Analytics > Log & Crash Search > Unity Standalone SDK User Guide

Log & Crash Unity SDK sends logs to a Log & Crash Search collector server.

Below describe benefits and features of Log & Crash Unity SDK.

- Send logs to a collector server.
- Send crash logs occurred in an app to a collector server.
- Retrieve and search logs sent from Log & Crash Search.

Supporting Environment

- Common - Unity3D v4.0 or higher
- Android - Andorid SDK 2.3.3 API or higher

Download

Go to TOAST Document and download **Unity SDK**.

```
[DOCUMENTS] > [Download] > [Analytics > Log & Crash Search] > [Unity SDK]
```

Install

- Double-click downloaded toast-logncrash-android-unity-sdk.unitypackage and import it to your project.

Sample Description

To execute the sample, double-click **Assets > LogNCrash > Sample > SampleScene**. The sample describes examples of initialization, log delivery, and error occurrence.

Example

1. Initialize with LogNCrashSettings

Select **LogNCrash > Edit Settings** in the Unity menu to create LogNCrashSettings. Use AssetDatabase to define user's Appkey and SDK operations of LogNCrashSettings.

- Appkey: User's Appkey
- URL: Collector address: use https://api-logncrash.cloud.toast.com.
- Version: Log version
- Send Warning: Whether to collect warning logs occurred in Unity
- Send Error: Whether to collect error logs occurred in Unity
- Send Debug Warning: Whether to collect warning logs occurred by user's debug object use

in Unity
- Send Debug Error: Whether to collect error logs occurred by user's debug object use in Unity

Enter information for LogNCrashSettings and call Initialize function that has no LogNCrash object parameter, in order to initialize by reading such information of LogNCrashSettings.

```
using Toast.LogNCrash;
namespace Toast.LogNCrash
{
    public class SampleScript : MonoBehaviour
    {
        void Start ()
        {
            LogNCrash.Initialize ();
        }
    }
}
```

2. Initialize with Script Enter parameters to LogNCrash.Initialize to initialize. The parameters provide information on server address, Appkey, version, port, and whether to execute Send Thread Lock.

```
using Toast.LogNCrash;
namespace Toast.LogNCrash
{
    public class SampleScript : MonoBehaviour
    {
        void Start ()
        {
            LogNCrash.Initialize ("https://api-logncrash.cloud.toast.com",
"appkey", "1.0.0", 80,  true);
            LogNCrash.StartSendThread ();
        }
    }
}
```

- Appkey: User's Appkey
- URL: Collector address: set collector information of http and https
- Version: Log version
- Port: Set 80, 443, depending on the protocol
- SendThreadLock: Save logs, which occur when it is true, in a queue without sending to server before StartSendThread is called. Nevertheless, if a native crash occurs, unlock ThreadLock and send the logs.

API Details

Specify Custom Fields

```
public static void AddCustomField(string key, string val)
public static void RemoveCustomField(string key)
public static void RemoveAllCustomFields()
```

- Parameters
  - key: string
    - [in] key of custom field, custom key must start with an alphabet or a number, and include “A~Z, a~z, 0~9, - _”.
  - value: string
    - [in] value of custom field
- Note
  - Following keywords are occupied by SDK and hence cannot be used:
    - projectName
      - projectVersion
      - host
      - body
      - logLevel
      - userID
      - Platform
      - DmpData
      - Unity3D
      - Locale
      - CountryCode
      - SessionID
      - ExceptionType
      - NeloSDK
      - NetworkType
      - DeviceModel
    - DeviceID
      - @logType
  - When the value of a custom field is NULL or empty, SDKs do not send the field to a server.

Manage Default Setting

```
public static void SetLogSource(string value)
public static string GetLogSource()
```

- Get or newly specify a log source.

```
public static void SetLogType(string value)
public static string GetLogType()
```

- Get or newly specify a log type.

Filter Levels

- In Unity SDK, send logs of a FATAL level only by default setting. In ERROR or WARN levels, many logs may occur due to variables (such as time, route, and progress level.).
    - Send Error: Send ERROR-level logs occurred at a system.
    - Send Warning: Send WARN-level logs occurred at a system.
    - Send Debug Error: Send ERROR-level logs induced by a user.
    - Send Debug Warning: Send WARN-level logs induced by a user.

Example of API Use

- Refer to **html > index.html**.

Collect IP Address

```
public static void SetEnableHost:(bool flag)
```

- true: Get an ip address and save in the host field. false: Save"-" in the host field.

Send Logs

```
//send info log message
public static void Info(string strMsg)

//send debug log message
public static void Debug(string strMsg)

//send warn log message
public static void Warn(string strMsg)

//send fatal log message
public static void Fatal(string strMsg)

//send error log message
public static void Error(string strMsg)
```

- Parameters
    - strMsg: string
        - [in] Log messages to send

Crash Callbacks

```
public void Crash_Send_Complete_Callback(string message) {
        Debug.Log("Crash_Send_Complete_Callback : " + message);
}

void Start() {
        LogNCrashCallBack.ExceptionDelegate += Crash_Send_Complete_Callback;
}
```

- The ExceptionDelegate callback is called after crashes in Unity CSharp are sent to server: it is not called for native crashes.

