|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document History** | | | | |
| **Version** | **Date** | **Author** | **Section** | **Changes** |
| 0.1 | 08-07-2015 | Vinayakkumar G Udikeri | All | Initial draft |
| 0.2 | 28-07-2015 | Vinayakkumar G Udikeri | All | Initial draft |
| 0.3 | 13-08-2015 | Vinayakkumar G Udikeri | All | Initial draft |
| 0.4 | 21-10-2015 | Vinayakkumar G Udikeri | All | HSDP incorporation and configuration changes |
| 0.5 | 04-12-2015 | Vinayakkumar G Udikeri | All | Dependant libs update ,Flow configuration,Janrain Initialization,Tagging,Progaurd,Callbacks |
| 0.6 | 04-02-2016 | Vinayakkumar G Udikeri |  | Dynamic Configuration |
| 0.7 | 04-03-2016 | Vinayakkumar G Udikeri |  | Artifactory Support and More UI Configuration with Color |
| 0.8 | 06-04-2016 | Vinayakkumar G Udikeri |  | COPPA changes |
| 0.9 | 19-04-2016 | Vinayakkumar G Udikeri |  | Drawable support ,Complete COPPA module ,Manifest permissions |
| 1.0 | 09-05-2016 | Vinayakkumar G Udikeri |  | LocaleMatch 2.0.0,Tagging 3.0.0,Additional color attributes,COPPA localization,Source path change |

Registration Android Integration

|  |  |
| --- | --- |
| **Author** | Vinayakkumar G Udikeri |
| **Approved by** |  |
| **Email Id** | vinayak.kumar.udikeri@philips.com |

[1. Introduction 4](#_Toc450698232)

[2. Prerequisites 5](#_Toc450698233)

[3. Integration 6](#_Toc450698234)

[3.1 Referring from Artifactory 6](#_Toc450698235)

[3.1.1 Changes at top level build.gradle 6](#_Toc450698236)

[3.1.2 Changes at app level build.gradle 7](#_Toc450698237)

[3.2 Referring from Local libs 9](#_Toc450698238)

[3.2.1 Dependencies 9](#_Toc450698239)

[3.3 Integration jar/aar files 9](#_Toc450698240)

[3.3.1 Select Project mode view for Project 9](#_Toc450698241)

[3.3.2 Paste .jar/.aar files to libs directory from reference app. 10](#_Toc450698242)

[3.3.3 Select build.gradle file 11](#_Toc450698243)

[3.3.4 Refer Registration dependencies along with .aar files in build.gradle as mentioned below 11](#_Toc450698244)

[3.3.5 Clean project 12](#_Toc450698245)

[3.3.6 Paste Registration related assets in assets folder from reference app 12](#_Toc450698246)

[3.3.7 Repeat step 3.2.6 13](#_Toc450698247)

[3.3.8 Android Manifest Changes 13](#_Toc450698248)

[4. Configuration 14](#_Toc450698249)

[4.1 UI Configurations [registration\_config.xml] 14](#_Toc450698250)

[4.1.1 Copy registration\_config.xml from reference app project and paste into project’s values folder and alter as per vertical need 17](#_Toc450698251)

[4.2 Flow Configurations 23](#_Toc450698252)

[4.2.1 Standard 23](#_Toc450698253)

[4.2.2 HSDP 23](#_Toc450698254)

[4.3 Registration Configuration 23](#_Toc450698255)

[4.3.1 Static Configuration 23](#_Toc450698256)

[4.3.2 Dynamic Configuration 27](#_Toc450698257)

[4.3.3 Hybrid Configuration 34](#_Toc450698258)

[5. Janrain Initialization 35](#_Toc450698259)

[5.1 UI Priority function 35](#_Toc450698260)

[5.1.1 Create account on Top 35](#_Toc450698261)

[5.1.2 Sign In On top 36](#_Toc450698262)

[5.2 Configuration initialization 37](#_Toc450698263)

[5.3 Custom Language in Registartion UI 38](#_Toc450698264)

[6. Launching Registration component 39](#_Toc450698265)

[6.4 Activity 39](#_Toc450698266)

[6.4.1 Supported orientation 39](#_Toc450698267)

[6.4.2 Launching default mode 39](#_Toc450698268)

[6.4.3 Launching default mode with fixed orientation 39](#_Toc450698269)

[6.4.4 Launching with context of Account settings 39](#_Toc450698270)

[6.5 Fragment 40](#_Toc450698271)

[6.5.1 Launching as fragment 40](#_Toc450698272)

[6.5.2 Backstack handling 41](#_Toc450698273)

[6.5.3 Updating Title 41](#_Toc450698274)

[7. Call-back’s 42](#_Toc450698275)

[8. APP tagging 43](#_Toc450698276)

[9. HSDP specific API’s 45](#_Toc450698277)

[9.1 HSDP UUID 45](#_Toc450698278)

[9.2 HSDP Access Token 45](#_Toc450698279)

[10. Library version’s 46](#_Toc450698280)

[10.1 Janrain Library 46](#_Toc450698281)

[10.2 Locale Match Library 46](#_Toc450698282)

[10.3 Registration Library 46](#_Toc450698283)

[11. Supporting apps with Over 65K Methods 47](#_Toc450698284)

[12. Progaurd rules for Registration API component 48](#_Toc450698285)

[12.1 General 48](#_Toc450698286)

[12.2 Jump 48](#_Toc450698287)

[12.3 Localematch 49](#_Toc450698288)

[12.4 Registration API 49](#_Toc450698289)

[12.5 Network APIs 49](#_Toc450698290)

[12.6 HSDP 49](#_Toc450698291)

[12.7 Other platform specific 49](#_Toc450698292)

[12.8 Hocky SDK 50](#_Toc450698293)

[12.9 Complete for configuration for Registration API 50](#_Toc450698294)

[13. Manifest permission’s 53](#_Toc450698295)

[14. Demo apps 54](#_Toc450698296)

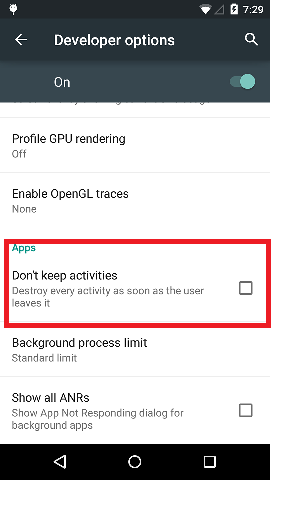
# Introduction

This document provides an overview of integration Registration Android UI feature in Android Mobile applications.

Source Path: <https://atlas.natlab.research.philips.com/stash/scm/ur/user_registration_android.git>

# Prerequisites

* + 1. Vertical project is configured for Android Studio
    2. Setting->Developer Options->Don’t Keep Activities should be unchecked.[If developer mode is on]
    3. Android API version should on 10[Gingerbread]



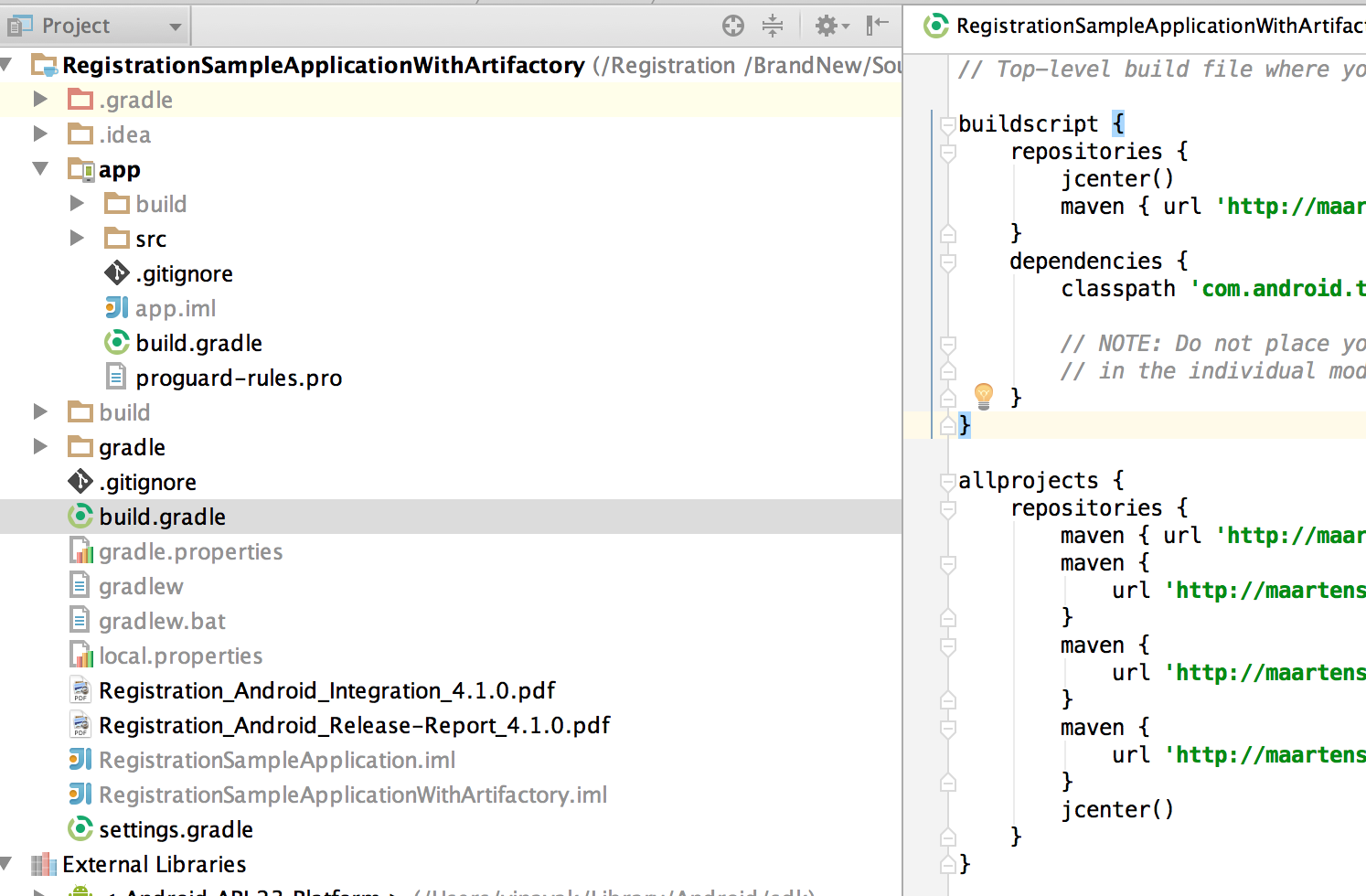
# Integration

Integration can be done in following ways

## Referring from Artifactory

Steps to consume User registration from Artifactory following things needs to be done from App side

