1. **PROJECT UNITY CALL NOTIFICATION (FOR PRESS EXIT BUTTON) FOR BOTH**

Android Studio project

Build.gradle

buildscript {  
 repositories {  
 google()  
 jcenter()  
 }  
 dependencies {  
 classpath **'com.android.tools.build:gradle:3.2.0'** }  
}  
allprojects {  
 repositories {  
 google()  
 jcenter()  
 flatDir {  
 dirs **'libs'** }  
 }  
}  
apply **plugin**: **'com.android.application'**dependencies {  
 implementation fileTree(**dir**: **'libs'**, **include**: [**'\*.jar'**])  
 implementation(**name**: **'support-compat-25.3.1'**, **ext**: **'aar'**)  
 implementation(**name**: **'unilocalnotification-release'**, **ext**: **'aar'**)  
}

AndroidManifest.xml

<**application  
 android:icon="@mipmap/app\_icon"  
 android:label="@string/app\_name"  
 android:theme="@style/UnityThemeSelector"**>  
 <**activity  
 android:name="com.yuan.noti.UnityPlayerActivity"  
 android:configChanges="mcc|mnc|locale|touchscreen|keyboard|keyboardHidden|navigation|orientation|screenLayout|uiMode|screenSize|smallestScreenSize|fontScale|layoutDirection|density"  
 android:hardwareAccelerated="false"  
 android:label="@string/app\_name"  
 android:launchMode="singleTask"  
 android:screenOrientation="fullSensor"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 <**meta-data  
 android:name="unityplayer.UnityActivity"  
 android:value="true"** />  
 </**activity**>

…

<**uses-permission android:name="android.permission.INTERNET"** />

UnityPlayerActivity.java

**public class** UnityPlayerActivity **extends** Activity  
{  
 **protected** UnityPlayer **mUnityPlayer**; *// don't change the name of this variable; referenced from native code  
 // Setup activity layout* @Override **protected void** onCreate(Bundle savedInstanceState)  
 {  
 requestWindowFeature(Window.***FEATURE\_NO\_TITLE***);  
 **super**.onCreate(savedInstanceState);  
 **mUnityPlayer** = **new** UnityPlayer(**this**);  
 setContentView(**mUnityPlayer**);  
 **mUnityPlayer**.requestFocus();  
 }  
 @Override **protected void** onNewIntent(Intent intent)  
 {  
 *// To support deep linking, we need to make sure that the client can get access to  
 // the last sent intent. The clients access this through a JNI api that allows them  
 // to get the intent set on launch. To update that after launch we have to manually  
 // replace the intent with the one caught here.* setIntent(intent);  
 }  
 *// Quit Unity* @Override **protected void** onDestroy ()  
 {  
 **mUnityPlayer**.destroy();  
 **super**.onDestroy();  
 }  
 *// Pause Unity* @Override **protected void** onPause()  
 {  
 **super**.onPause();  
 **mUnityPlayer**.pause();  
 }  
 *// Resume Unity* @Override **protected void** onResume()  
 {  
 **super**.onResume();  
 **mUnityPlayer**.resume();  
 }  
 @Override **protected void** onStart()  
 {  
 **super**.onStart();  
 **mUnityPlayer**.start();  
 }  
 @Override **protected void** onStop()  
 {  
 **super**.onStop();  
 **mUnityPlayer**.stop();  
 }  
 *// Low Memory Unity* @Override **public void** onLowMemory()  
 {  
 **super**.onLowMemory();  
 **mUnityPlayer**.lowMemory();  
 }  
 *// Trim Memory Unity* @Override **public void** onTrimMemory(**int** level)  
 {  
 **super**.onTrimMemory(level);  
 **if** (level == ***TRIM\_MEMORY\_RUNNING\_CRITICAL***)  
 {  
 **mUnityPlayer**.lowMemory();  
 }  
 }  
 *// This ensures the layout will be correct.* @Override **public void** onConfigurationChanged(Configuration newConfig)  
 {  
 **super**.onConfigurationChanged(newConfig);  
 **mUnityPlayer**.configurationChanged(newConfig);  
 }  
 *// Notify Unity of the focus change.* @Override **public void** onWindowFocusChanged(**boolean** hasFocus)  
 {  
 **super**.onWindowFocusChanged(hasFocus);  
 **mUnityPlayer**.windowFocusChanged(hasFocus);  
 }  
 *// For some reason the multiple keyevent type is not supported by the ndk.  
 // Force event injection by overriding dispatchKeyEvent().* @Override **public boolean** dispatchKeyEvent(KeyEvent event)  
 {  
 **if** (event.getAction() == KeyEvent.***ACTION\_MULTIPLE***)  
 **return mUnityPlayer**.injectEvent(event);  
 **return super**.dispatchKeyEvent(event);  
 }  
  
