GAMEDNA

GOOGLE ANALYTICS

PLUGIN FOR UE4

Contents

1. License

gameDNA studio grants you a non-exclusive, non-transferable, non-sublicensable license for a single User to use, reproduce, display, perform, and modify the Google Analytics Plugin for Unreal Engine 4 ® for any lawful purpose (the "License"). The License becomes effective on the date you buy Google Analytics Plugin for Unreal Engine 4 ®. The License does not grant you any title or ownership in the Licensed Technology.

You may Distribute the Google Analytics Plugin for Unreal Engine 4 ® incorporated in object code format only as an inseparable part of a Product to end users. The Product may not contain any Plugin Content in uncooked source format.

2. Introduction

Google Analytics Provider plugin for Unreal Engine 4 ® lets you integrate out-of-the-box Google Analytics into your Unreal Engine project for all platforms. It fully implements Unreal Analytics Provider system so you can use all available features exposed by this (including Analytics Blueprint Library).

Required Unreal Engine 4.10 or above because of Android Plugin support!

Current plugin version: 1.6.0

Support: support@gamednastudio.com

FEATURES:

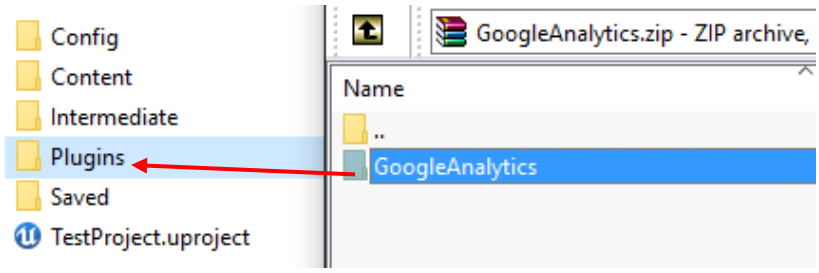
- Two special nodes only for Google Analytics Provider plugin: *Record Google Event* and *Record Google Screen*
- Implements Unreal Analytics Provider system
- All features are exposed to Blueprints
- No additional libraries are needed!
- Real-time sessions tracking (you can see who is currently playing your game!)
- Tracking events and screens
- Registering In-App Purchases
- Registering user progression
- Sending info about age, gender, location and interests
- Automatic errors, exceptions and crash reporting
- Automatic system info tracking (e. g. device, operating system, app version)
- Automatic caching events when user is offline and send them to Google Analytics when user is online
- And more features provided by Google Analytics

WHAT YOU GET:

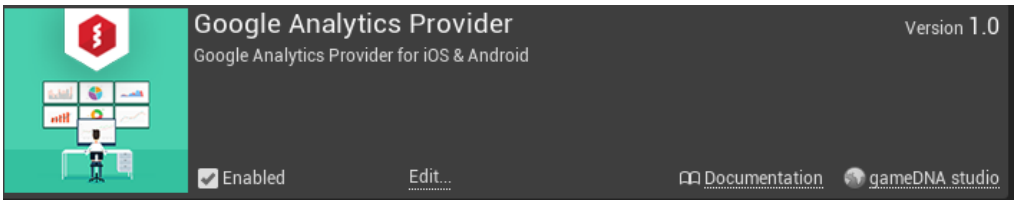
- Full source code
- Compatible with all UE4 platforms (all-in-one)
- Easy plugin setup
- All future updates
- Support via e-mail (support@gamednastudio.com)

3. Getting started

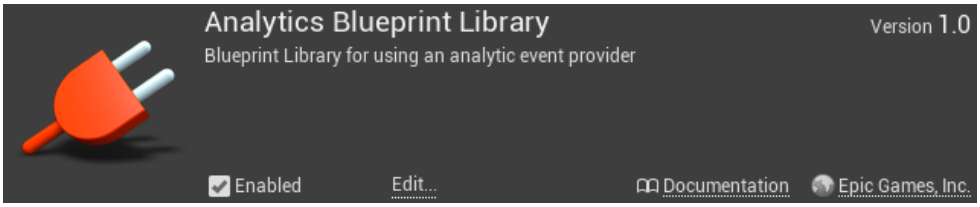
1. Unpack plugin archive to *Plugins* folder in your Unreal project folder and start editor.