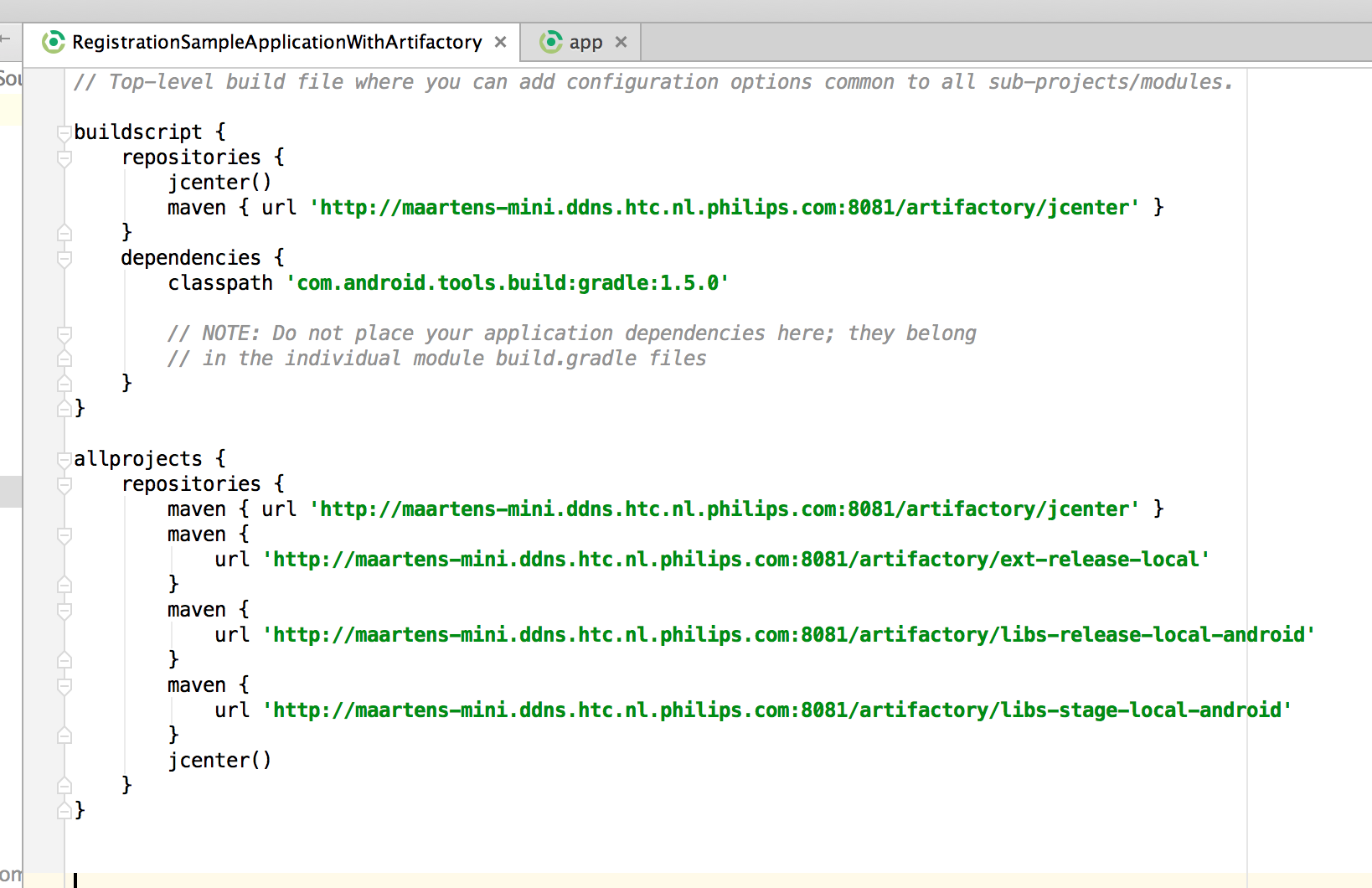
### Changes at top level build.gradle



Need to mention the maven repo links and dependent url link in top build.gradle

Root build.gardle must contain following dependent links.

***// Top-level build file where you can add configuration options common to all sub-projects/modules.  
  
buildscript {  
 repositories {  
 jcenter()  
 maven { url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/jcenter' }  
 }  
 dependencies {  
 classpath 'com.android.tools.build:gradle:1.5.0'  
  
 // NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files  
 }  
}  
  
allprojects {  
 repositories {  
 maven { url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/jcenter' }  
 maven {  
 url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/ext-release-local'  
 }  
 maven {  
 url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/libs-release-local-android'  
 }  
 maven {  
 url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/libs-stage-local-android'  
 }  
 jcenter()  
 }  
}***



### Changes at app level build.gradle

Need to mention artifactory dependences for registration in dependencies {} block

It can be mentioned in any one of the following ways

1. compile(group:'com.philips.cdp', name:'registrationApi', version:'5.0.0')
2. compile(group: 'com.philips.cdp', name: 'registrationApi', version: "5.0.0", ext: 'aar') {  
    exclude group: 'com.android.support'  
    transitive = true  
    }
3. compile 'com.philips.cdp:registrationApi:5.0.0'
4. ***apply plugin: 'com.android.application'  
     
     
   android {  
    compileSdkVersion 23  
    buildToolsVersion "23.0.2"  
     
    packagingOptions {  
    exclude 'META-INF/DEPENDENCIES.txt'  
    exclude 'META-INF/LICENSE.txt'  
    exclude 'META-INF/NOTICE.txt'  
    exclude 'META-INF/NOTICE'  
    exclude 'META-INF/LICENSE'  
    exclude 'META-INF/DEPENDENCIES'  
    exclude 'META-INF/notice.txt'  
    exclude 'META-INF/license.txt'  
    exclude 'META-INF/dependencies.txt'  
    exclude 'META-INF/LGPL2.1'  
    }  
     
    defaultConfig {  
    applicationId "com.philips.cdp.registration.demo.app"  
    minSdkVersion 15  
    targetSdkVersion 23  
    versionCode 1  
    versionName "1.0"  
    }  
    buildTypes {  
    release {  
    minifyEnabled false  
    proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
    }  
    }  
   }  
     
   dependencies {  
    compile fileTree(dir: 'libs', include: ['\*.jar'])  
    compile 'com.android.support:appcompat-v7:23+'  
     
    compile(group: 'com.philips.cdp', name: 'registrationApi', version: "5.0.0", ext: 'aar') {  
    exclude group: 'com.android.support'  
    transitive = true  
    }  
   }***

***Note : Demo app is available for reference along with this package***

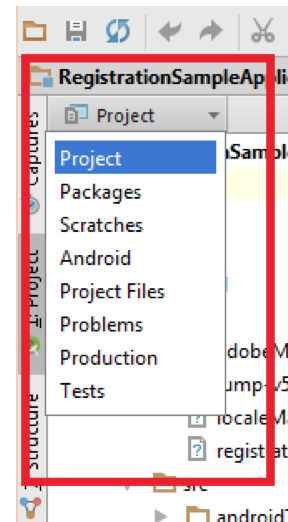
## Referring from Local libs

### Dependencies

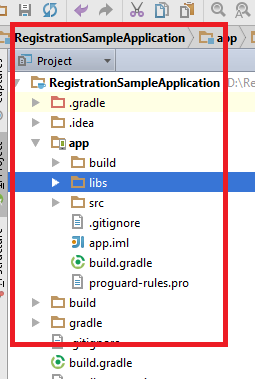
* jump.aar[v.6.0.0]
* localeMatch.aar[v.2.0.0]
* registrationApi.aar[v.5.0.0]
* tagging.aar[v.3.0.0]
* hsdp.aar[v.1.0.3]
* ntpClient.aar[v.1.0.2]
* adobeMobileLibrary.jar[v.4.5.1]
* org.apache.http.legacy.jar
* jackson-annotations.jar[v.2.6.0]
* jackson-core.jar[v.2.6.1]
* jackson-databind.jar[v.2.6.1]
* okhttp-2.5.0.jar
* okhttp-apache-2.5.0.jar
* okio-1.6.0.jar

