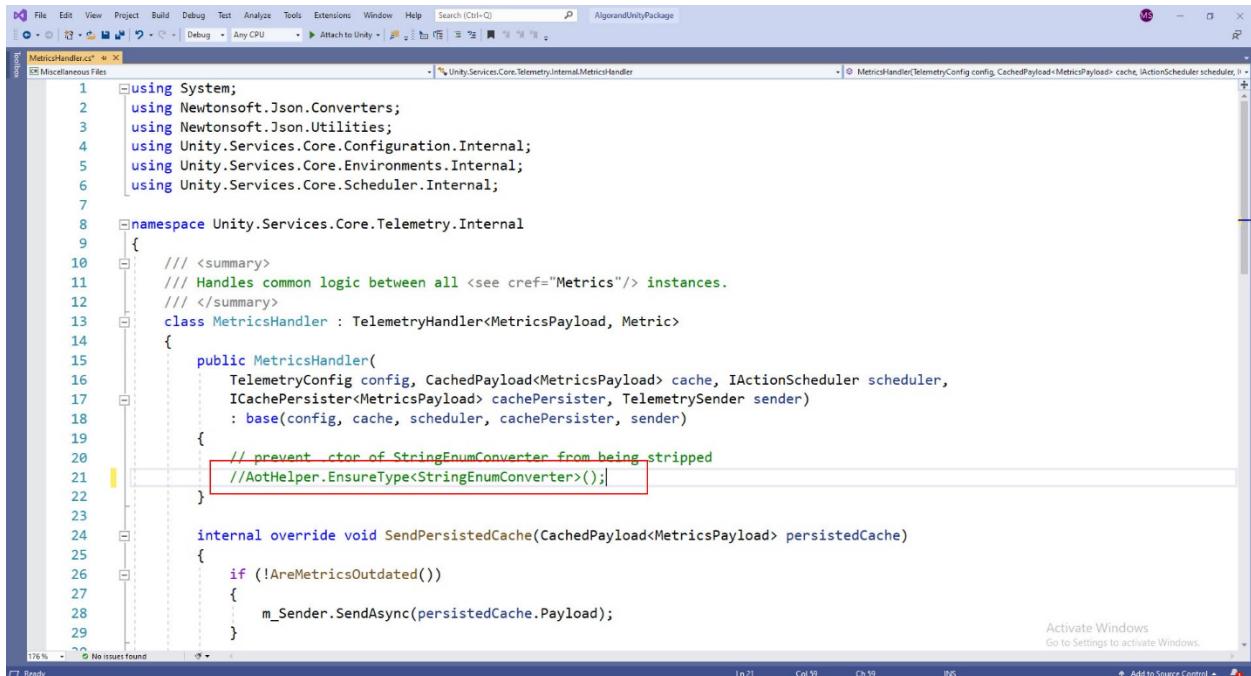


```
using System;
using Newtonsoft.Json.Converters;
using Newtonsoft.Json.Utilities;
using Unity.Services.Core.Configuration.Internal;
using Unity.Services.Core.Environments.Internal;
using Unity.Services.Core.Scheduler.Internal;

namespace Unity.Services.Core.Telemetry.Internal
{
    /// <summary>
    /// Handles common logic between all <see cref="Metrics"/> instances.
    /// </summary>
    class MetricsHandler : TelemetryHandler<MetricsPayload, Metric>
    {
        public MetricsHandler(
            TelemetryConfig config, CachedPayload<MetricsPayload> cache, IActionScheduler scheduler,
            ICachePersister<MetricsPayload> cachePersister, TelemetrySender sender)
            : base(config, cache, scheduler, cachePersister, sender)
        {
            // prevent ctor of StringEnumConverter from being stripped
            //AotHelper.EnsureType<StringEnumConverter>();
        }

        internal override void SendPersistedCache(CachedPayload<MetricsPayload> persistedCache)
        {
            if (!AreMetricsOutdated())
            {
                m_Sender.SendAsync(persistedCache.Payload);
            }
        }
    }
}
```

In my case, after I did the 1<sup>st</sup> step, same AotHelper error still occurred so we can repeat the step. Go to Console, double click the AotHelper error notification, visual studio will open and disable the AotHelper line:

