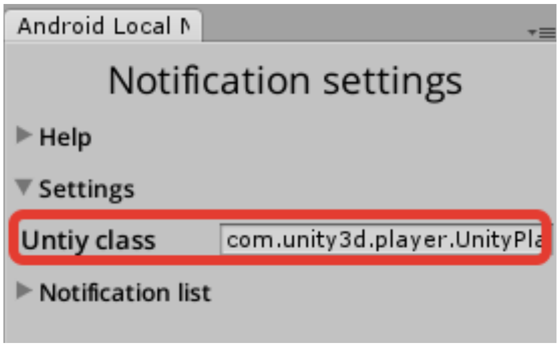
Ultimate Android Notifications : Local And Push

For latest documentation please refer to GitHub Wiki

Install

1. Import this plugin into your Unity project.
2. Check if you have **AndroidManifest.xml** in Assets/Plugins/Android folder.
   If you don't - add this manifest (https://yadi.sk/d/PIzCJa-jif8Zo) to *Assets/Plugins/ Android*folder.
   If you do - check if it contains UnityPlayerNativeActivity or the one that extends it.
   If you have UnityPlayerNativeActivity - you are good to go.
   If you have activity that extends **UnityPlayerNativeActivity**- set its full name (e.g. com.unity3d.player.UnityPlayerNativeActivity) in *Window->Ultimate Local Notifications -> Settings*



Untiy class com.unity3d.player.UnityPla

Creating notifications

The notifications are created using **NotificationBuilder** class. Its constructor takes 3 arguments (required notification fields) - id of the notification, title and notification text.

```
int id = 1;
string title = "Notification titile";
string body = "Notification body";

NotificationBuilder builder = new NotificationBuilder(id, title, body);
```

Then you can add extra options or edit the values of the existing using NotificationBuilder interface. All methods with description you can find in **NotificationBuilder.cs** file.

e.g.

```
int id = 1;
string title = "Notification titile";
string body = "Notification body";

TimeSpan inteval = new TimeSpan(2, 0, 0); //2 hour interval

NotificationBuilder builder = new NotificationBuilder(id, title, body);
builder.setTicker          ("New notification arrived!")
       .setDefaults        (NotificationBuilder.DEFAULT_ALL)
       .setAlertOnlyOnce   (true)
       .setDelay           (10000)     // show notification in 10 seconds
       .setRepeating       (true)
       .setAutoCancel      (true)
       .setInterval        (interval); // repeat notification every 2 hours
```

After notification has been built, you have to schedule it

```
AndroidNotifications.scheduleNotification(builder.build());
```

NOTE: If want to set custom notification icons - they must be located in*Assets/Plugins/Android/Notfications/res/drawable* folder (or one of those - for more info check out android developer guides on *drawable* folders).
Custom sounds must be located in *Assets/Plugins/Android/Notfications/res/raw* folder

Cancelling notifications (both repeating and one-time)

To cancel the notification, simply call AndroidNotifications.cancelNotification(...).

e.g.

```
//cancel notification with id 7
AndroidNotifications.cancelNotification(7);
```

Clearing shown notifications

To clear certain notification use AndroidNotifications.clear(int id).

```
// clear shown notification with id 7
AndroidNotifications.clear(7);
```

To clear all shown notifications use AndroidNotifications.clearAll().

```
// clear all shown notifications
AndroidNotifications.clearAll();
```

Updating notifications

To update one-time or repeating notification, schedule a notification with updated data but with ID of the notification you want to update.

Showing android toast notfication

To show a toast notification use AndroidNotifications.showToast(string text).

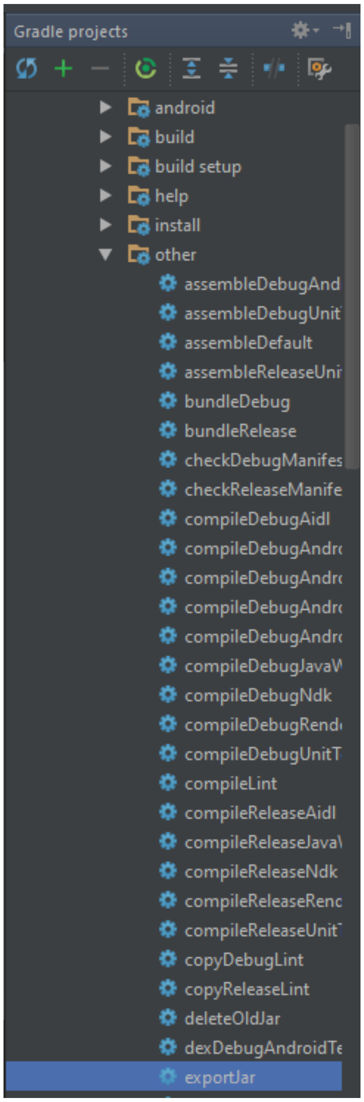
e.g.

```
AndroidNotifications.showToast("Download completed");
```

Modifying a plugin

Source code of the plugin is included in the package. You can easily extend it if you want. Java library is created with **AndroidStudio**. (you may also want to mofidy few paths in**build.gradle** file in **deleteOldJar** and **exportJar** tasks. Check comments in those tasks)

To export jar from AndroidStudio go to *Gradle Projects/Tasks/Other* and run **exportJar**task.



