**Notification plugin documentation**

**Objective:**

Notification Phonegap application allows any device to register itself on GCM and makes itself valid to receive notifications from server side. This plugin is compatible with PhoneGap 2.8.1.

**Steps:**

**Before you start to look at the functionality of camera plugin make sure that you have following software’s installed.**

* android-sdk
* phonegap-2.8.1
* sencha touch 2.0.1.1.

Apart from above mentioned prerequisites, we also must have a url where we can upload our image file.

**Steps for Notification plugin functionality.**

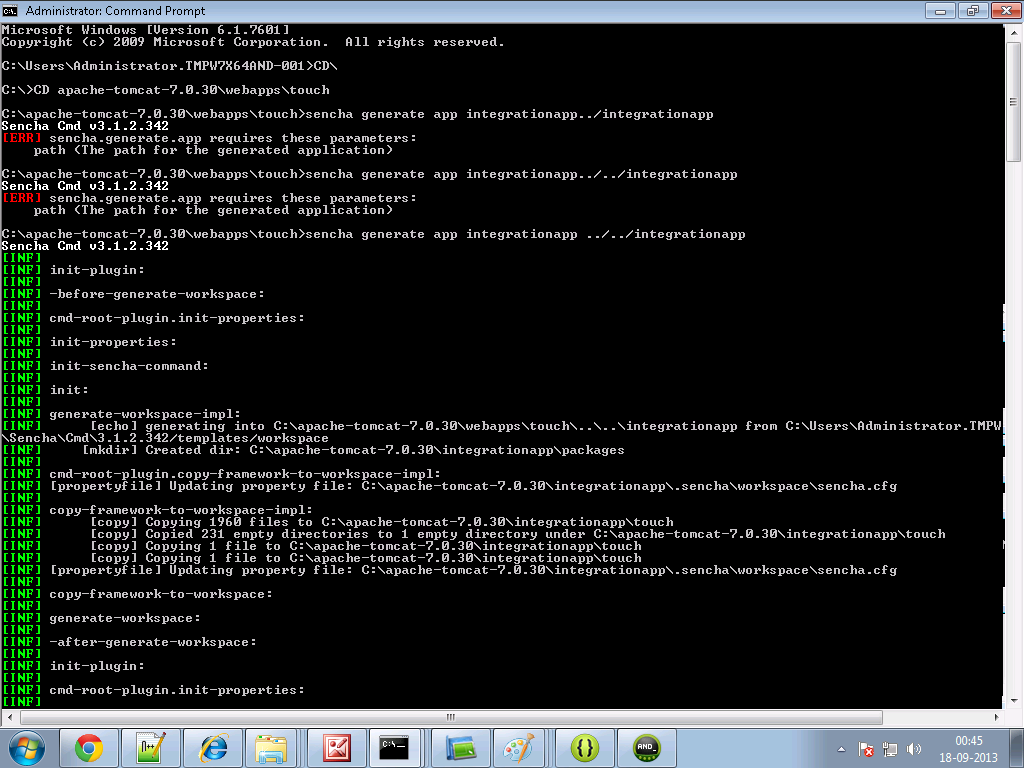
1. **First generate project on GCM note down the server key and project id.**

**To create project please refer to**

[**https://zen.myatos.net/space/in/Digital\_Campus\_India/wiki?pageId=4450953**](https://zen.myatos.net/space/in/Digital_Campus_India/wiki?pageId=4450953)