## Integration jar/aar files

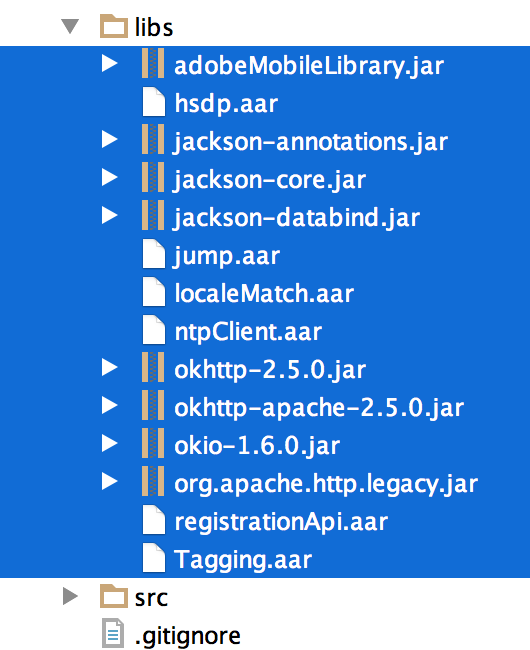
### Select Project mode view for Project



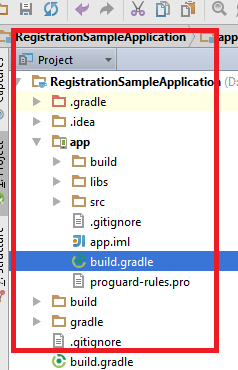
Crate lib folder if not exists.



### Paste .jar/.aar files to libs directory from reference app.



### Select build.gradle file



### Refer Registration dependencies along with .aar files in build.gradle as mentioned below

*Note: Since Registration Android was built with* ***'com.android.support:appcompat-v7:23+. version so vertical app support libs should be greater or equal to mentioned versions***

***android {  
 packagingOptions {  
 exclude 'META-INF/DEPENDENCIES.txt'  
 exclude 'META-INF/LICENSE.txt'  
 exclude 'META-INF/NOTICE.txt'  
 exclude 'META-INF/NOTICE'  
 exclude 'META-INF/LICENSE'  
 exclude 'META-INF/DEPENDENCIES'  
 exclude 'META-INF/notice.txt'  
 exclude 'META-INF/license.txt'  
 exclude 'META-INF/dependencies.txt'  
 exclude 'META-INF/LGPL2.1'  
 }  
}***

***repositories {  
 flatDir {  
 dirs 'libs'  
 }  
}  
dependencies {***

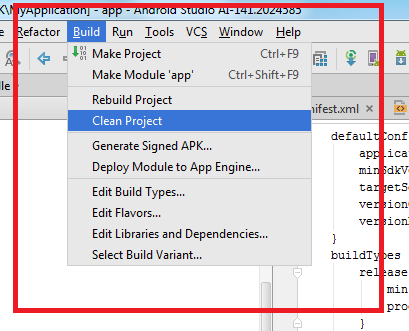
***compile fileTree(dir: 'libs', include: ['\*.jar'])  
compile 'com.android.support:appcompat-v7:23+'  
  
compile files('libs/jackson-annotations.jar')  
compile files('libs/jackson-core.jar')  
compile files('libs/jackson-databind.jar')  
compile files('libs/adobeMobileLibrary.jar')  
compile files('libs/org.apache.http.legacy.jar')  
compile files('libs/okhttp-2.5.0.jar')  
compile files('libs/okhttp-apache-2.5.0.jar')  
compile files('libs/okio-1.6.0.jar')***

***compile(name: 'localeMatch', ext: 'aar')  
compile(name: 'jump', ext: 'aar')  
compile(name: 'registrationApi', ext: 'aar')  
compile(name: 'tagging', ext: 'aar')  
compile(name: 'ntpClient', ext: 'aar')  
compile(name: 'hsdp', ext: 'aar')***

***}***

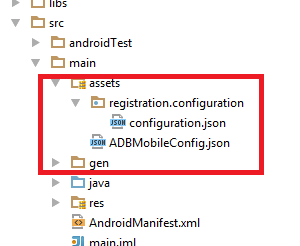
### Clean project

Build->Clean Project



### Paste Registration related assets in assets folder from reference app

**Note : Don’t change or rename folder structure**



### Repeat step 3.2.6

### Android Manifest Changes

To support Registration Android following changes needs to be done mandatorily

* + 1. Needs to mention additional xmlns

**xmlns:tools=**[**http://schemas.android.com/tools**](http://schemas.android.com/tools)

* + 1. Project should have one application class and should be defined in manifest.

Ex :**android:name=".RegistrationApplication"**

* + 1. Asset merging preference

**tools:replace="android:icon,android:theme"**

***<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 package="com.philips.cdp.registration.sample">  
  
 <application  
 android:name="com.philips.cdp.registration.sample.RegistrationApplication"  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:theme="@style/AppTheme"  
 tools:replace="android:icon,,android:label,android:theme">  
 <activity  
 android:name="com.philips.cdp.registration.sample.RegistrationSampleActivity"  
 android:label="@string/app\_name">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>***

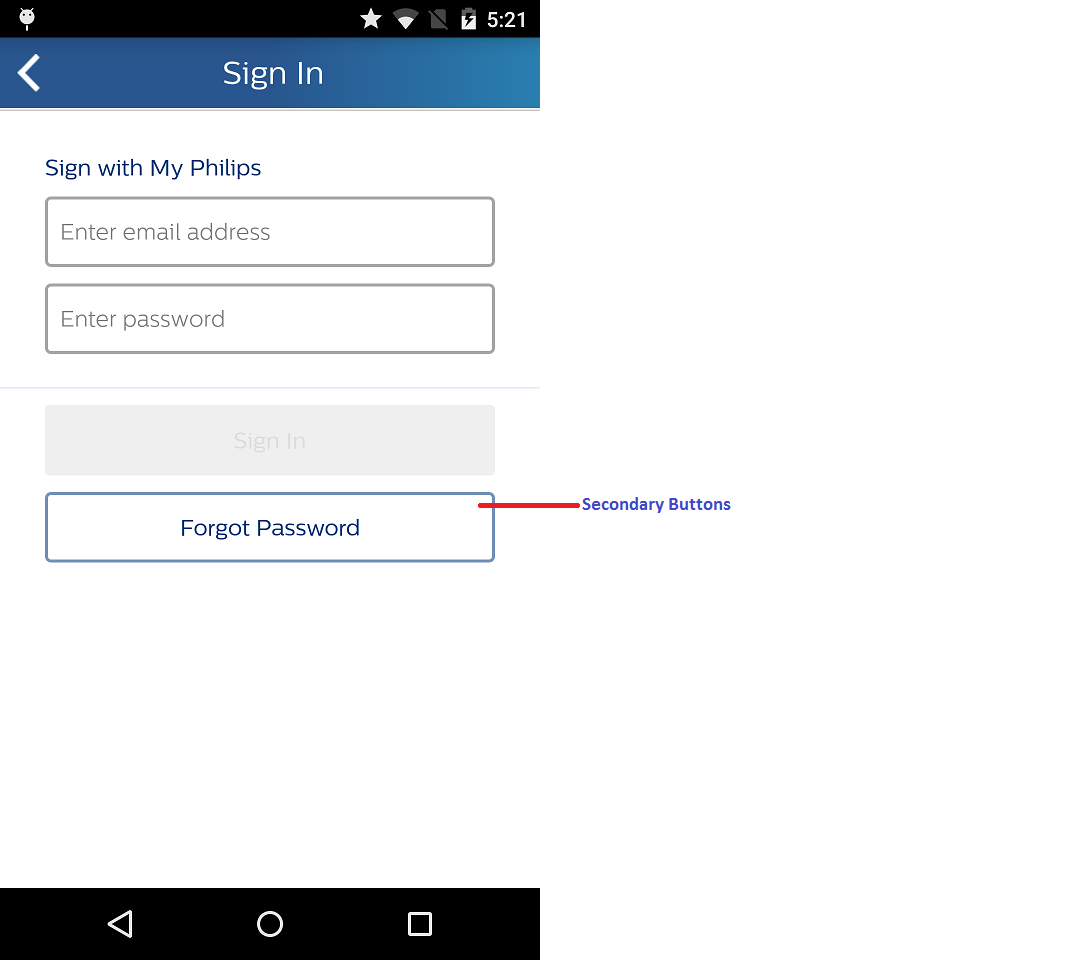
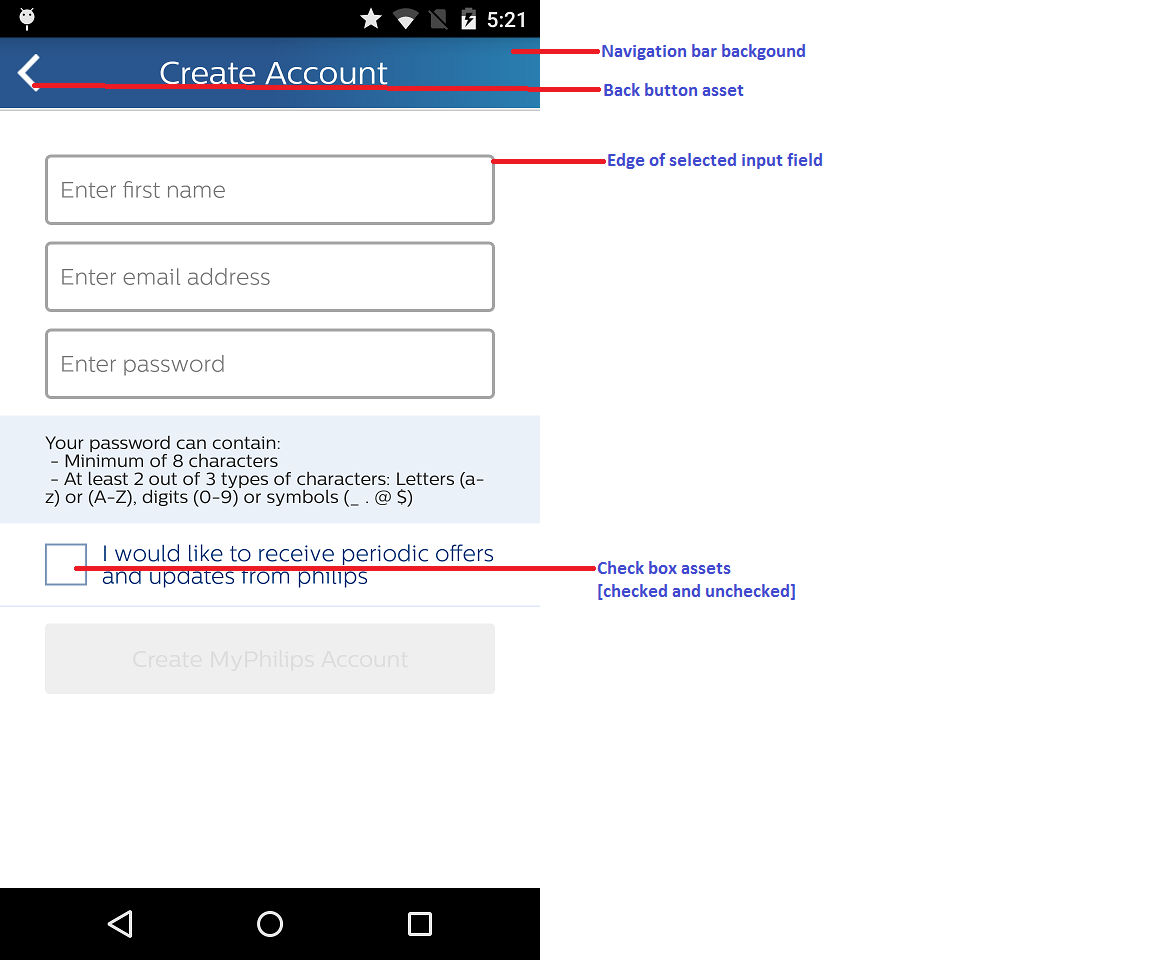
# Configuration

