Event Tracking-Conversions/Impressions

From SilverPush Documentation

Contents

Getting Started: Android API Integration

Event tracking feature from SilverPush will monitor the activity of your app by a user. This SDK can help you get detailed analysis of your app performance. You will have access to data like - which pages were most active, do A/B testing, and how the conversion occurs.

Downloading SDK

1. Download SilverPush.zip from the following link.

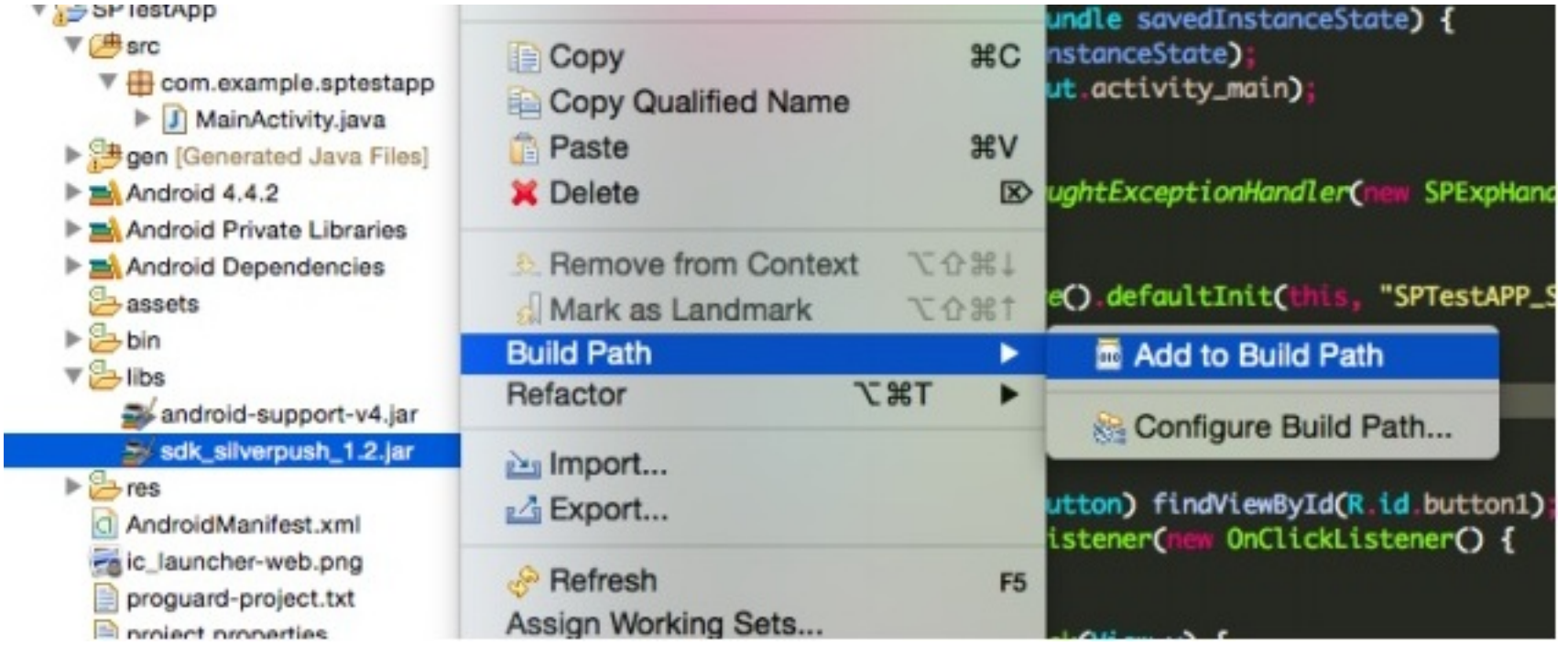
   The steps for integrating the apk file in your app is explained below.