2. Enable *Google Analytics Provider in Edit -> Plugins -> Analytics -> Project -> Google Analytics Provider*.



3. If you want to use Blueprint nodes for *Google Analytics Provider*, go to *Edit -> Plugins -> Built-In -> Analytics* and enable *Analytics Blueprint Library*.



4. Open *Config/DefaultEngine.ini* file and add following lines:

```
[Analytics]
ProviderModuleName=GoogleAnalytics
TrackingIdAndroid=<ENTER YOUR TRACKING ID FOR ANDROID>
TrackingIdIOS=<ENTER YOUR TRACKING ID FOR IOS>
TrackingIdUniversal=<ENTER YOUR TRACKING ID FOR PLATFORMS OTHER THAN
IOS OR ANDROID>

[AnalyticsDevelopment]
ProviderModuleName=GoogleAnalytics
TrackingIdAndroid=<ENTER YOUR TRACKING ID FOR ANDROID>
TrackingIdIOS=<ENTER YOUR TRACKING ID FOR IOS>
TrackingIdUniversal=<ENTER YOUR TRACKING ID FOR PLATFORMS OTHER THAN
IOS OR ANDROID>
```

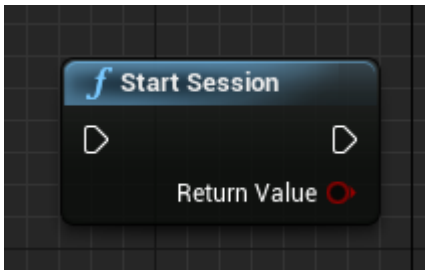
```
[AnalyticsTest]
ProviderModuleName=GoogleAnalytics
TrackingIdAndroid=<ENTER YOUR TRACKING ID FOR ANDROID>
TrackingIdIOS=<ENTER YOUR TRACKING ID FOR IOS>
TrackingIdUniversal=<ENTER YOUR TRACKING ID FOR PLATFORMS OTHER THAN
IOS OR ANDROID>
```

TIP: *Google Analytics* account for platforms other than iOS & Android should be set as website.

5. That's all! *Google Analytics Provider* works now! ☺

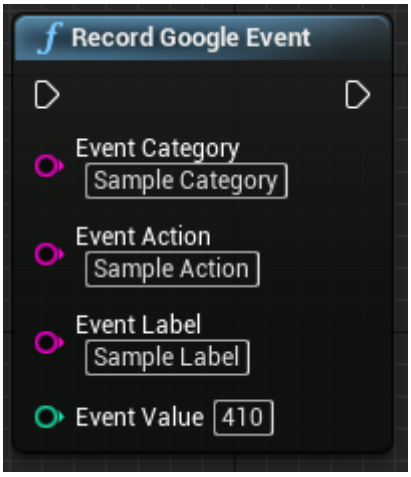
4. Recording events in Blueprints

Before recording any events you have to call the *Start Session* function. You should do it before any other Google Analytics function.



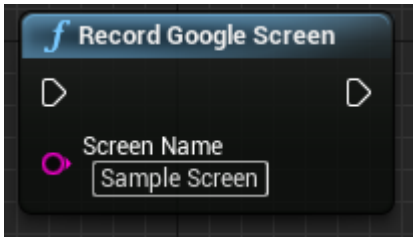
This plugin implements two special nodes only for Google Analytics.

a) **Record Google Event** - events are a useful way to collect data about a user's interaction with interactive components of your game, like button presses or the use of a particular item in a game.



b) **Record Google Screen** - screens in Google Analytics represent content users are viewing within your game. The equivalent concept in web analytics is a

pageview. Measuring screen views allows you to see which content is being viewed most by your users, and how they are navigating between different pieces of content. A screen view consists of a single string field that will be used as the screen name in your Google Analytics reports.