Registration Android can be configurable in two types UI and Flow configurations

## UI Configurations [registration\_config.xml]

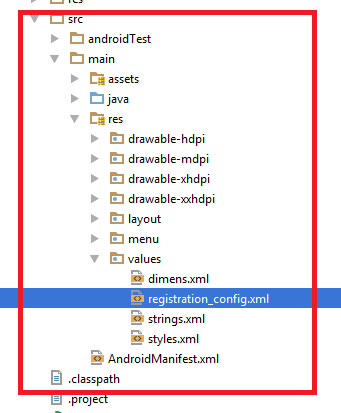
**Configurable elements**

****



UI Theming ;for altering UI look and feel as per vertical specific follow bellow steps

### Copy registration\_config.xml from reference app project and paste into project’s values folder and alter as per vertical need



Color codes need to be defined as follows, if not mentioned default configuration are picked

#### Primary Button

***Vertical specific configuration:*** *<!-- All primary buttons -->  
<!-- Primary button enabled color -->*<**color name="reg\_btn\_bg\_enabled\_color"**>#013378</**color**>  
*<!-- Primary button disabled color -->*<**color name="reg\_btn\_bg\_disabled\_color"**>#EFEFEF</**color**>  
*<!-- Primary button pressed color -->*<**color name="reg\_btn\_pressed\_color"**>#0F204B</**color**>

***Default configuration:*** *<!-- All primary buttons -->  
<!-- Primary button enabled color -->  
<color name="reg\_btn\_bg\_enabled\_color">#013378</color>  
<!-- Primary button disabled color -->  
<color name="reg\_btn\_bg\_disabled\_color">#EFEFEF</color>  
<!-- Primary button pressed color -->  
<color name="reg\_btn\_pressed\_color">#0F204B</color>*

#### Edge of input field’s

***Vertical specific configuration:*** *<!-- Edge of selected input field -->  
<!-- Edit Text box border highlight color -->  
<color name="reg\_et\_highlight\_color">#28558D</color>*

***Default configuration:***

*<color name="reg\_et\_highlight\_color">#28558D</color>*

#### Secondary button color

***Vertical specific configuration:*** *<!-- Secondary buttons (forgot password) -->  
<!-- Forgot password button -->  
<!-- Forgot password buton border color -->  
<color name="reg\_btn\_forgot\_border">#7793B9</color>  
<!-- Forgot password highlight border color -->  
<color name="reg\_highlight\_forgot\_bg">#7793B9</color>*

***Default configuration:***

*<!-- Secondary buttons (forgot password) -->  
<!-- Forgot password button -->  
<!-- Forgot password buton border color -->  
<color name="reg\_btn\_forgot\_border">#7793B9</color>  
<!-- Forgot password highlight border color -->  
<color name="reg\_highlight\_forgot\_bg">#7793B9</color>*

#### Attention block color

***Vertical specific configuration:***

*<!-- Attention block color -->  
<!-- Description text background color -->  
<color name="reg\_text\_desc\_bg">#EBF1F9</color>*

***Default configuration:***

*<!-- Description text background color -->  
<color name="reg\_text\_desc\_bg">#EBF1F9</color>*

#### Link color

***Vertical specific configuration:***

*<!-- Link color -->  
<color name="reg\_hyperlink\_highlight\_color">#1470AA</color>*

***Default configuration:***

*<color name="reg\_hyperlink\_highlight\_color">#1470AA</color>*

#### Navigation bar color

***Vertical specific configuration:***

*<!-- Navigation bar background -->  
<!-- Strating color of Header:background start gradient color -->  
<color name="reg\_header\_bg\_start\_color">#28558D</color>  
<!-- Ending color of Header:background end gradient color -->  
<color name="reg\_header\_bg\_end\_color">#287EB0</color>*

***Default configuration:***

*<!-- Navigation bar background -->  
<!-- Strating color of Header:background start gradient color -->  
<color name="reg\_header\_bg\_start\_color">#28558D</color>  
<!-- Ending color of Header:background end gradient color -->  
<color name="reg\_header\_bg\_end\_color">#287EB0</color>*

#### Navigation bar font color

***Vertical specific configuration:***

*<!-- Navigation bar font color -->  
<!-- Header title text color -->  
<color name="reg\_header\_text\_color">#FFFFFF</color>*

***Default configuration:***

*<!-- Header title text color -->  
<color name="reg\_header\_text\_color">#FFFFFF</color>*

#### Back button asset

***Vertical specific configuration:***

*<!—Back arrow color -->  
<color name="reg\_back\_arrow\_color">#FFFFFF</color>*

***Default configuration:***

*<!—Back arrow color -->  
<color name="reg\_back\_arrow\_color">#FFFFFF</color>*

#### Font color explanatory text

***Vertical specific configuration:***

*<!-- Font color explanatory texts -->  
<!-- Description text color -->  
<color name="reg\_text\_desc\_color">#252525</color>*

***Default configuration:***

*<!-- Font color explanatory texts -->  
<!-- Description text color -->  
<color name="reg\_text\_desc\_color">#252525</color>*

#### Layout background

Background is set by the drawable ,Keep proper aspect ration drawable in respective folders for fit properly in UI.Even color also can be mentioned as drawable.

***Vertical specific configuration:***

*<!—Background as color -->  
<drawable name="reg\_layout\_bg">#FFFFFF</drawable>*

*<!—Background as drawable -->  
<drawable name="reg\_layout\_bg">@drawable/ic\_launcher</drawable>*

***Default configuration:***

*<!-- Background -->  
<drawable name="reg\_layout\_bg">#FFFFFF</drawable >*

#### Primary text color

***Vertical specific configuration:***

*<!-- Primary text color -->  
<color name="reg\_text\_heading\_one\_color">#003377</color>*

***Default configuration:***

*<!-- Primary text color -->  
<color name="reg\_text\_heading\_one\_color">#003377</color>*

#### Warning color

***Vertical specific configuration:***

*<!-- Warning color -->  
<color name="reg\_error\_box\_color">#E88200</color>  
<color name="reg\_err\_msg\_color">#E88200</color>*

***Default configuration:***

*<!-- Warning color -->  
<color name="reg\_error\_box\_color">#E88200</color>  
<color name="reg\_err\_msg\_color">#E88200</color>*

#### Social button color

***Vertical specific configuration:***

*<!-- Social button enabled color -->  
<color name="reg\_btn\_social\_bg\_enabled\_color">#0E66A1</color>  
<!-- Social button pressed color -->  
<color name="reg\_btn\_social\_highlight\_color">#033D78</color>*

***Default configuration:***

*<!-- Social button enabled color -->  
<color name="reg\_btn\_social\_bg\_enabled\_color">#0E66A1</color>  
<!-- Social button pressed color -->  
<color name="reg\_btn\_social\_highlight\_color">#033D78</color>*

#### Check box



***Vertical specific configuration:***

*<!-- Check box border color -->  
<color name="reg\_check\_box\_border\_color">#C4150C</color>  
  
