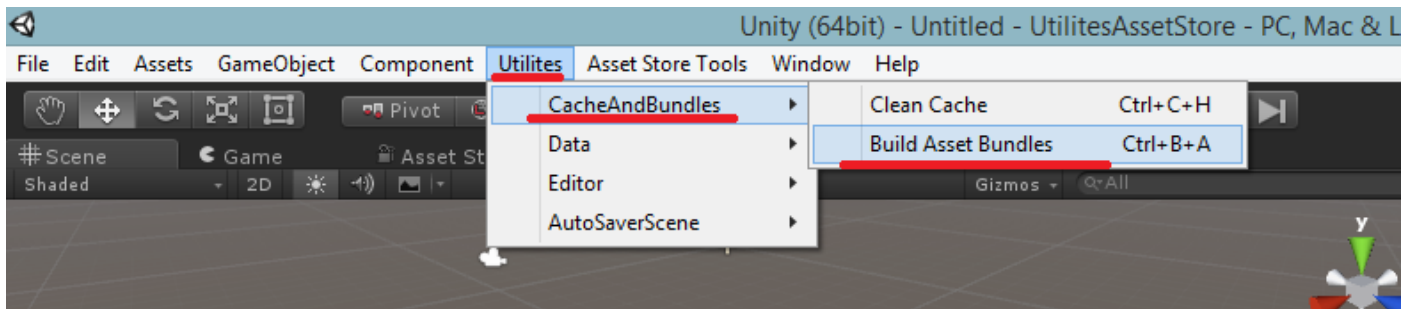


Project Utilites

Asset Bundles and Cache

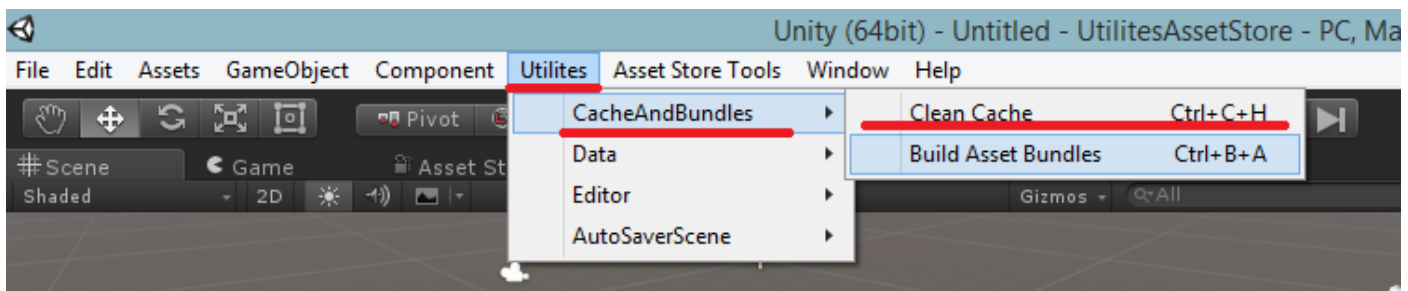
This plugin can create asset bundles with your content in your project for all of your using platforms

For build Asset Bundles look at the top menu and find tab “Utilites” in dropped menu choose “CacheAndBundles” and in new open menu choose “Build Asset Bundles”.



After build, you can find your asset bundles in your root folder of your project in “AssetBundles” folder like a “D:/MyProject/AssetBundles/”.

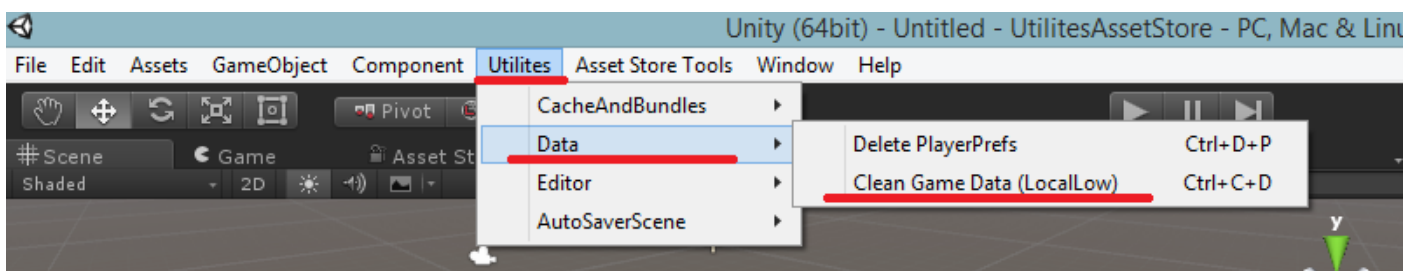
Sometimes you need to debug asset bundles for your project in editor. With clear cache you can clear all of downloaded and cached data on your computer (Mac\Win\Linux) but you haven’t time to find and write script for this. This functionality included in this plugin. For clear cache look at the top menu and find tab “Utilites” in dropped menu choose “CacheAndBundles” and in new open menu choose “Clean Cache”.