Google Analytics Provider plugin fully implements Unreal Analytics Provider system so you can use all available features exposed by this (including Analytics Blueprint Library).

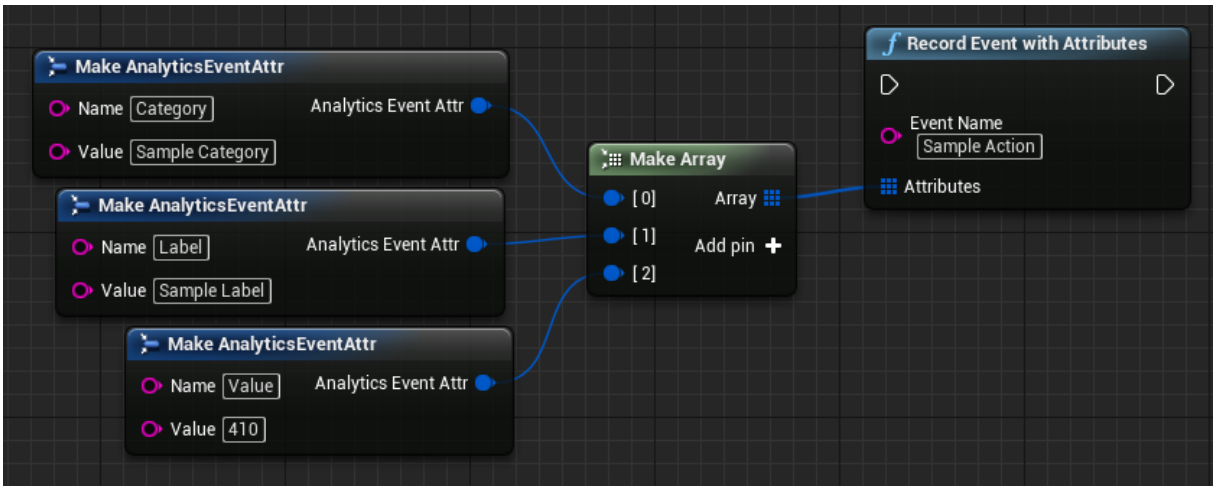
You can learn more about Unreal Analytics Provider here:
https://docs.unrealengine.com/latest/INT/Gameplay/Analytics/index.html

and about Analytics Blueprint Library here:
https://docs.unrealengine.com/latest/INT/Gameplay/Analytics/Blueprints/index.html