Set User IDs

```
public static void SetUserId(string userID)
public static string GetUserID()
```

- Must set the value to get statistics per user.
- Parameter
  - userID: string
    - [in] User ID to sort out users

Remove Duplicates

The Remove Duplicates logic has been applied to general logs for 2.4.0 or higher SDKs: the logic is enabled with initialization.

In the case of general logs, do not send logs that have the same content in the body and logLevel.

For crash logs, do not send logs that have the same stackTrace and condition values.

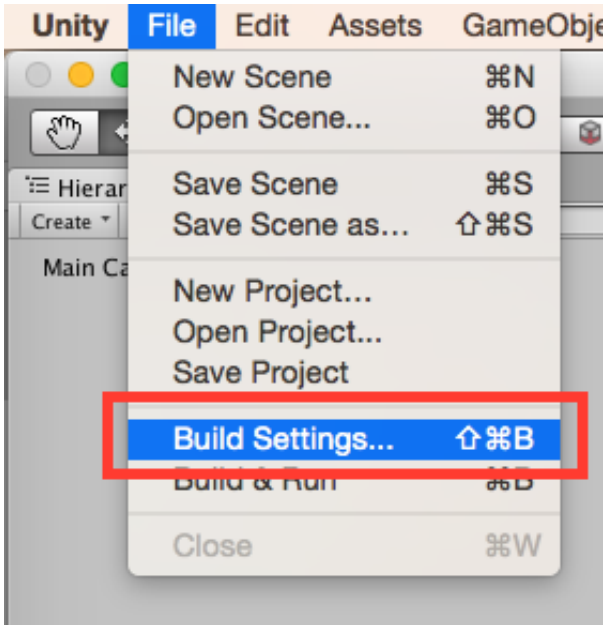
The function may be disabled by using the function below, after initialization.

```
public static void SetDeduplicate(bool flag)
```

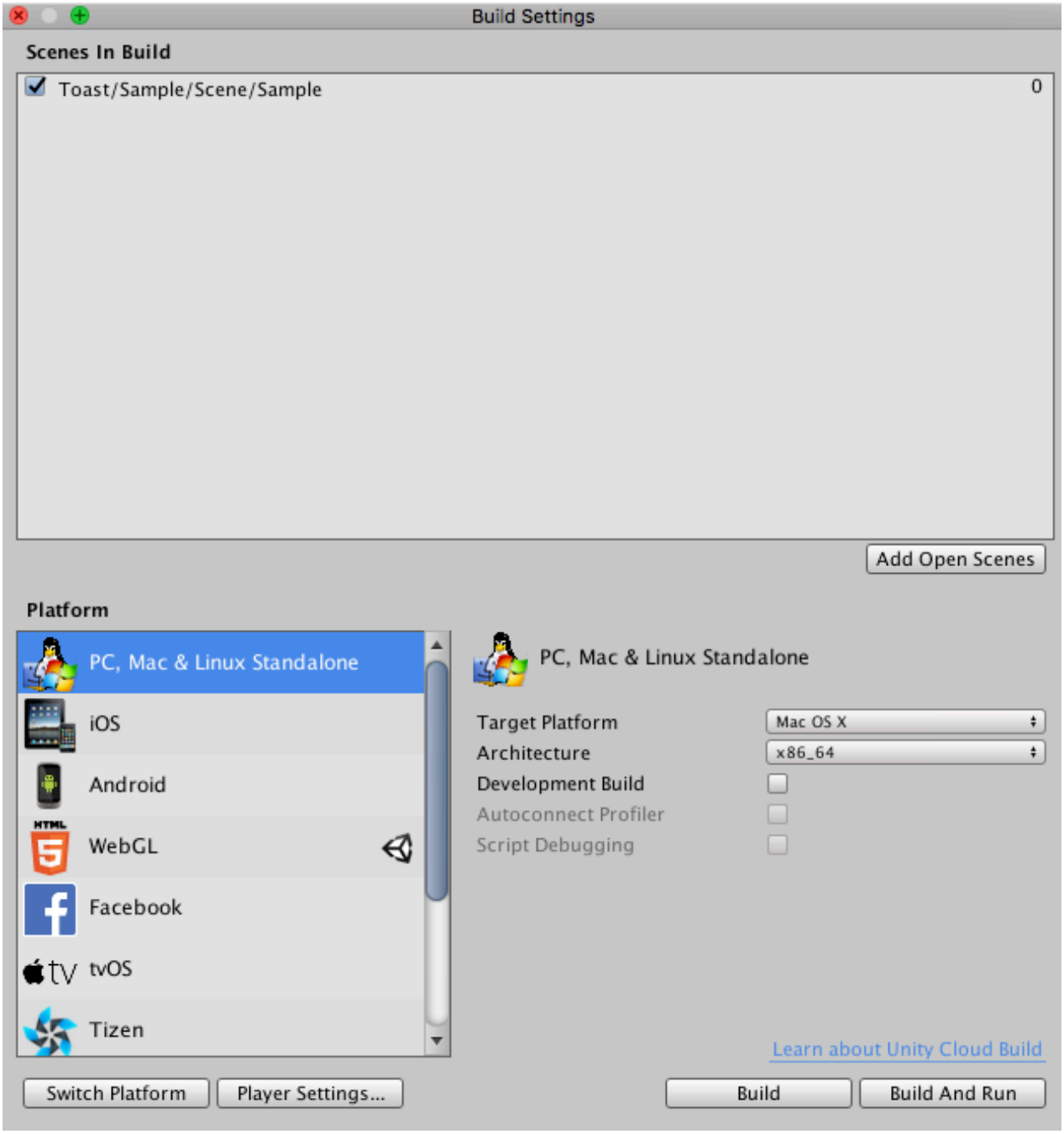
true: (Default) Remove duplicates is enabled
false: Remove duplicates is disabled