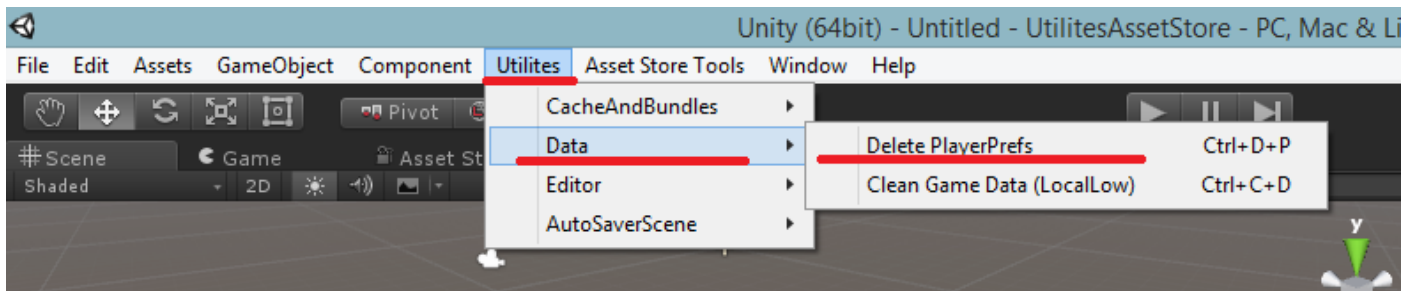
Cached Data in “PersistentDataPath” and PlayerPrefs

If you have some Data Manager, where you saved data in PersistentDataPath

(C:/Users/UserName/AppData/LocalLow/CompanyName/GameName/) with “Clean Game Data” function you can clean all data like a saves of your project. For clean game data look at the top menu and find tab “Utilites” in dropped menu choose “Data” and in new open menu choose “Clean Game Data (LocalLow)”.

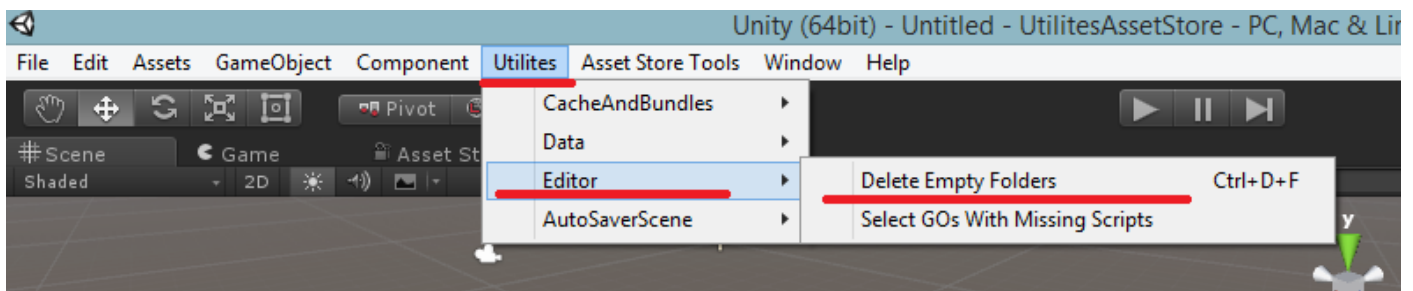


Of course, big count of projects using PlayerPrefs functionality and of course developer needs an easy access to clear PlayerPrefs data. For clear PlayerPrefs look at the top menu and find tab “Utilites” in dropped menu choose “Data” and in new open menu choose “Delete Player Prefs”.

