Unity

Getting Started

Accengage SDK allows you to track the execution and display of In-App and to handle Push notifications of your application.

This Unity documentation explains the various methods available and how to use them.

In order to understand more deeply the features of the SDK, please refer also to the native iOS and Android documentations.

Plugin Version 2.0.0:

- Android SDK version 3.6.0
- iOS SDK version 6.1.1

Changelog

2.0 (Major) - 30 June 2017 - Download

- Complete rework of the Unity plugin
- Features
    - In-App Notifications
    - Local Notifications
    - Push Notifications
    - Analytics
- Use 6.1.1 iOS SDK
- Use 3.6.0 Android SDK

Integration

Common Integration

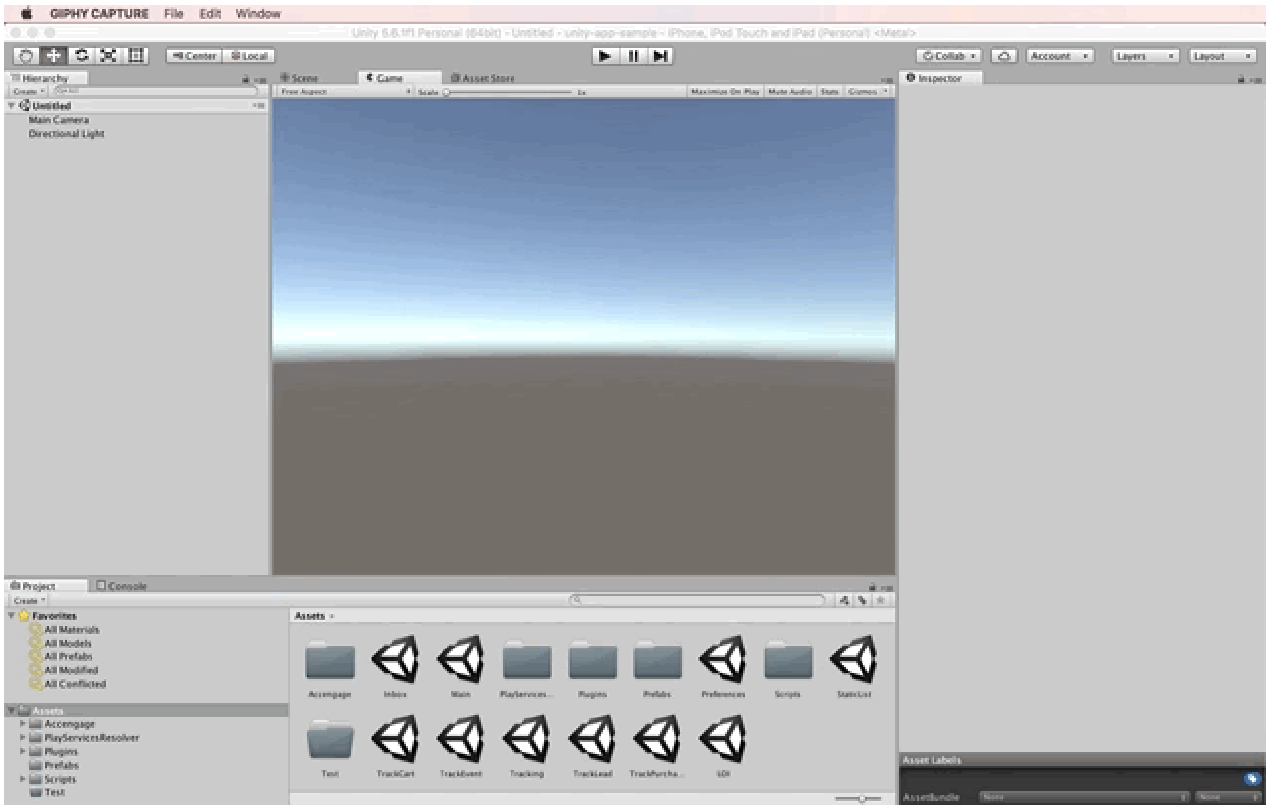
Download our last Unity Plugin Package.

In Unity, go to *Assets* > *Import Package* > *Custom Package...* and choose the Accengage SDK Unity Package.

Android specific integration

First, go to *Assets* > *Play Services Resolver* > *Android Resolver* > *Resolve Client Jars* and wait the Google Play Services and Support Library dependencies resolution. AAR dependencies should be in *Assets/Plugins/Android* folder.

Go to *Window* > *Accengage* > *Android* > *Update Manifests*. This will update the Accengage SDK manifests with your package name. Apply this update each time you change the package name or updating the Accengage Unity Plugin.



Then go to *Window* > *Accengage* > *Android* > *Configuration*. Then set Accengage SDK settings. Partner Id and Private Key is mandatory. The GCM Sender Id need to be prefixed with **gcm:** (gcm:123456789).

When settings are completed, click on *Update Project*.

iOS specific integration

Go to *Window* > *Accengage* > *iOS* > *Configuration*. Then set Accengage SDK settings. Partner Id and Private Key is mandatory.

Check Enable Push Notifications if you need it.

When settings are completed, click on *Update Project*.