<!-- Check box tick color -->  
<color name="reg\_check\_box\_tick\_color">#C4150C</color>*

***Default configuration:***

*<!-- Check box border color -->  
<color name="reg\_check\_box\_border\_color">#C4150C</color>  
  
<!-- Check box tick color -->  
<color name="reg\_check\_box\_tick\_color">#C4150C</color>*

#### Error alert



***Vertical specific configuration:***

*<!-- Check error symbol color -->  
<color name="reg\_error\_symbol\_color">#E88200</color>  
  
<!-- Check error exclamation symbol color -->  
<color name="reg\_exclamation\_color">#FFFFFF</color>*

***Default configuration:***

*<!-- Check error symbol color -->  
<color name="reg\_error\_symbol\_color">#E88200</color>  
  
<!-- Check error exclamation symbol color -->  
<color name="reg\_exclamation\_color">#FFFFFF</color>*

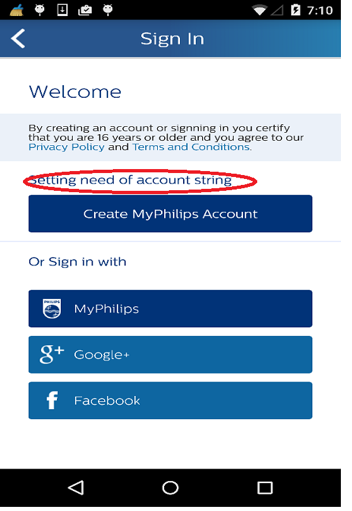
#### Vertical specific texts

##### Setting need of account string

Vertical apps may or may not provide why user account is needed.

If need to provide then vertical apps needs to add following string in the respective localized ***string.xml*** file which will be reflected in in following screen

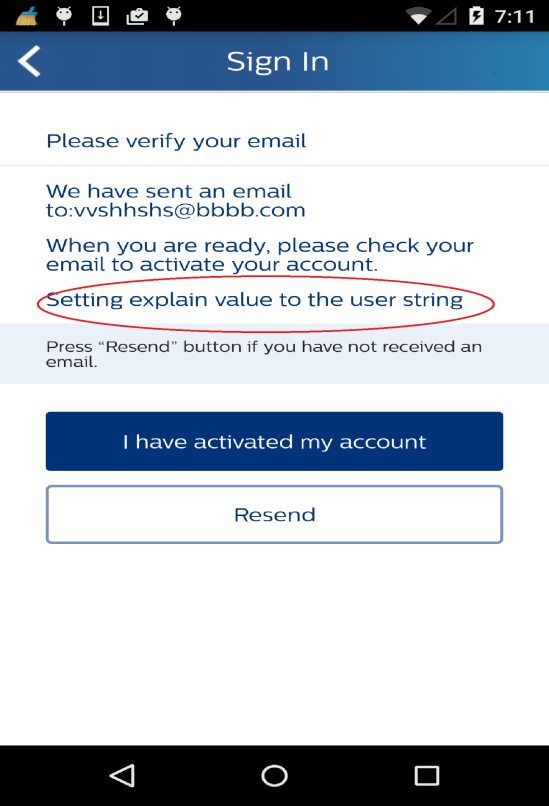
***<string name="Welcome\_needAccountto\_lbltxt">Setting need of account string </string>***

****

##### Setting explain value to the user string

Repeat 4.1.1.16.1 step but use string key as bellow mentioned

***<string me="Activation\_explain\_value\_to\_user"></string>***



#### Dialog background

*Vertical specific configuration:*

*<!--Dailog background-->  
<drawable name="reg\_dailog\_bg">#FFFFFF</drawable>*

*Default configuration:*

*<!--Dailog background-->  
<drawable name="reg\_dailog\_bg">#FFFFFF</drawable>*

## Flow Configurations

### Standard

No additional parameter required. Framework will take care of initializing

it in Standard flow. Depending on provided configuration.json.

### HSDP

HSDP configuration is section 4.2.1+ Dynamic configuration for HSDP part mentioned in Dynamic configuration . If configuration json contains then Runtime exception is thrown.

## Registration Configuration

Registration can be configured in static or dynamic or hybrid way.

### Static Configuration

In Static configuration Vertical Apps must define one ***configuration.json*** file as mentioned below. Once added then can not be altered by code if any changes need to be done then should go with the new build it self.

The ***configuration.json*** should be added to the project and json need to update depend on the project.

* + JanrainConfiguration
  + PILConfiguration
  + Flow
  + SigninProviders

Example RegistrationConfiguration.json:

******

#### JanRainConfiguration (Mandatory)

#### JanRainConfiguration is required to be able to connect to janrain services. It requires following fields:

**RegistrationClientID:** Assign mandatory client id for each environment.

For e.g. “Development”: “xxxxxxxxxx”

Please contact **Digital Services <digital.services@philips.com>** for Client ID for janrain and Microsite ID

***Note :***

***Its highly recommended that app should use STAGING environment mandatorily during development, and STAGING webpages connected to the Evaluation data stores.***

#### PILConfiguration (Mandatory)

PIL configuration requires following three elements:



Note: Please contact **Digital Services <**<mailto:digital.services@philips.com>**>** for Client ID for janrain and Microsite ID

#### Flow (Optional)



#### SigninProviders

Add the entire providers array to default, if you need any country specify sign in providers. Add the list of providers with country code, please refer to the example for more details***.[Note My Philips is always shown so need to configure other providers only]***

Signin Providers supported by registration currently:

facebook

googleplus

***Ex configuration.json for COPPA flow***

***{***

***"JanRainConfiguration": {***

***"RegistrationClientID": {***

***"Development": "xxxxxxxxxxxxxxxxxxx",***

***"Evaluation": "xxxxxxxxxxxxxxxxx",***

***"Production": "xxxxxxxxxxxxxxxxx"***

***}***

***},***

***"PILConfiguration": {***

***"MicrositeID": xxxxxxxxx,***

***"RegistrationEnvironment": " Evaluation",***

***"CampaignID": "xxxxxxxxxxxxxxxx"***

***},***

***"Flow":{***

***"EmailVerificationRequired" : “true”***

***},***

***"SigninProviders": {***

***"NL": [***

***"twitter",***

***"facebook",***

***"googleplus"***

***],***

***"default": [***

***"facebook",***

***"googleplus"***

***]***

***}***

***}***

***Ex configuration.json for Standard flow***

***{***

***"JanRainConfiguration": {***

***"RegistrationClientID": {***

***"Development": "xxxxxxxxxxxxxxxxxxx",***

***"Evaluation": "xxxxxxxxxxxxxxxxx",***

***"Production": "xxxxxxxxxxxxxxxxx"***

***}***

***},***

***"PILConfiguration": {***

***"MicrositeID": xxxxxxxxx,***

***"RegistrationEnvironment": " Evaluation",***

***},***

***"Flow":{***

***"EmailVerificationRequired" : “true”***

***},***

***"SigninProviders": {***

***"NL": [***

***"twitter",***

***"facebook",***

***"googleplus"***

***],***

***"default": [***

***"facebook",***

***"googleplus"***

***]***

***}***

***}***

### Dynamic Configuration

It is possible to change or define entire Registration configuration in code dynamically by following APIs . Which are same as static configuration.

This has been enabled with the RegistrationDynamicConfiguration class

Note : Dynamic configuration are take as priority over the static

#### JanRainConfiguration

For replacing or adding Janarin Configuration ids are quite simple now it can be done in following ways . Registration Framework will take care of replacing and adding ids .

API :

Present in **RegistrationDynamicConfiguration** class.

***/\*\*  
 \* Set Janrain Configuration  
 \*  
 \* @param janRainConfiguration  
 \*/  
public void setJanRainConfiguration(JanRainConfiguration janRainConfiguration) {}***

***/\*\*  
 \* Get Janrain Configuration  
 \*  
 \* @return RegistrationDynamicConfiguration  
 \*/  
public JanRainConfiguration getJanRainConfiguration() {}***

***/\*\*  
 \* Set Janrain client Ids  
 \*  
 \* @param clientIds Object of RegistrationClientId  
 \*/  
public void setClientIds(RegistrationClientId clientIds) {}***

Present in **RegistrationClientId** class

***/\*\*  
 \* Set production id  
 \*  
 \* @return String  
 \*/  
public void setProductionId(String productionId) {}  
  
  
  
/\*\*  
 \* Set evaluation id  
 \*  
 \* @return String  
 \*/  
public void setEvaluationId(String evaluationId) {}  
  
  
  
/\*\*  
 \* Set development id  
 \*  
 \* @return String  
 \*/  
public void setDevelopmentId(String developmentId) {}  
  
  
  
/\*\*  
 \* Set testing id  
 \*  
 \* @return String  
 \*/  
public void setTestingId(String testingId) {}  
  
  
  
/\*\*  
 \* Set stagging id  
 \*  
 \* @return String  
 \*/  
public void setStagingId(String stagingId) {}***

Example :

***RegistrationClientId registrationClientId = new RegistrationClientId();  
registrationClientId.setEvaluationId("Evaluation Id");  
registrationClientId.setDevelopmentId("Development Id");  
registrationClientId.setStagingId("Stagging Id");  
registrationClientId.setTestingId("Testing Id");  
registrationClientId.setProductionId("Production Id");  
  
RegistrationDynamicConfiguration.getInstance().getJanRainConfiguration().setClientIds(registrationClientId);***

#### PILConfiguration

For replacing or adding PIL Configuration ids are quite simple now it can be done in following ways . Registration Framework will take care of replacing and adding ids .