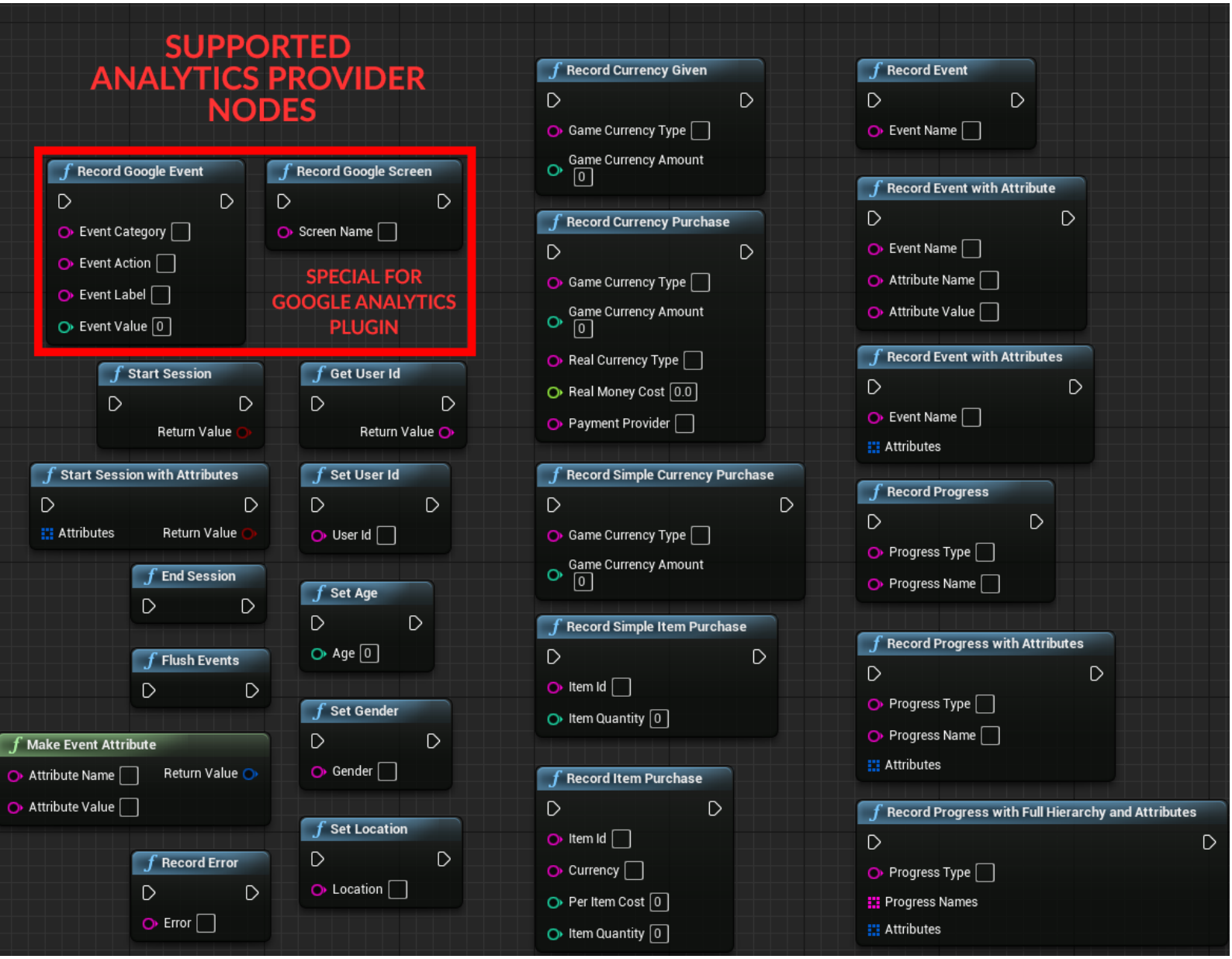
LIMITATIONS:
Google Analytics supports only following attributes for events:

- Event Category (String)
- Event Action (String) – known as Event Name in Unreal Analytics Provider system
- Event Label (String)
- Event Value (Integer)

So if you are using standard Unreal Analytics Provider functions, you can make something like this and that's exactly what special Record Google Event node is.



5. Supported Analytics Provider Nodes



6. Recording events in C++

You can use Unreal Analytics Provider for C++ or simply call special functions for Google Analytics Provider plugin.

Add the following to your **.Build.cs* file private dependency module name:

```
PrivateDependencyModuleNames.AddRange(new string[] { "GoogleAnalytics" });
```

Import these headers anywhere in source code project:

```
#include "GoogleAnalyticsBlueprintLibrary.h"
#include "Runtime/Analytics/Analytics/Public/Analytics.h"
#include
"Runtime/Analytics/Analytics/Public/Interfaces/IAnalyticsProvider.h"
```

a) Start session:

```
FAnalytics::Get().GetDefaultConfiguredProvider()->StartSession();
```

b) Record Google Event:

```
UGoogleAnalyticsBlueprintLibrary::RecordGoogleEvent(TEXT("Event Category"),
TEXT("Event Action"), TEXT("Event Label"), <INTEGER VALUE>);
```

c) Record Google Screen:

```
UGoogleAnalyticsBlueprintLibrary::RecordGoogleScreen(TEXT("Screen Name"));
```