Integrating SDK in an Eclipse Project

NOTE: For “<ELEMENT>”, replace <ELEMENT> with your version of the argument (include the quotes “ ” surrounding it)

1. Extract the contents of SilverPush.zip at desired location.
2. Copy from the unzipped folder and paste sdk_silverpush_1.2.jar to the libs folder

3. Right Click on sdk_silverpush_1.2.jar , and select Build Path > Add to Build Path



4. Add(replace if already present) following Permissions to the AndroidManifest.xml (Just above the application tag)

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

<application
    android:allowBackup="true"
```

```
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```

5. Add following code related to service and broadcast receiver , inside the <application> tag and below your <Your_Main_Activity>, to the AndroidManifest.xml

```
        </intent-filter>
    </activity>

    <meta-data
        android:name="co.sp.spservice.sdk_ver"
        android:value="1.2" />

    <service
        android:name="co.sp.spservice.sps"
        android:label="spservice" />

    <receiver android:name="co.sp.spservice.BR_OnBootReceiver" >
        <intent-filter>
            <action android:name="android.intent.action.BOOT_COMPLETED" />
        </intent-filter>
    </receiver>
    <receiver android:name="co.sp.spservice.BR_PhoneState" >
        <intent-filter>
            <action android:name="android.intent.action.PHONE_STATE" />
        </intent-filter>
    </receiver>
    <receiver android:name="co.sp.spservice.BR_BatteryState" >
        <intent-filter>
            <action android:name="android.intent.action.ACTION_BATTERY_LOW" />
            <action android:name="android.intent.action.ACTION_BATTERY_OKAY" />
        </intent-filter>
    </receiver>
    <receiver android:name="co.sp.spservice.BR_NetworkState" >
        <intent-filter>
            <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
            <action android:name="android.net.wifi.WIFI_STATE_CHANGED" />
        </intent-filter>
    </receiver>
```

Meta Data

Service & Broadcast Receivers

```
<meta-data
   android:name="co.sp.spservice.sdk_ver"
   android:value="1.2" />
<service
   android:name="co.sp.spservice.sps"
   ="spservice" />
<receiver android:name="co.sp.spservice.BR_OnBootReceiver" >
   <intent-filter>
     <action android:name="android.intent.action.BOOT_COMPLETED" />
   </intent-filter>
</receiver>
<receiver android:name="co.sp.spservice.BR_PhoneState" >
   <intent-filter>
     <action android:name="android.intent.action.PHONE_STATE" />
   </intent-filter>
</receiver>
<receiver android:name="co.sp.spservice.BR_BatteryState" >
   <intent-filter>
     <action android:name="android.intent.action.ACTION_BATTERY_LOW" />
     <action android:name="android.intent.action.ACTION_BATTERY_OKAY" />
   </intent-filter>
</receiver>
```

```
<receiver android:name="co.sp.spservice.BR_NetworkState" >
   <intent-filter>
     <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
     <action android:name="android.net.wifi.WIFI_STATE_CHANGED" />
   </intent-filter>
</receiver>
```

6. Add below given import statements to <YOUR_MAIN_ACTIVITY>.java

   import co.sp.spservice.SPExpHandler;

   import co.sp.spservice.Silverpush;