API :

Present in **RegistrationDynamicConfiguration** class.

***/\*\*  
 \* Get PIL Configuration  
 \*  
 \* @return PILConfiguration  
 \*/  
public PILConfiguration getPilConfiguration() {}  
  
/\*\*  
 \* Set PIL Configuration  
 \*  
 \* @param pilConfiguration  
 \*/  
public void setPilConfiguration(PILConfiguration pilConfiguration) {}***

Present in **PILConfiguration** class

***/\*\*  
 \* Set Microsite Id  
 \*  
 \* @param micrositeId String  
 \*/  
public void setMicrositeId(String micrositeId) {}***

***/\*\*  
 \* Set Registration Environment  
 \*  
 \* @param registrationEnvironment String  
 \*/  
public void setRegistrationEnvironment(final String registrationEnvironment) {}  
/\*\*  
 \* Set Registration Environment  
 \*  
 \* @param registrationEnvironment  
 \*/  
public void setRegistrationEnvironment(final Configuration registrationEnvironment) {}***

***public void setCampaignID(String campaignID) {}***

Example:

***RegistrationDynamicConfiguration.getInstance().getPilConfiguration().setCampaignID("Campain ID");  
RegistrationDynamicConfiguration.getInstance().getPilConfiguration().setMicrositeId("Microsite ID");  
RegistrationDynamicConfiguration.getInstance().getPilConfiguration().setRegistrationEnvironment(Configuration);***

#### Flow

API :

Present in **RegistrationDynamicConfiguration** class.

***/\*\*  
 \* Get Flow  
 \*  
 \* @return Flow  
 \*/  
public Flow getFlow() {}  
  
/\*\*  
 \* Set Flow  
 \*  
 \* @param flow  
 \*/  
public void setFlow(Flow flow) {}***

Present in **Flow** class

***/\*\*  
 \* Status of email verification required  
 \*  
 \* @return boolean  
 \*/  
public Boolean isEmailVerificationRequired() {}  
  
/\*\*  
 \* Set Email verification status  
 \*  
 \* @param emailVerificationRequired  
 \*/  
public void setEmailVerificationRequired(Boolean emailVerificationRequired) {}  
  
/\*\*  
 \* Status of terms and condition acceptance  
 \*  
 \* @return boolean  
 \*/  
public Boolean isTermsAndConditionsAcceptanceRequired() {}  
  
  
/\*\*  
 \* Set terms and condition acceptance required or no  
 \*  
 \* @param termsAndConditionsAcceptanceRequired  
 \*/  
public void setTermsAndConditionsAcceptanceRequired(Boolean termsAndConditionsAcceptanceRequired) {}  
  
/\*\*  
 \* Get min age limit map  
 \*  
 \* @return Map of Country and Age restrictions  
 \*/  
public HashMap<String, String> getMinAgeLimit() {}  
  
/\*\*  
 \* Set Min age Limit  
 \*  
 \* @param minAgeLimit Map with Key value pair of country code and min age  
 \*/  
  
public void setMinAgeLimit(HashMap<String, String> minAgeLimit) {}  
  
/\*\*  
 \* Get minimium age for country  
 \*  
 \* @param countryCode Country code  
 \* @return integer value of min age if mapping available else 0  
 \*/  
public int getMinAgeLimitByCountry(String countryCode) {}***

Example :

***RegistrationDynamicConfiguration.getInstance().getFlow().setEmailVerificationRequired(false);  
RegistrationDynamicConfiguration.getInstance().getFlow().setTermsAndConditionsAcceptanceRequired(false);  
  
HashMap<String, String> ageMap = new HashMap<>();  
map.put("Country code", "Age Limit");  
  
RegistrationDynamicConfiguration.getInstance().getFlow().setMinAgeLimit(ageMap);***

#### HSDP Configuration (Optional)

API :

Present in **RegistrationDynamicConfiguration** class.

***/\*\*  
 \* Get HSDP configuration  
 \*  
 \* @return HSDPConfiguration  
 \*/  
public HSDPConfiguration getHsdpConfiguration() {}  
  
  
  
/\*\*  
 \* Set HSDP configuration  
 \*  
 \* @param hsdpConfiguration  
 \*/  
public void setHsdpConfiguration(HSDPConfiguration hsdpConfiguration) {}***

Present in **HSDPConfiguration** class

***/\*\*  
 \* Get All HSDP information with all configuration  
 \*  
 \* @return Map of Configuration and its informantion.  
 \*/  
public HashMap<Configuration, HSDPInfo> getHsdpInfos() {}  
  
/\*\*  
 \* Set HSDP Informations for different Configurations  
 \*  
 \* @param hsdpInfos Map of Configuration and its informantion.  
 \*/  
public void setHsdpInfos(HashMap<Configuration, HSDPInfo> hsdpInfos) {}  
  
/\*\*  
 \* Get HSDP information for specified configuration  
 \*  
 \* @param configuration  
 \* @return HSDPInfo Object  
 \*/  
public HSDPInfo getHSDPInfo(Configuration configuration) {}  
  
/\*\*  
 \* Set HSDP Information for specific Configuration  
 \*  
 \* @param configuration  
 \* @param hsdpInfo HSDP Information  
 \*/  
public void setHSDPInfo(Configuration configuration, HSDPInfo hsdpInfo) {}***

Present in **HSDPInfo** class

***/\*\*  
 \* Get Shared Id  
 \*  
 \* @return String shared id  
 \*/  
public String getSharedId() {}  
  
/\*\*  
 \* Set shared id  
 \*  
 \* @param sharedId String  
 \*/  
public void setSharedId(String sharedId) {}  
  
/\*\*  
 \* Get secret id  
 \*  
 \* @return String  
 \*/  
public String getSecreteId() {}  
  
/\*\*  
 \* Set secrete id  
 \*  
 \* @param secreteId String  
 \*/  
public void setSecreteId(String secreteId) {}  
  
/\*\*  
 \* Get base URL  
 \*  
 \* @return String  
 \*/  
public String getBaseURL() {}  
  
/\*\*  
 \* Set base URL  
 \*  
 \* @param baseURL  
 \*/  
public void setBaseURL(String baseURL) {}  
  
/\*\*  
 \* Get Application name  
 \*  
 \* @return String Application  
 \*/  
public String getApplicationName() {}  
  
/\*\*  
 \* Set application name  
 \*  
 \* @param applicationName  
 \*/  
public void setApplicationName(String applicationName) {}***

Example

***HSDPConfiguration hsdpConfiguration = new HSDPConfiguration();  
  
HSDPInfo hsdpInfo = new HSDPInfo();  
hsdpInfo.setApplicationName("Appname");  
hsdpInfo.setBaseURL("url");  
hsdpInfo.setSecreteId("secrete id");  
hsdpInfo.setSharedId("shared id");  
  
hsdpConfiguration.setHSDPInfo(Configuration.EVALUATION, hsdpInfo);  
  
RegistrationDynamicConfiguration.getInstance().setHsdpConfiguration(hsdpConfiguration);***

#### SigninProviders

API :

Present in **RegistrationDynamicConfiguration** class.

***/\*\*  
 \* Get Sign in providers  
 \*  
 \* @return SigninProviders  
 \*/  
public SigninProviders getSignInProviders() {}  
  
/\*\*  
 \* Set Social providers  
 \*  
 \* @param socialProviders  
 \*/  
public void setSignInProviders(SigninProviders socialProviders) {}***

Present in **SigninProviders** class

***/\*\*  
 \* Set providers in Hash map first param is country code and other is enabled providers list in Hash map  
 \*  
 \* @param providers  
 \*/  
public void setProviders(HashMap<String, ArrayList<String>> providers) {}  
  
/\*\*  
 \* Get providers  
 \*  
 \* @param countryCode Country code  
 \* @return List of providers  
 \*/  
public ArrayList<String> getProvidersForCountry(String countryCode) {}***

Example :

***//Configure Signin Providers  
HashMap<String, ArrayList<String>> providers = new HashMap<String, ArrayList<String>>();  
ArrayList<String> values1 = new ArrayList<String>();  
ArrayList<String> values2 = new ArrayList<String>();  
ArrayList<String> values3 = new ArrayList<String>();  
values1.add("googleplus");  
values1.add("facebook");***

***values2.add("googleplus");  
values2.add("facebook");  
  
values3.add("googleplus");  
values3.add("facebook");  
  
providers.put("NL", values1);  
providers.put("US", values2);  
providers.put("DEFAULT", values3);  
RegistrationDynamicConfiguration.getInstance().getSignInProviders().setProviders(providers);***

### Hybrid Configuration

Hybrid configuration is combination of static and dynamic Vertical app can provide both configuration. Registration framework will prioritize based on being Dynamic configuration higher priorities.

# Janrain Initialization

Janrain Initialization should be done from onCreate() of Application class of vertical projects along with Tagging setting and UI priority

## UI Priority function

UI priority function enables app to launch Create account at TOP or BOTTOM.Needs to set during initialization. By default Create account is TOP.