Build

1. Click **File > Build Settings**.



- Select **PC, Mac & Linux Standalone Platform** and click **Player Settings**.



2. Click **Build And Run** in the **Build settings**.

Use External CrashHandler

- Existing SDKs have deployed logMessageReceived during initialization to register CrashHandler of Unity for a LogNCrash callback function.
- The structure has been modified to allow applications to be made both for CrashHandler and external CrashHandler (refer to MultihandlerSample).

Applications

- Send a false parameter to the LogNCrash.SetCrashHandler function to prevent CrashHandler from being automatically registered.
- Must set before the initialize function.

```
LogNCrash.SetCrashHanlder (false);
LogNCrash.Initialize ();
```

- Then, use the LogNCrash.unity3dHandleException function to deliver CrashHandler parameters to LogNCrash object.

```
void OnEnable()
{
        Application.logMessageReceived += HandleLog;
}

void HandleLog(string logString, string stackTrace, LogType type)
{
        if (LogNCrash.isInitialized) {
            LogNCrash.unity3dHandleException (logString, stackTrace, type);
        }
}
```

Diverge Build Environment with AssetDataBase

- Click **LogNCrash > Edit Settings** in the menu and create AssetDataBase to save simple data.
- For a build using BuildPipeline.BuildPlayer, diverge the build environment by using LogNCrashSettings.Setter_BuildType and LogNCrashSettings.Getter_BuildType.

```
using UnityEditor;
using UnityEngine;
using Toast.LogNCrash.Implementation;

public class lncAndroidBuildPipeline: MonoBehaviour
{
    [MenuItem("Build/Build Android (Alpha)")]
    public static void AndroidAlphaBuildScript()
    {
        BuildPlayerOptions buildPlayerOptions = new BuildPlayerOptions();
        buildPlayerOptions.scenes = new[]
{"Assets/Toast/Sample/Scene/Command/commandScene.unity"};
        buildPlayerOptions.locationPathName = "AndroidBuild.apk";
        buildPlayerOptions.target = BuildTarget.Android;
        buildPlayerOptions.options = BuildOptions.AutoRunPlayer;

        LogNCrashSettings.Setter_BuildType =
LogNCrashSettings.BuildType.alpha;

        BuildPipeline.BuildPlayer(buildPlayerOptions);
    }

    [MenuItem("Build/Build Android (Real)")]
```

```
    public static void AndroidRealBuildScript()
    {
        BuildPlayerOptions buildPlayerOptions = new BuildPlayerOptions();
        buildPlayerOptions.scenes = new[]
{"Assets/Toast/Sample/Scene/Command/commandScene.unity"};
        buildPlayerOptions.locationPathName = "AndroidBuild.apk";
        buildPlayerOptions.target = BuildTarget.Android;
        buildPlayerOptions.options = BuildOptions.AutoRunPlayer;

        LogNCrashSettings.Setter_BuildType = LogNCrashSettings.BuildType.real;

        BuildPipeline.BuildPlayer(buildPlayerOptions);
    }
}
```

- LogNCrash operations depend on values saved in AssetDataBase by order.

```
using Toast.LogNCrash.Implementation;

void Start () {
                if (LogNCrashSettings.Getter_BuildType ==
LogNCrashSettings.BuildType.real) {
                        SetReal ();
                } else if (LogNCrashSettings.Getter_BuildType ==
LogNCrashSettings.BuildType.alpha) {
                        SetAlpha ();
                } else {
                        UnityEngine.Debug.Log ("Default Type");
                }
}
```

- There are five build types.

```
public enum BuildType{
        real, alpha, beta, development, test
    }
```