 *// Pass any events not handled by (unfocused) views straight to UnityPlayer* @Override **public boolean** onKeyUp(**int** keyCode, KeyEvent event) { **return mUnityPlayer**.injectEvent(event); }  
 @Override **public boolean** onKeyDown(**int** keyCode, KeyEvent event) { **return mUnityPlayer**.injectEvent(event); }  
 @Override **public boolean** onTouchEvent(MotionEvent event) { **return mUnityPlayer**.injectEvent(event); }  
 */\*API12\*/* **public boolean** onGenericMotionEvent(MotionEvent event) { **return mUnityPlayer**.injectEvent(event); }  
}

Unity Notification use librabries:

**'support-compat-25.3.1'**, **ext**: **'aar'**

**'unilocalnotification-release'**, **ext**: **'aar'**

**\*\*\* START FROM HERE**  
- Import package **UniLocalNotification**.unitypackage

- Usage:

Call Register function from class NotiManager.cs

public class NotiManager : MonoBehaviour  
{  
 private void Awake()  
 {  
 UniLocalNotification.Initialize();  
 }  
 public void Register(int timeDelay)  
 {  
 UniLocalNotification.Register(timeDelay, "This is your notification", "Game Notifications");  
 }  
 public void CheckIsLocalNotificaionPermitted()  
 {  
 bool isPermitted = UniLocalNotification.IsLocalNotificationPermitted();  
 }  
 public void OpenAppSetting()  
 {  
 UniLocalNotification.OpenAppSetting();  
 }  
 public void CancelAll()  
 {  
 UniLocalNotification.CancelAll();  
 }  
 private void Update()  
 {  
 if (Input.GetKeyDown(KeyCode.Escape))  
 {  
 Application.Quit();  
 }  
 }  
 private void OnApplicationQuit()  
 {  
 UniLocalNotification.Register(3, "This is your message", "Game Notification");  
 }  
}

**FOR BUTTON HOME AND ĐA NHIỆM BUTTON**

private void OnApplicationFocus(bool hasFocus)  
{  
 if (hasFocus)  
 UniLocalNotification.Register(3, "You are focus", "Game Notification");  
 else  
 UniLocalNotification.Register(3, "You are not focus", "Game Notification");  
   
}  
private void OnApplicationPause(bool pauseStatus)  
{  
 if (pauseStatus)  
 UniLocalNotification.Register(3, "You are pause", "Game Notification");   
 else  
 UniLocalNotification.Register(3, "You are not pause", "Game Notification");  
}

- Press home button:

* You are not focus
* You are not pause

- Press da nhiem button:

* You are not focus
* You are pause
* You are not pause

**PUSH NOTIFICATION BY USING FIREBASE CLOUD MESSAGING**

<https://www.youtube.com/watch?v=v5uIUEuulm4>

Step 1. Import FirebaseMessaging.unitypackage

Step 2. Create game object, attach script:

public class **FCMNofitication** : MonoBehaviour  
{  
 void **Start**()  
 {  
 FirebaseMessaging.TokenReceived += OnTokenReceived;  
 FirebaseMessaging.MessageReceived += OnMessageReceived;  
 }  
  
 private void OnMessageReceived(object sender, MessageReceivedEventArgs e)  
 {  
 }  
  
 private void OnTokenReceived(object sender, TokenReceivedEventArgs e)  
 {  
 }  
  
 void Update()  
 {  
 }  
}

Step 3. Create message from FCM