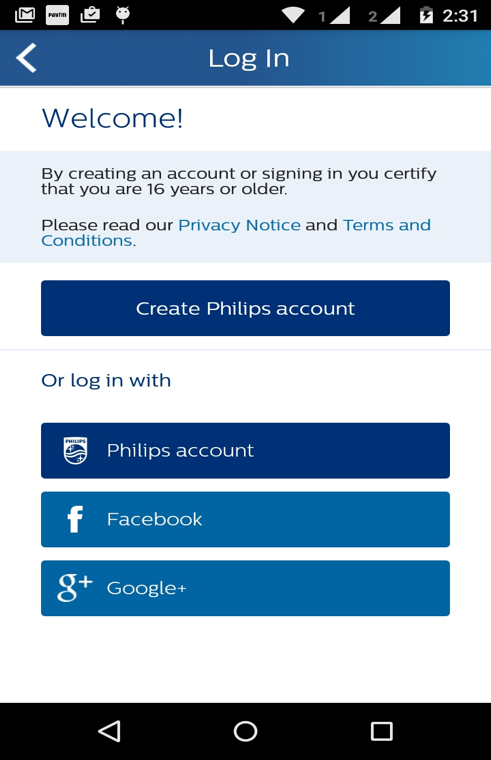
Possible selection is out of

**public enum RegistrationFunction {  
 Registration,  
 SignIn  
}**

### Create account on Top

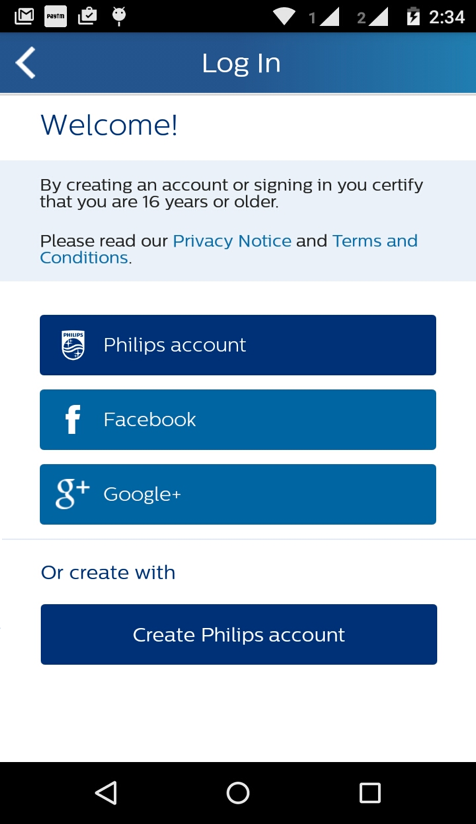
**//REGISTRATION UI PRIORITY(CREATE OR SIGNIN**

**RegistrationConfiguration.getInstance().setPrioritisedFunction(RegistrationFunction.Registration);**



### Sign In On top

**RegistrationConfiguration.getInstance().setPrioritisedFunction(RegistrationFunction.SignIn);**



## Configuration initialization

*public class XXXXX extends Application {*

*@Override  
public void onCreate() {  
 super.onCreate();*

*Tagging.enableAppTagging(true);  
Tagging.setTrackingIdentifier("integratingApplicationAppsId");  
Tagging.setLaunchingPageName("demoapp:home");  
Tagging.setDebuggable(true);  
Tagging.init( this, "Philips Registration");*

*String languageCode = Locale.getDefault().getLanguage();  
String countryCode = Locale.getDefault().getCountry();  
PILLocaleManager localeManager = new PILLocaleManager(this);  
localeManager.setInputLocale(languageCode,countryCode);*

*RegistrationConfiguration.getInstance().setPrioritisedFunction(RegistrationFunction.Registration);  
RegistrationHelper.getInstance().initializeUserRegistration(this);*

*}*

## Custom Language in Registartion UI

Their is no addition API provided in Registration if App need Specific Language UI to be displayed .Its Will be part of App initialization i.e. language passed during initialization process will be reflected.

Note : Need to initialize once again to change the Language

# Launching Registration component

* As Activity
* As Fragment

## Activity

### Supported orientation

Supported orientation integer constant

* ***ActivityInfo.SCREEN\_ORIENTATION\_PORTRAIT***
* ***ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE***
* ***ActivityInfo.SCREEN\_ORIENTATION\_SENSOR\_PORTRAIT***
* ***ActivityInfo.SCREEN\_ORIENTATION\_SENSOR\_LANDSCAPE***

### Launching default mode

Support in all orientation

Should call ***RegistrationLaunchHelper.launchDefaultRegistrationActivity (this);*** which takes Context as param

### Launching default mode with fixed orientation

Should call ***RegistrationLaunchHelper.launchDefaultRegistrationActivityWithFixedOrientation(this,orientationInetger);***which takes Context and Orientation integer as params

### Launching with context of Account settings

#### Launching with account settings in default mode

Support all orientation should call

***RegistrationLaunchHelper.launchRegistrationActivityWithAccountSettings(this);*** which takes Context as param

#### Launching with account settings with fixed orientation

Should call ***RegistrationLaunchHelper. launchRegistrationActivityWithFixedOrientationWithAccountSettings (this,orientationInetger);***

which takes Context and Orientation integer as params

#### Launching without account settings with fixed orientation

Should call ***RegistrationLaunchHelper. launchRegistrationActivityWithFixedOrientationWithOutAccountSettings (this,orientationInetger);***

which takes Context and Orientation integer as params

## Fragment

* Launching as fragment
* Back handling
* Updating title

### Launching as fragment

Should write and call following method to launch as fragment which take container Id , FragmentActivity ,AccountSettings screen as params.

***launchRegistrationFragment(R.id.fl\_reg\_fragment\_container, this);***

/\*\*  
 *\* Launch registration fragment  
 \*/****private void*** *launchRegistrationFragment(****int*** *container, FragmentActivity fragmentActivity,boolean isAccountSettings ) {*

*try {  
 FragmentManager mFragmentManager = fragmentActivity.getSupportFragmentManager();  
 RegistrationFragment registrationFragment = new RegistrationFragment();  
 Bundle bundle = new Bundle();  
 bundle.putBoolean(RegConstants.ACCOUNT\_SETTINGS,isAccountSettings);  
 registrationFragment.setArguments(bundle);  
 registrationFragment.setOnUpdateTitleListener(this);  
 FragmentTransaction fragmentTransaction = mFragmentManager.beginTransaction();  
 fragmentTransaction.replace(container, registrationFragment,  
 RegConstants.REGISTRATION\_FRAGMENT\_TAG);  
 fragmentTransaction.commitAllowingStateLoss();  
 } catch (IllegalStateException e) {  
 RLog.e(RLog.EXCEPTION,  
 "RegistrationActivity :FragmentTransaction Exception occured in addFragment :"  
 + e.getMessage());  
 }*

*}*

### Backstack handling

Should follow following code to handle back stack

*@Override  
public void onBackPressed() {  
if(!RegistrationLaunchHelper.isBackEventConsumedByRegistration(this)) {*

*// not consumed vertical code goes here // actual code  
 super.onBackPressed();  
}*

*}*

### Updating Title

Vertical projects should implement ***RegistrationTitleBarListener*** and should be referenced during launching of Registration fragment.Which override following methods

* ***updateRegistrationTitle*** *:*Triggered when fragment count is only 1 and Vertical app should write logic to update title and showing hamburger menu
* ***updateRegistrationTitleWithBack:*** Triggered when fragment count is greater than 1 and vertical app should update title and show back button
* ***updateRegistrationTitleWithOutBack*** *:* Triggred when welcome screen comes and need to do same activity as above two methods .

Ex:

*@Override  
public void updateRegistrationTitle(int titleResourceID) {  
 // Update title and show hamberger  
 //ivBack.setVisibility(View.INVISIBLE);  
 ivBack.setVisibility(View.VISIBLE);  
 TextView tvTitle = ((TextView) findViewById(R.id.tv\_reg\_header\_title));  
 tvTitle.setText(getString(titleResourceID));  
}  
  
@Override  
public void updateRegistrationTitleWithBack(int titleResourceID) {  
 // update title and show back  
 ivBack.setVisibility(View.VISIBLE);  
 TextView tvTitle = ((TextView) findViewById(R.id.tv\_reg\_header\_title));  
 tvTitle.setText(getString(titleResourceID));  
}  
  
@Override  
public void updateRegistrationTitleWithOutBack(int titleResourceID) {  
 // update title and show back  
 ivBack.setVisibility(View.INVISIBLE);  
 TextView tvTitle = ((TextView) findViewById(R.id.tv\_reg\_header\_title));  
 tvTitle.setText(getString(titleResourceID));  
}*

# Call-back’s

* On Click of Continue button in Welcome screen
* On Click of Terms & Conditions in Home screen
* On Click of Privacy policy in Home screen
* On Logout Success
* On Logout Failed
* On Logout success due to invalid access token

Vertical project should implement UserRegistrationListener which have

*@Override  
public void onUserRegistrationComplete(Activity activity) {*

*//Called on click of continue button   
}  
  
@Override  
public void onPrivacyPolicyClick(Activity activity) {  
//Called on click of privacy policy  
}  
  
@Override  
public void onTermsAndConditionClick(Activity activity) {  
 // Called on click of terms and condition   
}*

*@Override  
public void onUserLogoutSuccess() {  
//Called on logout success  
}  
  
@Override  
public void onUserLogoutFailure() {  
//on logout failed   
}  
  
@Override  
public void onUserLogoutSuccessWithInvalidAccessToken() {  
//on login success on invalid access token   
}*

And needs to register and unregister listener as follows

Example

In onCrate of implementing Activity

*RegistrationHelper.getInstance().registerUserRegistrationListener(this)*

In onDestoy of calling Activity.