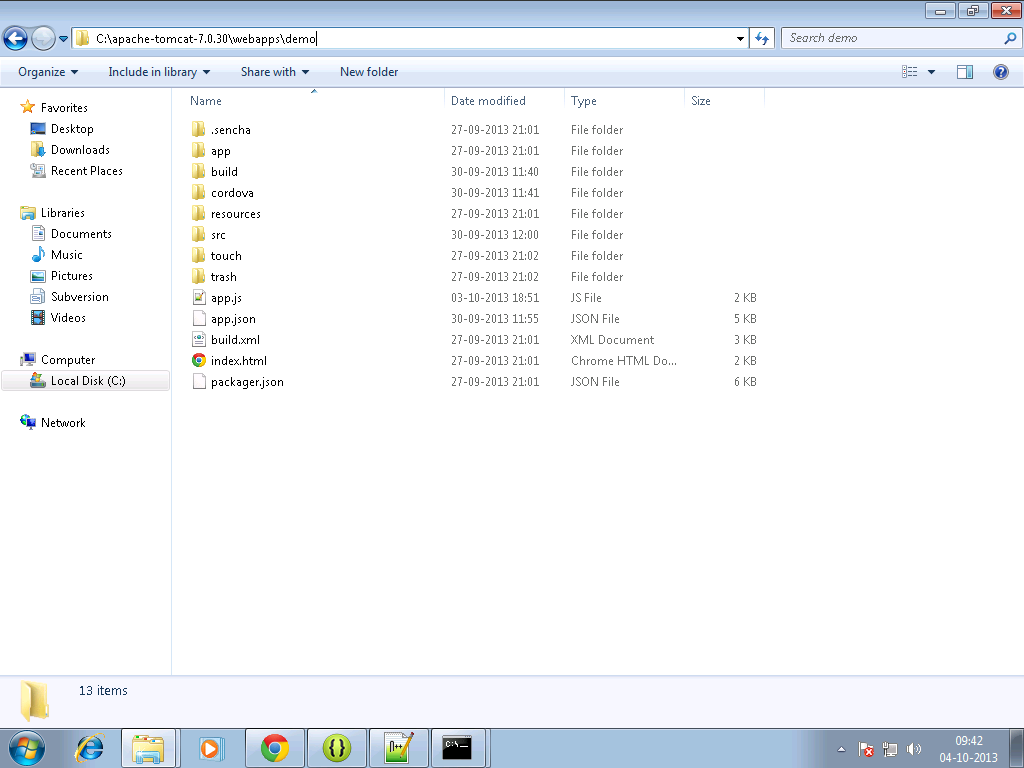
1. **create a new application with the help of following command in cmd:**

**C:\apache-tomcat-7.0.30\webapps\touch>** **sencha generate app <appname> ../<appname>**



1. **Create a folder and name it as cordova in your sencha <appname> directory.**

**As you can see in the screenshot below-**



1. **Copy the PushNotification .js cordova.js ,cordova.js\_android, cordova.js\_ios and cordova.js\_plugins, in them.**
2. **Make an entry in the js tag of app.json file residing in your<appname> directory as follows:**

js": [  
   {  
            "path": "cordova/cordova.js",  
            "update": "delta"  
        },

{

"path": "cordova/PushNotification.js"

},

]

1. **Add the Notification Controller to the controller directory.**

****

registerDevice : function () {

var pushNotification = window.plugins.pushNotification;

if (device.platform == 'android' || device.platform == 'Android') {

pushNotification.register(ICICIDemo.app.getControllerInstances()['ICICIDemo.controller.NotificationController'].recieveGCMRegistrationId,

ICICIDemo.app.getControllerInstances()['ICICIDemo.controller.NotificationController'].errorHandler, {

"senderID" : "**624986650855**",

"ecb" :

}

},

**Also replace the sender id in registerDevice() with your projects product id as given while creating gcm.**

1. **Make Entry of NotificationController in your app.js.**
2. **Write the sencha code to register your device on GCM to receive notification for a particular project.**

ICICIDemo.app.getControllerInstances()['ICICIDemo.controller.NotificationController'].registerDevice();

**You can call this method on whenever your app gets launched or wherever appropriate.**

1. **On registering device on gcm it return a registration id.**
2. **You then send it to your web service which will then use this registration id to send you notifications.**

**You can send the registration id to your web service handleRegistrationId () function in notificationController.**

if (sendToServer) {

Ext.Ajax.request({

url: **DemoApp.app.registerDeviceURL** + "?registrationId=" + registrationId ,

method: 'POST',

headers : {

'Content-Type' : 'application/json',

},

success: function(response){

//alert('sent');