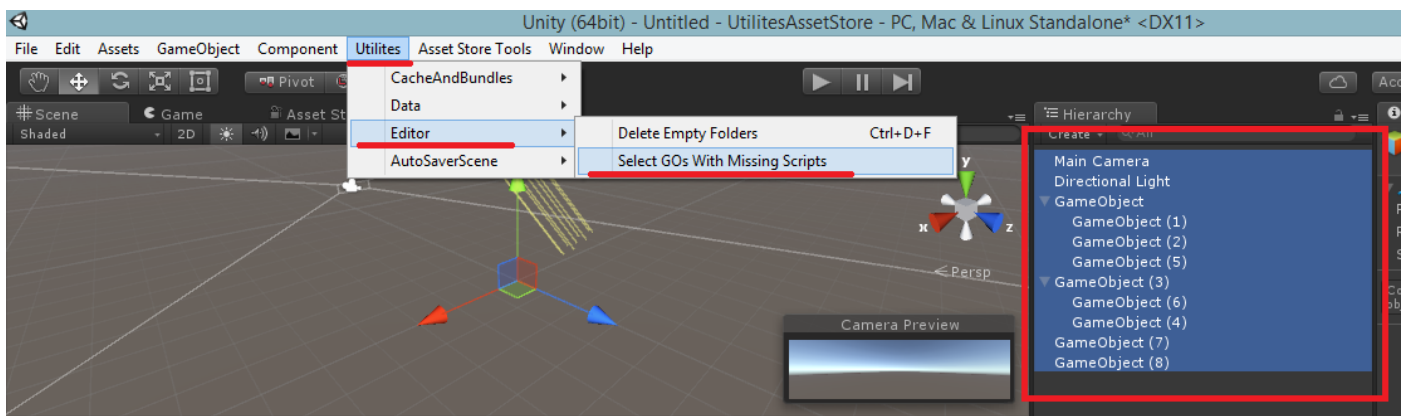


Project File Hierarchy Utilites

Every big projects have a big count of folders in project hierarchy, but sometimes hierarchy have a big count of empty folders. And if you want to delete all of empty folders you can use our functionality. For Delete Empty Folders look at the top menu and find tab “Utilites” in dropped menu choose “Editor” and in new open menu choose “Delete Empty Folders”.



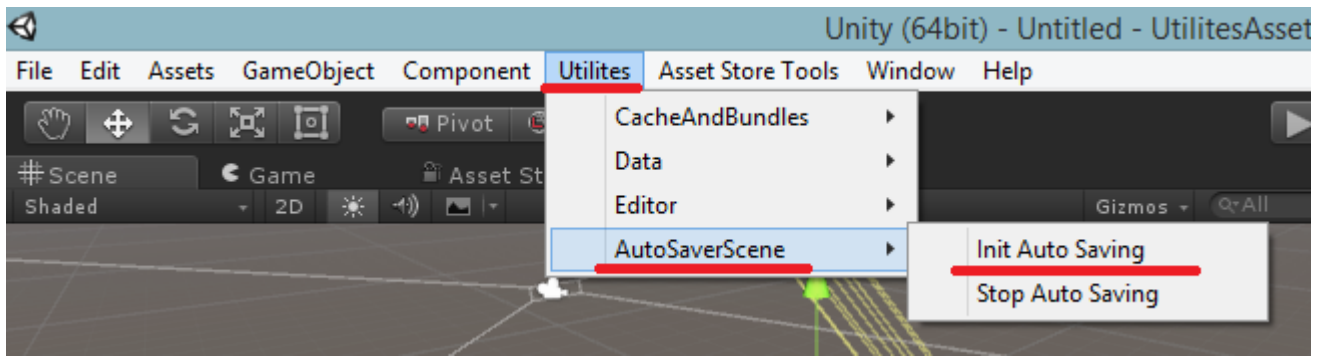
Sometimes in project we have missing scripts on Game Objects in scene hierarchy. Missing scripts create warnings and will drop project performance. For clear all of missing scripts you can select all object in scene hierarchy and look at the top menu and find tab “Utilites” in dropped menu choose “Editor” and in new open menu choose “Select GOs With Missing Scripts”. After that code will select all objects in scene where have missing scripts and you can remove them clicks right mouse button to script component header and choose “Remove Component”.



Warning: selected objects need to active in hierarchy. If missing scripts not found nothing object not be selected.

Auto Saving Scenes in Editor

All Unity developers think about the automatically saving scenes when they are working in editor. This plugin have functionality for this. For start auto saving look at the top menu and find tab “Utilites” in dropped menu choose “AutoSaverScene” and in new open menu choose “Init Auto Saving”. And scenes will automatically saved every 2 minutes (can be changed in code, by default is 2 minutes).



If you want to stop auto saving you can select “Stop Auto Saving”.

Warning: If you hide editor window, auto saver automatically stopped.

