PRXClient IOS Integration

|  |  |  |  |
| --- | --- | --- | --- |
| Document History | | | |
| Version | Date | Section | Changes |
| 1.0.0 |  | All | Initial Release |
| 3.0.0 | 19- Jan-2017 | All | Now using Appinfra |

|  |  |
| --- | --- |
| Author | Anurag , G Kavya , Adarsh Shetty |
| Approved by |  |
| Email ID | [anurag.gautam@philips.com](mailto:anurag.gautam@philips.com)  [kavya.g.kurpad@philips.com](mailto:kavya.g.kurpad@philips.com)  adarsha.shetty@philips.com |

Table of Contents

[Introduction. 3](#_Toc471832963)

[1 Client 3](#_Toc471832964)

[2 RequestManager 3](#_Toc471832965)

[3 Responsehandler 3](#_Toc471832966)

[4 Product/ProductSummary/ProductAssets 3](#_Toc471832967)

[Installation 3](#_Toc471832968)

[Code Example - Quick integration 3](#_Toc471832969)

[Create request 3](#_Toc471832970)

[Execute Request 3](#_Toc471832971)

# Introduction.

The main functionality of this library is to download any data related to product present on PRX. It can be used by consumer care, registration and different applications. This library can be reused by other projects with minimal development changes as a generic network component as well. PRX client library exposes classes and APIs to clients to send a request and get a response. Library also helps clients to customise the requests.

1 Client

It can be an application, consumer care component or registration component.

2 Request Manager

It provides set of public APIs for placing requests from client and also talks to Network wrapper class for performing network operations.

3 Response handler

Handles response like invoking respective builders to build the response. It also invokes listener/blocks.

4 Product/ ProductSummary/ ProductAssets Model data for each request type.

# Installation

# \* Add prx to dependencies in gradle file

# compile(group: 'com.philips.cdp', name: 'prx', version: '3.0.0-SNAPSHOT', ext: 'aar')

### 

### Create request

### 

ProductSummaryRequest mProductSummeryBuilder = new ProductSummaryRequest(selectedCtn, mRequestTag);

### Execute Request

Create Request Manager with appinfra and call execute API

//Dont create new instance of Appinfra instead inject the existing instance

PRXDependencies prxDependencies = new PRXDependencies(context , mAppInfra);

RequestManager mRequestManager = new RequestManager();

mRequestManager.init(prxDependencies); // pass prxdependency.

mRequestManager.executeRequest(prxRequest, new ResponseListener() {

@Override

public void onResponseSuccess(ResponseData responseData) {

String str = responseData.getClass().toString();

}

@Override

public void onResponseError(PrxError prxError) {

Log.d(TAG, "Response Error Message PRX: " + prxError.getDescription());

}

});

**Note :**

1. **Added AppInfra dependency in PRX**

**2)    We are taking Locale from ServiceDiscovery , Hence setting LocaleMatch in Prx is not required . We are using LocaleMatch only to get the Enums  Sector and CTN values.**

**3)    Created PRXDependency class in PRX to inject AppInfra . Example usage in Demo App :**

**PRXDependencies   prxDependencies = new PRXDependencies(context , mAppInfra); // use existing appinfra instance**

**RequestManager mRequestManager = new RequestManager();**

**mRequestManager.init(prxDependencies); // pass prxdependency**

**ProductSummaryRequest mProductSummeryBuilder = new ProductSummaryRequest(selectedCtn, mRequestTag);**

**mProductSummeryBuilder.setSector(selectedSector);**

**mProductSummeryBuilder.setCatalog(selectedCatalog);**

**mRequestManager.executeRequest(prxRequest, new ResponseListener() {**

**@Override**

**public void onResponseSuccess(ResponseData responseData) {**

**String str = responseData.getClass().toString();**

**}**

**@Override**

**public void onResponseError(PrxError prxError) {**

**Log.d(TAG, "Response Error Message PRX: " + prxError.getDescription());**

**}**

**});**

    4) **Sector and catalog can be set via getter/setter or It can initialized via contructor.**

**For ex: ProductSupportRequest mProductSupportBuilder = new ProductSupportRequest(selectedCtn, selectedSector, selectedCatalog, mRequestTag);**

Note :

**PRXDependencies(Context context, AppInfraInterface appInfra) Is deprecated .**

**We have added new Constructor**

**public PRXDependencies(Context context, AppInfraInterface appInfra , String parentTLA)**

**Here the 3rd Parameter is the Three letter Acronym which will be used to identify the component and it will be logged .**