Gradle projects

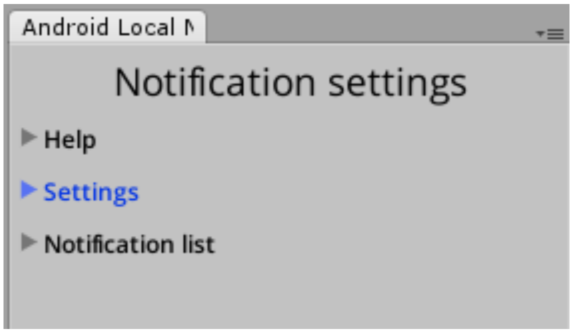
- android
- build
- build setup
- help
- install
- other
  - assembleDebugAnd
  - assembleDebugUnit
  - assembleDefault
  - assembleReleaseUni
  - bundleDebug
  - bundleRelease
  - checkDebugManifes
  - checkReleaseManife
  - compileDebugAidl
  - compileDebugAndr
  - compileDebugAndr
  - compileDebugAndr
  - compileDebugAndr
  - compileDebugJavaW
  - compileDebugNdk
  - compileDebugRend
  - compileDebugUnitT
  - compileLint
  - compileReleaseAidl
  - compileReleaseJava
  - compileReleaseNdk
  - compileReleaseRenc
  - compileReleaseUnit
  - copyDebugLint
  - copyReleaseLint
  - deleteOldJar
  - dexDebugAndroidTe
  - exportJar

In Unity plugin is under Plugins/Android/Notifications folder. It is stored as android library project so manifest, resources and libraries are merged with other during building.

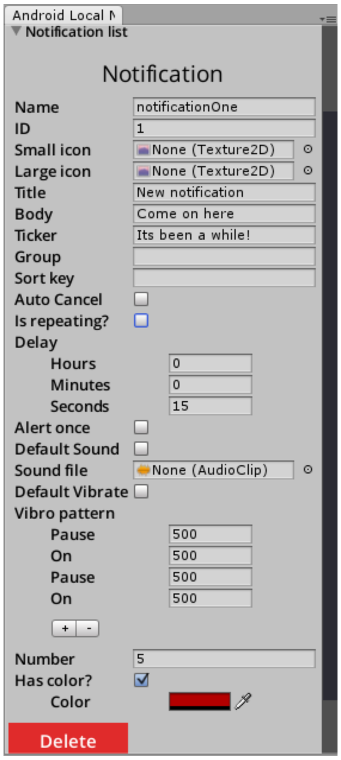
To debug this plugin in AndroidStudio add **Area730Log** log tag to you logcat filter.

Notification editor

Plugin comes with editor extension that allows you to create notifications without the line of code. To open the notification editor window go to *Window -> Android Local Notifications* .



In **Help** section you will find some useful links. In **Settings** section you can set custom Unity class if your activity extends *UnityPlayerNativeActivity*. In **Notification List** section you can add and modify notifications.



Name	notificationOne
ID	1
Small icon	None (Texture2D)
Large icon	None (Texture2D)
Title	New notification
Body	Come on here
Ticker	Its been a while!
Group	
Sort key	
Auto Cancel	□
Is repeating?	□
Delay	
Hours	0
Minutes	0
Seconds	15
Alert once	□
Default Sound	□
Sound file	None (AudioClip)
Default Vibrate	□
Vibro pattern	
Pause	500
On	500
Pause	500
On	500
Number	5
Has color?	☑
Color	🟥

When you set custom notification sound or icons in editor window - **they will be automatically copied to Notifications/res/drawable and Notifications/res/raw folders***. Though you will still need to add resized versions to drawable-hdpi and other folders using icon generators mentioned above*.

For detailed information on notification options please refer to official Android docs

Push Notification With OneSignal

To configure push notification for android platform follow next steps:

Create GMS application by following this tutorial instruction.

Do step 3 to config your AndroidManifest.xml

Go to *Assets/Area730/Notifications/PushNotification* drag and drop PushController.prefab or just add *CrossPlatformPushNotificationController.cs* script to your gameobject.

Fill the values in *CrossPlatformPushNotificationController.cs*.

After these steps you will be able to send push notification using OneSignal service.

Other

All classes are located in **Area730.Notifications** namespace

Example scene with sample code is included in the package (*Assets/Area730/Notifications/Examples*)

If you have any questions or suggestions – drop me a letter:
support@area730.com

Check out my other products