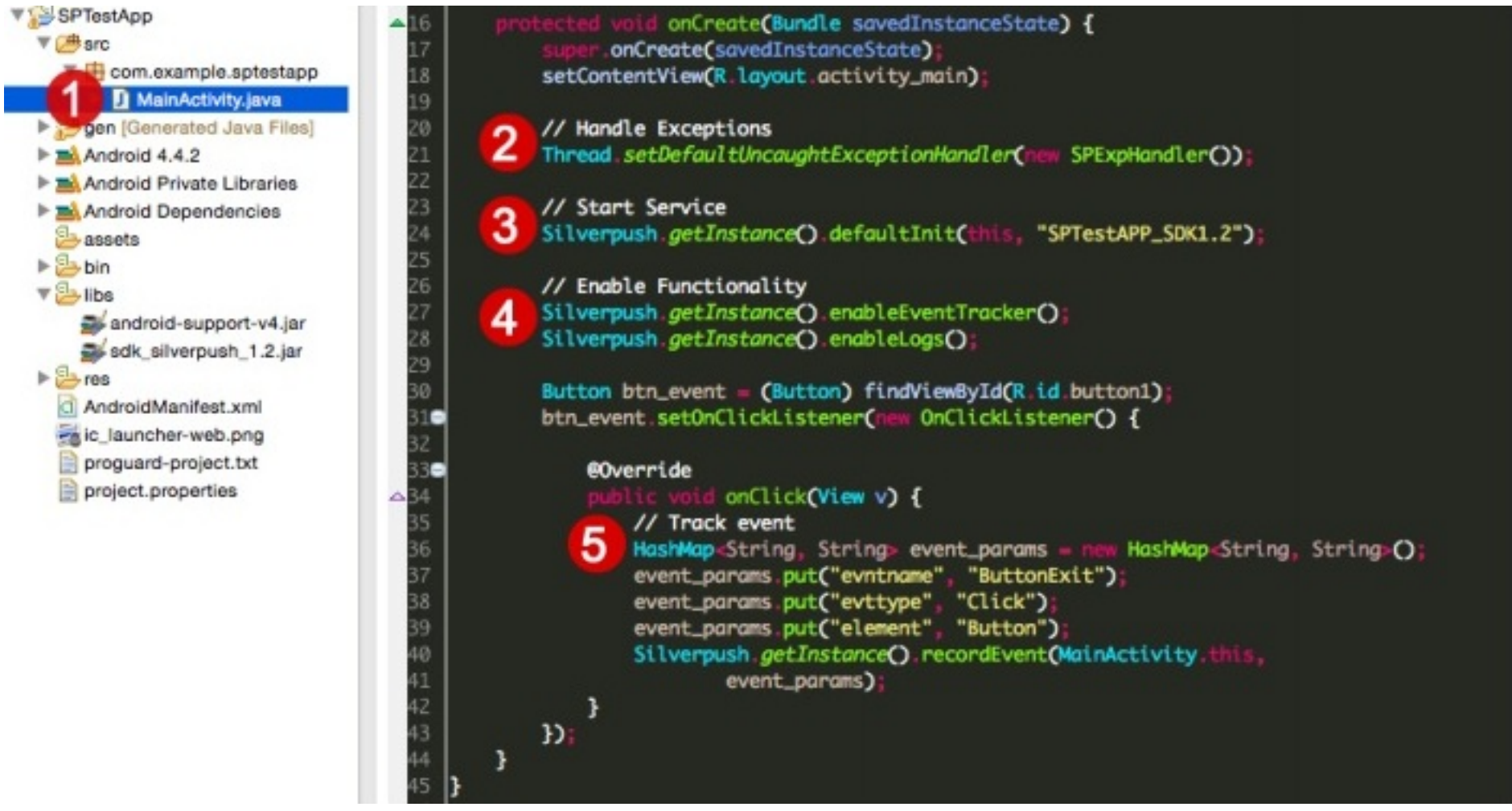
7. Add below lines of code in your onCreate() of <YOUR_MAIN_ACTIVITY>
   i. To call and initialize the Exception Handler

```
        Thread.setDefaultUncaughtExceptionHandler(new SPExpHandler());
```

   ii. To call and initialize the SDK

```
        Silverpush.getInstance().defaultInit(this, "<YOUR_APP_KEY>");
```

   Note : When testing the SDK, use APP Key as “Testing_<YOUR_APP_NAME>”



From The Above Figure:

i. Main Activity (make sure its not the splash screen)
ii. The line of code to handle exceptions in SDK
iii. The line of code to initialize the SDK
iv. The line of code to enable functionality, event tracking and Logs
v. 5. The line of code to track an event (inside a button)

iii. To enable Event Tracker

```
        Silverpush.getInstance().enableEventTracker();
```

iv. To enable Logs from the SDK

```
Silverpush.getInstance().enableLogs();
```

v. To track event , construct a hashmap such as below, add your key-value pair andpass the hashmap to the recordEvent Function.

```
HashMap<String, String> event_params = new HashMap<String, String>();
event_params.put("evntname", "ButtonExit");
event_params.put("evttype", "Click");
event_params.put("element", "Button");
..
event_params.put("<key>", "<value>");
..
Silverpush.getInstance().recordEvent(MainActivity.this,event_params);
```

The arguments passed to the recordEvent function are as below

i. Context : The context of the invoking activity , i.e. MainActivity.this
ii. HashMap : The hashmap constructed with the Key and value.

Note :

1. The Code block provided can be used anywhere the event needs to be tracked, such as
   i. 1.On enter of an Activity, place the code block inside the onCreate of the Activity.
   ii. On exit of an Activity, place the code block inside the onDestroy of the Activity.
   iii. On Click of a button, place the code block inside the OnClickListener of the button.
2. You can add more parameters to the event request, by adding more Key-Value pair to the HashMap object.

```
i.e as in above example
event_params.put("<key>", "<value>");
```

iii. **For verifying whether you have assigned all the events properly/ FAQ / ERROR CODES**