*RegistrationHelper.getInstance().unRegisterUserRegistrationListener(this)*

# APP tagging

App tagging is supported in Registration Android and tested against

Adobe JAR version : [**v4.5.1 - Android**](https://github.com/Adobe-Marketing-Cloud/mobile-services/releases/tag/v4.5.1-Android)

ADBMobileConfig.json :

*{  
 "version" : "1.0",  
 "acquisition": {  
 "server": "c00.adobe.com"  
 },  
  
 "analytics" : {  
 "referrerTimeout": 5,   
 "rsids" : "philipsmobileappsdev",  
 "server" : "philips.112.2o7.net",  
 "charset" : "UTF-8",  
 "ssl" : false,  
 "offlineEnabled" : true,  
 "lifecycleTimeout" : 30,  
 "batchLimit" : 0,  
 "privacyDefault" : "optedin",  
 "poi" : [  
 ]  
 },  
 "target" : {  
 "clientCode" : "amsdk",  
 "timeout" : 5  
 },  
 "audienceManager" : {  
 "server" : ""  
 }  
}*

Tagging has following params to be set during initialization

*public class RegistrationApplication extends Application {  
  
 private RegistrationHelper mRegistrationHelper;  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 //Tagging  
 Config.setContext(getApplicationContext());*

*//Janrain Initialization  
 mRegistrationHelper = RegistrationHelper.getInstance();*

*//Params context and locale should be passed from application  
 //Coppa flow should be set to true  
 mRegistrationHelper.setCoppaFlow(false);  
  
 //Tagging specific  
 //Enabling or disabling registration tagging  
 AppTagging.enableAppTagging(true);*

*//Mandatory to set  
 AppTagging.setTrackingIdentifier("IntegratingApplicationAppId");*

*//Optional may set or may not set  
 AppTagging.setLaunchingPageName("RegistrationLaunchingPageName");  
  
 mRegistrationHelper.intializeRegistrationSettings(this,  
 Locale.getDefault());  
 }  
}*

# HSDP specific API’s

HSDP UUID and access token can be accessed from the DIUserProfile.java

## HSDP UUID

*User user = new User(context);  
user.getUserInstance(context).getHsdpUUID();*

## HSDP Access Token

*User user = new User(context);  
user.getUserInstance(context).getHsdpAccessToken();*

# Library version’s

Use following methods to get respective versions of registration

## Janrain Library

***Jump.getJumpVersion()***

## Locale Match Library

***PILLocaleManager.getLacaleMatchVersion()***

## Registration Library

***RegistrationHelper.getRegistrationApiVersion()***

# Supporting apps with Over 65K Methods

This is special case if app which has more than 65K methods do follow below link for reference .

<https://developer.android.com/tools/building/multidex.html>

According to this do following changes in gradle and application class

In Gradle file:

android {  
    compileSdkVersion 21  
    buildToolsVersion "21.1.0"  
  
    defaultConfig {  
        ...  
        minSdkVersion 14  
        targetSdkVersion 21  
        ...  
  
        // Enabling multidex support.  
        multiDexEnabled true  
    }  
    ...  
}  
  
dependencies {  
  compile 'com.android.support:multidex:1.0.1'  
}

In Application Class:

@Override  
public void onCreate() {  
   MultiDex.install(this);

  Super.onCreate();

}’

# Progaurd rules for Registration API component

As per Android Documentations mentioned in link <http://developer.android.com/tools/help/proguard.html> code will be behave unexpected way due to arbitrary rules .So following are progaurd rules one which needs to be added in vertical apps progaurd file to work registration as desired.

Note : Here all the configuration is mentioned pick one is needed as per your project flow

## General

*##General and network  
-keep public class javax.net.ssl.\*\*  
-keepclassmembers public class javax.net.ssl.\*\* {\*;}  
-keepclassmembers public class org.apache.http.\*\* {\*;}  
-dontwarn org.apache.\*\*  
-keep class org.apache.http.\*\* { \*; }  
-keep class android.net.http.\*\* { \*; }  
-renamesourcefileattribute SourceFile  
-keepattributes SourceFile,LineNumberTable  
-keepattributes InnerClasses,Exceptions*

*Tagging*

*-keep public class com.adobe.mobile.\*\* {\*;}*

*-keep public class com.philips.cdp.tagging.\*\* {\*;}*

## 

## Jump

*#Janrain lib  
-keep public class com.janrain.android.\*\* {\*;}  
-keep class com.janrain.android.Jump$\* {\*;}  
-keep class com.philips.cdp.registration.User$\*{\*;}  
-keep class com.janrain.android.capture.Capture$\* {\*;}  
  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void init(android.content.Context);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static java.lang.String objectToString(java.io.Serializable);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static java.lang.Object stringToObject(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void migrateUserData(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static byte[] encrypt(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static byte[] decrypt(byte[]);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void generateSecretKey();  
}  
  
  
-keepclasseswithmembernames public class com.janrain.android.\*\* {\*;}  
-keepclasseswithmembernames public class com.janrain.android.Jump {\*;}  
-keepclasseswithmembernames public class com.janrain.android.JumpConfig {\*;}  
-keepclasseswithmembernames public class com.janrain.android.TradSignInUi {\*;}*

## Localematch

*-keep public class com.philips.cdp.localematch.\*\* {\*;}*

## Registration API

*-keep class com.philips.cdp.registration.\*\* {\*;}  
-dontwarn com.philips.cdp.registration.\*\**

## Network APIs

*-dontwarn org.apache.\*\**

*-keep class org.apache.http.\*\* { \*; }*

*-keep class android.net.http.\*\* { \*; }*

## HSDP

*-keep class com.philips.dhpclient.\*\* {\*;}  
-keep class com.fasterxml.jackson.annotation.\*\* {\*;}  
-keep class com.fasterxml.jackson.core.\*\* {\*;}  
-keep class com.fasterxml.jackson.databind.\*\* {\*;}*

## Other platform specific

*#GSM  
-keep class com.google.android.gms.\* { public \*; }  
-dontwarn com.google.android.gms.\*\*  
-dontwarn org.w3c.dom.bootstrap.DOMImplementationRegistry  
  
#webkit  
-keep class android.net.http.SslError  
-keep class android.webkit.WebViewClient  
  
-dontwarn android.webkit.WebView  
-dontwarn android.net.http.SslError  
-dontwarn android.webkit.WebViewClient  
  
#notification  
-dontwarn android.app.Notification*

## Hocky SDK

*-keepclassmembers class net.hockeyapp.android.UpdateFragment {\*;}*

## Complete for configuration for Registration API

*##Registration API specific  
  
##General and network  
-keep public class javax.net.ssl.\*\*  
-keepclassmembers public class javax.net.ssl.\*\* {\*;}  
-keepclassmembers public class org.apache.http.\*\* {\*;}  
-dontwarn org.apache.\*\*  
-keep class org.apache.http.\*\* { \*; }  
-keep class android.net.http.\*\* { \*; }  
-renamesourcefileattribute SourceFile  
-keepattributes SourceFile,LineNumberTable  
-keepattributes InnerClasses,Exceptions  
  
  
  
  
  
  
  
#Hockey app and enabling excpetion catching  
-keepclassmembers class net.hockeyapp.android.UpdateFragment {\*;}  
  
  
#Tagging lib and jar  
-keep public class com.adobe.mobile.\*\* {\*;}  
-keep public class com.philips.cdp.tagging.\*\* {\*;}  
  
  
#Janrain lib  
-keep public class com.janrain.android.\*\* {\*;}  
-keep class com.janrain.android.Jump$\* {\*;}  
-keep class com.philips.cdp.registration.User$\*{\*;}  
-keep class com.janrain.android.capture.Capture$\* {\*;}  
  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void init(android.content.Context);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static java.lang.String objectToString(java.io.Serializable);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static java.lang.Object stringToObject(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void migrateUserData(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static byte[] encrypt(java.lang.String);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static byte[] decrypt(byte[]);  
}  
  
-keep public class com.philips.cdp.security.SecureStorage {  
 public static void generateSecretKey();  
}  
  
  
  
  
  
-keepclasseswithmembernames public class com.janrain.android.\*\* {\*;}  
-keepclasseswithmembernames public class com.janrain.android.Jump {\*;}  
-keepclasseswithmembernames public class com.janrain.android.JumpConfig {\*;}  
-keepclasseswithmembernames public class com.janrain.android.TradSignInUi {\*;}  
  
  
  
#Locale match  
-keep public class com.philips.cdp.localematch.\*\* {\*;}  
  
#Registration API  
-keep class com.philips.cdp.registration.\*\* {\*;}  
-dontwarn com.philips.cdp.registration.\*\*  
  
#HSDP Lib  
-keep class com.philips.dhpclient.\*\* {\*;}  
-keep class com.fasterxml.jackson.annotation.\*\* {\*;}  
-keep class com.fasterxml.jackson.core.\*\* {\*;}  
-keep class com.fasterxml.jackson.databind.\*\* {\*;}  
  
  
#GSM  
-keep class com.google.android.gms.\* { public \*; }  
-dontwarn com.google.android.gms.\*\*  
-dontwarn org.w3c.dom.bootstrap.DOMImplementationRegistry  
  
#webkit  
-keep class android.net.http.SslError  
-keep class android.webkit.WebViewClient  
  
-dontwarn android.webkit.WebView  
-dontwarn android.net.http.SslError  
-dontwarn android.webkit.WebViewClient  
  
#notification  
-dontwarn android.app.Notification*

# Manifest permission’s

Registration uses the following permissions

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

# Demo apps

DemoApps are available with package