//save the registration id on local

window.localStorage.setItem("registrationId", registrationId);

},

error: function(error){

//alert('Error in sending Registration/Token Id to Server : ' + error);

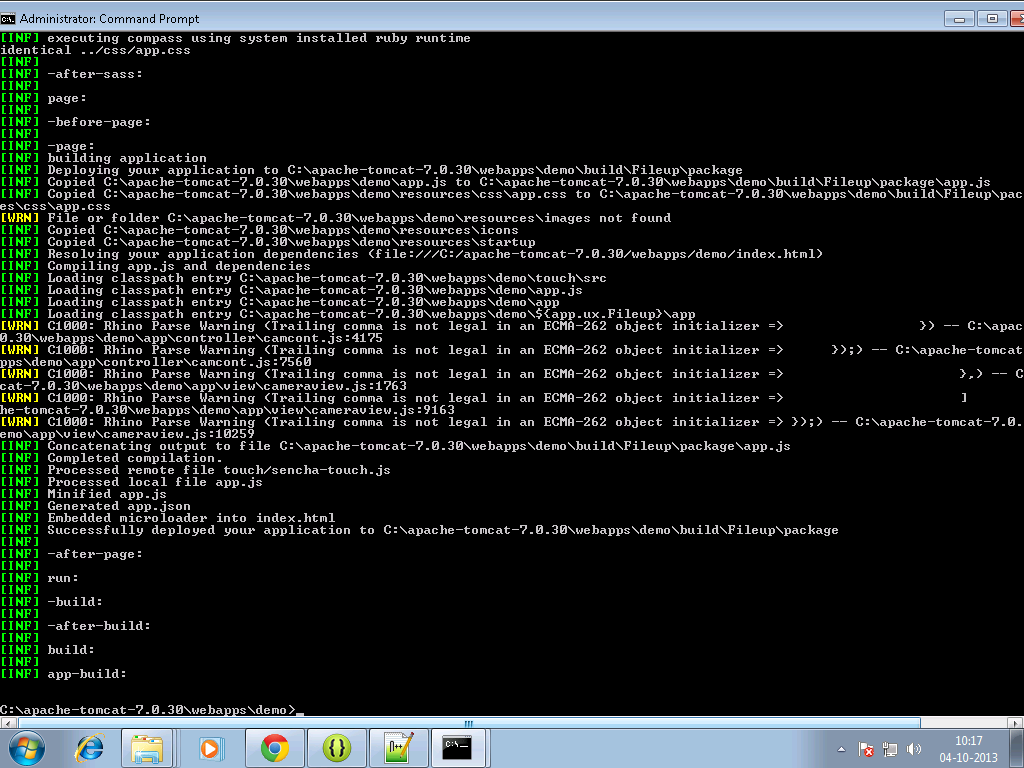
}

});

}

**6.** **In command prompt go to your sencha <appname> directory first, then Build the package of your sencha app by typing the following command :**

**Sencha app build package**

****

**7. Move to your sencha<app> directory, there a build folder will get created.**

**8. Copy files from package folder under <sencha appname>/build/ sencha < appname> to your www folder of android project for example:**