Once your Unity project is compiled into xcode, it must be configured to work properly with our SDK.

In your target application in Xcode, add the Accengage.Framework to Embedded Binaries.

Then follow the prerequisites in the Getting started section of the iOS documentation.

> If you add a new scene, the Xcode project will recompile fully. It will be necessary to reconfigure Xcode as above.

Push

Android

To enable push follow the android documentation to create your GCM Sender Id and set it in Accengage Unity Plugin Configuration.

Notification Icon

By default, we are using your launcher icon as a logo for your notifications.

However, since Android KitKat, Google guidelines indicate that you **must** use a **white and transparent flat icon**.

If you created such icon, you can indicate its location to our SDK by adding the resource name (from drawable folder) in the Android configuration.

For instance, if you want to use drawable/ic_action_search.png, please replace "your_icon_name" by "ic_action_search"

Notification Accent Color

In order to be compliant with the Lollipop notifications, you can define an accent colour for your notifications.

To indicate to our SDK that it should use your accent colour, in the Android configuration plugin set the accent color with the choosen color.

Your accent colour can be a ARGB/RGB colour like the one defined in this example.

iOS

If you check Enable Push Notifications in configuration menu, the registration is automatic.

You can do it manually if you want:

```
Acc.GetiOS.RegisterForUserNotifications();
```

iOS 10 Rich Notifications

The latest version of the plugin supports iOS 10 Rich Notifications. However, to be able to receive these notifications you'll need to add the corresponding extension and the AccengageExtension framework manually to your project.

Please see our iOS Media attachments Documentation for the required steps and more information on this topic.

Suspend push display

You can suspend the push notification display by using:

```
Acc.Get.SetPushServiceEnabled(false);
```

In-App

The Accengage SDK provides you with notifications display without any additional code to write in order to support them.

It is possible to disable the display of InApp, for example if you are using splashscreen or if you have views in which you don't want them to appear using:

```
Acc.Get.SetInAppDisplayEnabled(false);
```

Tracking

Events

To use the additional parameters of the custom events for targeting purposes, they must formatted as a valid json string. Use a timestamp to represent a date.

If you want to track a specific event you can add to your code:

```
Acc.Get.TrackEvent(1001,
"{\"string-type-field\":\"value\",\"bool-type-field\":true,\"number-type-field\":720
1,\"date-type-field\":1498233848}");
```

Where 1001 is your event id.

Each event needs to be defined in the Accengage User Interface. Go to **Settings** > **Advanced Settings**, and add a new event of type *Custom* with the code used in trackEvent method. The label is only used for a display purpose.

Lead

If you want to send a Lead to Accengage Servers:

```
Acc.Get.TrackLead("Lead Label", "Lead Value");
```

Add to Cart

First, you will need to create an item to track, like this one:

```
var item = new Item
{
    Id = "ArticleID",
    Label = "Label",
    Category = "Category",
    Currency = CurrencyField.text,
    Price = 12.30,
    Quantity = 1
};
```

Note: Currency should be a valid 3 letters ISO4217 currency (EUR,USD,..)

Now you just need to specify the cart id and give us the item:

```
Acc.Get.TrackCart("CartID", item);
```

Purchase

You have 2 ways to send a Purchase tracking, with or without a list of Items.

Without items:

```
Acc.Get.TrackPurchase("Purchase ID", "Purchase Currency", 12.30);
```

Note: Currency should be a valid 3 letters ISO4217 currency (EUR,USD,..) and the last argument is the total price of this purchase.

If you want to add a list of items belonging to this purchase, you can:

```
Acc.Get.TrackPurchase("Purchase ID", "Purchase Currency", 12.30, items);
```

Update Device Info

You can create a device profile for each device in order to qualify the profile (for example, registering whether the user is opt in for or out of some categories of notifications). A device profile is a set of key/value that are uploaded to Accengage server. In order to update information about a device profile, add the following lines to your code:

```
var deviceInfos = new Dictionary<string, string>
{
    { "key1", "value1" },
    { "key2", "value2" }
};
Acc.Get.UpdateDeviceInfo(deviceInfos);
```

The keys and values must match Accengage user information field names and values to allow information to be correctly updated. To create these fields, follow the User Guide.

Views

You can identify each views of your application that the SDK can refer to using.

```
 Acc.Get.TrackScreenDisplay("MyView");
```

where MyView is the name of your view.

On iOS due to technical constraints, you must specify the stop of the tracking view.

It is necessary to use the method Acc.GetiOS.TrackScreenDismiss("MyView"), for example in an OnDestroy ().

Deeplinking

In the Accengage Interface, you can specify some custom parameters linked to click and/or display actions. The Accengage SDK sends a broadcast each time an item is displayed or clicked on in order to give you back these parameters.

Here is how to retrieve your custom parameters from any kind of notification.

Attach the OnAccCustomParameters method to the Main Camera.

```
void OnAccCustomParameters(string jsonString)
{
 AccCustomParameters accCustomParameters = AccCustomParameters.FromJsonString
(jsonString);
}
```

Other

Activate logs

In the Unity Android configuration or iOS configuration, check the *Enable SDK Logs* checkbox. You need to disable logs in production environment.