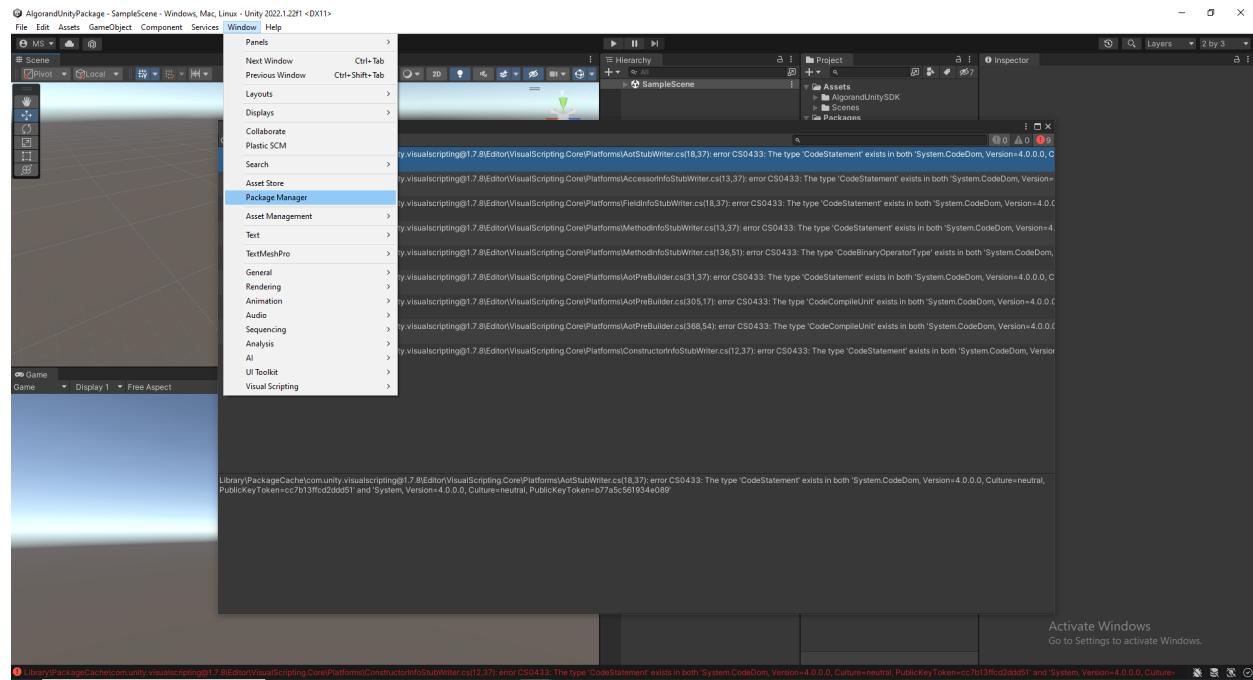


```
using System;
using Newtonsoft.Json.Converters;
using Newtonsoft.Json.Utilities;
using Unity.Services.Core.Configuration.Internal;
using Unity.Services.Core.Environments.Internal;
using Unity.Services.Core.Scheduler.Internal;

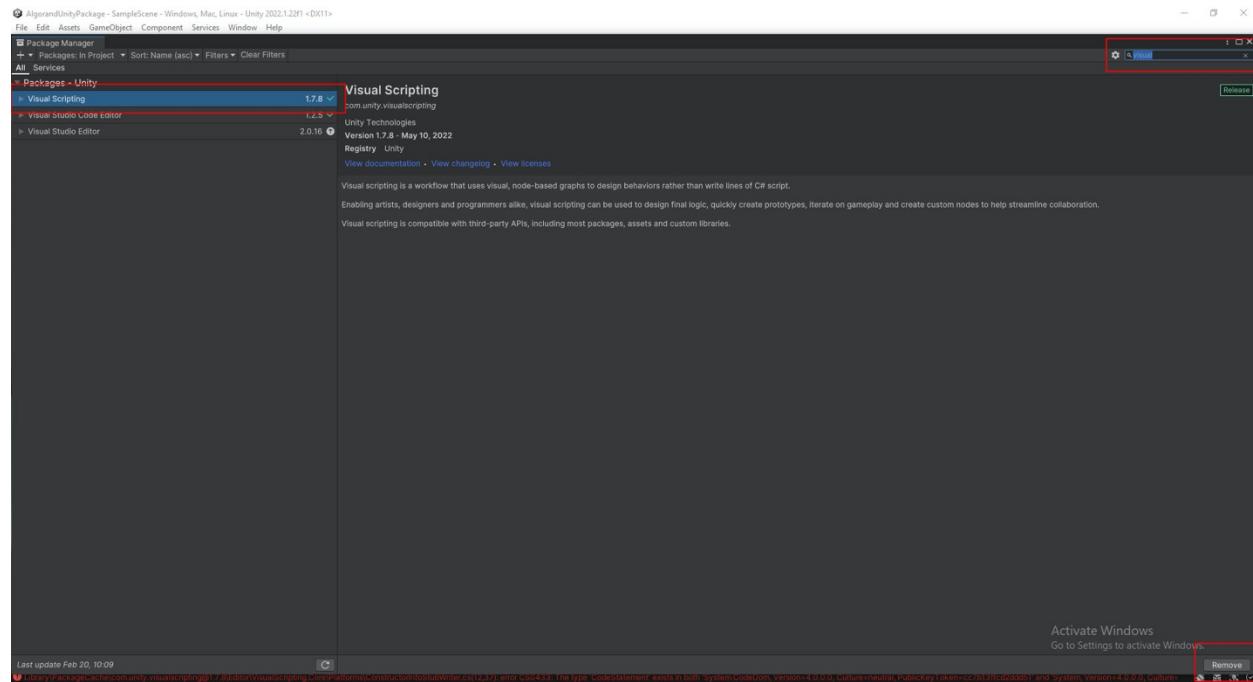
namespace Unity.Services.Core.Telemetry.Internal
{
    /// <summary>
    /// Handles common logic between all <see cref="Metrics"/> instances.
    /// </summary>
    class MetricsHandler : TelemetryHandler<MetricsPayload, Metric>
    {
        public MetricsHandler(
            TelemetryConfig config, CachedPayload<MetricsPayload> cache, IActionScheduler scheduler,
            ICachePersister<MetricsPayload> cachePersister, TelemetrySender sender)
            : base(config, cache, scheduler, cachePersister, sender)
        {
            // prevent ctor of StringEnumConverter from being stripped
            //AotHelper.EnsureType<StringEnumConverter>();
        }

        internal override void SendPersistedCache(CachedPayload<MetricsPayload> persistedCache)
        {
            if (!AreMetricsOutdated())
            {
                m_Sender.SendAsync(persistedCache.Payload);
            }
        }
    }
}
```

## 9. To fix Visual Scripting related error is pretty straightforward. Go to Window >> Package Manager



Search for 'visual' and you should find 'Visual Scripting' and hit 'Remove' button



## 10. Your Algorand-based Unity project should be ready to go !!!

Unity Package Source: <https://github.com/Vytek/AlgorandUnitySDK>

Feel free to reach me at: markus\_santoso@yahoo.com if you have any questions.

Happy Coding !!!