**C:\temp\TempAndroid\assets\www**

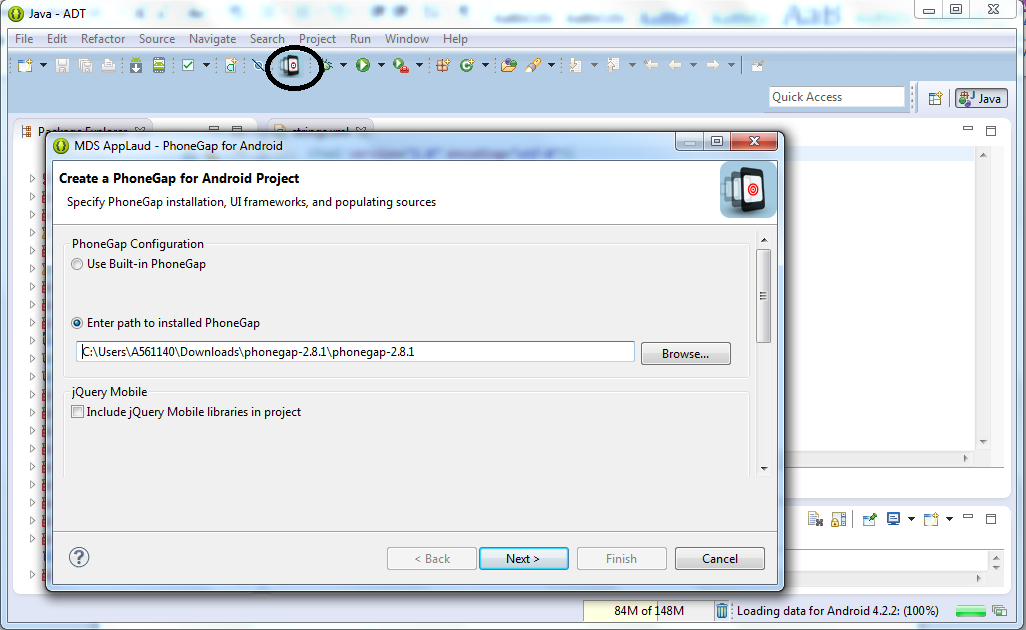
**9. build it - > apk will be in bin folder**

**Part 2: Integrating Sencha with Eclipse code:**

1. Create a phone gap project in the eclipse of the android sdk. For that we must have android sdk installed.

You can download the android sdk (ADT BUNDLE) form the following link:

<http://developer.android.com/sdk/index.html>



1. copy the contents of **src/android/com/** to your project's **src/com/** folder. copy the contents of **libs/** to your**libs/** folder. copy **{android\_sdk\_path}/extras/android/support/v13/android-support-v13.jar** to your **libs/**folder. The final hierarchy will likely look something like this:

{project\_folder}

libs

gcm.jar

android-support-v13.jar

cordova-3.4.0.jar

src

com

plugin

gcm

CordovaGCMBroadcastReceiver.java

GCMIntentService.java

PushHandlerActivity.java

PushPlugin.java

{company\_name}

{intent\_name}

{intent\_name}.java

2) Modify your **AndroidManifest.xml** and add the following lines to your manifest tag:

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.GET\_ACCOUNTS" />

<uses-permission android:name="android.permission.WAKE\_LOCK" />

<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />

<permission android:name="$PACKAGE\_NAME.permission.C2D\_MESSAGE" android:protectionLevel="signature" />

<uses-permission android:name="$PACKAGE\_NAME.permission.C2D\_MESSAGE" />

1. Modify your **AndroidManifest.xml** and add the following **activity**, **receiver** and **service** tags to your**application** section. (See the Sample\_AndroidManifest.xml file in the Example folder.)

<activity android:name="com.plugin.gcm.PushHandlerActivity"/>

<receiver android:name="com.plugin.gcm.CordovaGCMBroadcastReceiver" android:permission="com.google.android.c2dm.permission.SEND" >

<intent-filter>

<action android:name="com.google.android.c2dm.intent.RECEIVE" />

<action android:name="com.google.android.c2dm.intent.REGISTRATION" />

<category android:name="$PACKAGE\_NAME" />

</intent-filter>

</receiver>

<service android:name="com.plugin.gcm.GCMIntentService" />

1. Modify your **res/xml/config.xml** to include the following line in order to tell Cordova to include this plugin and where it can be found: (See the Sample\_config.xml file in the Example folder)

<feature name="PushPlugin">

<param name="android-package" value="com.plugin.gcm.PushPlugin" />

</feature>

1. Add the **PushNotification.js** script to your assets/www folder (or javascripts folder, wherever you want really) and reference it in your main index.html file. This file's usage is described in the **Plugin API** section below.

<script type="text/javascript" charset="utf-8" src="PushNotification.js"></script>