**Product Registration Android Integration**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document History** | | | | |
| **Version** | **Date** | **Author** | **Section** | **Changes** |
| 0.1 | 02-05-2016 | Yogesh HS | All | Initial draft |

|  |  |
| --- | --- |
| **Author** | Yogesh HS |
| **Approved By** |  |
| **Email Id** | yogesh.hs@philips.com |

**Table of Contents**

1. Introduction ---------------------------------------------------- 3

2. Integration ---------------------------------------------------- 3

2.1 Artifactory ---------------------------------------------------- 3

2.2 Root gradle changes ---------------------------------------------------- 4

2.3 Dependencies ---------------------------------------------------- 4

2.3.1 Gradle Dependencies ---------------------------------------------------- 4

2.3.2 Library Dependencies ---------------------------------------------------- 4

2.4 Proxy dependencies ---------------------------------------------------- 5

2.5 Prerequisites ---------------------------------------------------- 5

3. Steps for API integration ---------------------------------------------------- 5

4. Steps for API integration ---------------------------------------------------- 6

# Introduction

This document provides an overview of integration procedure for Product Registration library in android mobile applications.

Source Path: [https://atlas.natlab.research.philips.com:7999/pr/hor-productregistration-android.git](ssh://git@atlas.natlab.research.philips.com:7999/pr/hor-productregistration-android.git)

# Integration

Integration can be done in following ways.

# 2.1 Artifactory

All dependent libraries should be downloaded from artifactory.

**Artifactory path:**

compile(group: 'com.philips.cdp', name: 'product-registration-lib', version: '1.0.0-rc.1', ext: 'aar')

{  
 transitive = **true** }

# 2.2 Root gradle changes

buildscript {  
 repositories {  
 maven {  
 url **'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/jcenter'** }  
 dependencies {  
 classpath **'com.android.tools.build:gradle:1.5.0'** classpath **'org.jfrog.buildinfo:build-info-extractor-gradle:3.1.2'** classpath **'com.github.dcendents:android-maven-gradle-plugin:1.3'** }  
  
}  
}  
  
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()  
 }  
}

# 2.3 Dependencies

# 2.3.1 Gradle dependencies

Just by adding below gradle dependencies, Production Registration and nested possible libraries will be downloaded from artifactory.

compile(group: 'com.philips.cdp', name: 'product-registration-lib', version: '1.0.0-rc.1', ext: 'pom'){

transitive=**true**

}

# 2.3.2 Library dependencies

1: User Registration : 5.0.0-rc.10

2: PRX Client : 2.0.0-rc.7

3: Local Match : 2.0.0-rc.4

# 2.4 Proxy dependencies

Gradle dependencies can get some network/proxy related issues. In order to fix this issue, we are using below proxy settings in gradle.properties of root folder.

**systemProp.https.proxyHost**=**42.99.164.34**

**systemProp.https.proxyPort**=**10015**

We are using this proxy settings locally. But Eindhoven, does not use above proxy

Settings.

# 2.5 Prerequisites

i) Application need to know **CTN** number for each product used in app. Ex: HD8967/01.Max CTN is allowed.

ii) Sector.

iii) Catalog.

iv) Serial number.

v) Purchase date.

Example: **Sector**: B2C , **catalog**: CONSUMER, 1344 and yyy-mm-dd,

**Note:** Above information is used for backend services and mostly uses PRX system.

Hence please provide valid and complete CTN number as input.

# Steps for API integration

**Following to be invoked in Application class**

1. Invoke new ProdRegHelper().init();
2. Set the locale match , kindly find below code for reference

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

**Following to be invoked to register product**

// pass CTN, Sector, Catalog

Product product = new Product("HC540/83",Sector.B2C, Catalog.CONSUMER);

// set serial number

product.setSerialNumber(mSerialNumber);

// set purchase date in ("YYYY-MM-DD") format

product.setPurchaseDate(mPurchaseDate);

//set email configuration as true or false, if true this will email the product that was registered

product.sendEmail(String.valueOf(true));

final ProdRegListener listener = new ProdRegListener() {

@Override

public void onProdRegSuccess(RegisteredProduct registeredProduct, UserWithProducts userWithProducts) {

//on sucess additional implementation

}

@Override

public void onProdRegFailed(RegisteredProduct registeredProduct, UserWithProducts userWithProducts) {

//on failed additional implementation

}

};

// adding call back listener while registering product will trigger callbacks Success and Failed

prodRegHelper.addProductRegistrationListener(listener);

//API to register product for current signed-in user

prodRegHelper.getSignedInUserWithProducts().registerProduct(product);

**Note:**

Kindly process the call back object registeredProduct to get following information

1. To get CTN and Serial number of product registered
2. To get state of Registration (Registered, Pending, Failed)
3. To get error state when product registration failed. (Ex: Invalid Serial Number, Product Already Registered etc.,)

**Following to invoked to get Registered Products list**

ProdRegHelper prodRegHelper = new ProdRegHelper();

// Listener to be initialized to get call backs

final RegisteredProductsListener registeredProductsListener = new RegisteredProductsListener() {  
 @Override  
 public void getRegisteredProductsSuccess(final List<RegisteredProduct> registeredProducts, final long timeStamp) {}

prodRegHelper.getSignedInUserWithProducts().getRegisteredProducts(registeredProductsListener, Sector.B2C, Catalog.CONSUMER);

# Reference

If need more details about above api , refer API Spec Doc.