Debugger

Plugin have a functionality to debug logs in editor and builded project (saving into file like a Logs.txt). Logs can be three types: Simple, Warning and Error. For using this you can write logic in your scripts:

```
using UnityEngine;

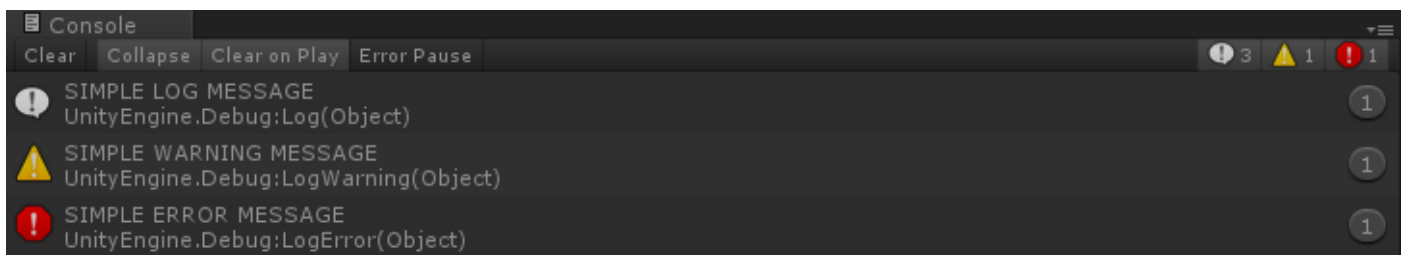
namespace FrostweepGames.Internal.Examples
{
    public class Example : MonoBehaviour
    {
        private const string PRIVATE_SECRET_KEY_FOR_DATA = "12345567890qwertyuiop";

        private void Start()
        {
            Utilites.DebugLog("SIMPLE LOG MESSAGE");
            Utilites.DebugWarning("SIMPLE WARNING MESSAGE");
            Utilites.DebugError("SIMPLE ERROR MESSAGE");

            string cryptoData = Utilites.Encrypt("MY SUPER ENCRYPTED DATA", PRIVATE_SECRET_KEY_FOR_DATA);
            Utilites.DebugLog(cryptoData);

            cryptoData = Utilites.Decrypt(cryptoData, PRIVATE_SECRET_KEY_FOR_DATA);
            Utilites.DebugLog(cryptoData);
        }
    }
}
```

Result In Editor:



Result in builded project:

```
SIMPLE LOG MESSAGE
Warning: SIMPLE WARNING MESSAGE
Error: SIMPLE ERROR MESSAGE
```

Cryptography

All project where have a DataManager like a saving progress in game have a encrypted files. If you want to encrypt your data you can use this utility for Encrypt and Decrypt data with Key – Value Pair. For using this you can write logic in your scripts:

```
using UnityEngine;

namespace FrostweepGames.Internal.Examples
{
    public class Example : MonoBehaviour
    {
        private const string PRIVATE_SECRET_KEY_FOR_DATA = "12345567890qwertyuiop";

        private void Start()
        {
            Utilites.DebugLog("SIMPLE LOG MESSAGE");

            Utilites.DebugWarning("SIMPLE WARNING MESSAGE");

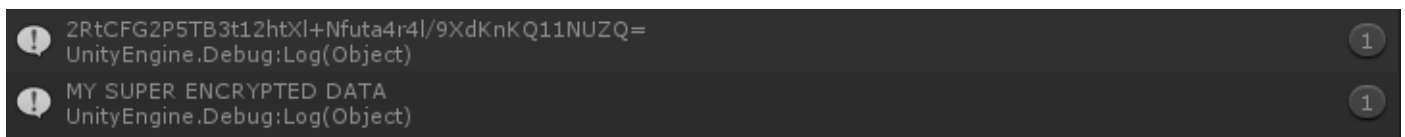
            Utilites.DebugError("SIMPLE ERROR MESSAGE");

            string cryptoData = Utilites.Encrypt("MY SUPER ENCRYPTED DATA", PRIVATE_SECRET_KEY_FOR_DATA);
            Utilites.DebugLog(cryptoData);

            cryptoData = Utilites.Decrypt(cryptoData, PRIVATE_SECRET_KEY_FOR_DATA);
            Utilites.DebugLog(cryptoData);
        }
    }
}
```

You can create your own key for encrypt\decrypt data.

Result in debug is:



```
2RtCFG2P5TB3t12htXl+Nfuta4r4l/9XdKnKQ11NUZQ=
UnityEngine.Debug:Log(Object)
MY SUPER ENCRYPTED DATA
UnityEngine.Debug:Log(Object)
```

We encrypt string “MY SUPER ENCRYPTED DATA” and receive that
“2RtCFG2P5TB3t12htXl+Nfuta4r4l/9XdKnKQ11NUZQ=”

After decrypting this “2RtCFG2P5TB3t12htXl+Nfuta4r4l/9XdKnKQ11NUZQ=” we receive “MY SUPER ENCRYPTED DATA”.