Software Release Report - AppInfra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document History | | | | |
| Version | Date | Author | Section | Changes |
| 2.2.0 | 05-July-17 | Ravi Kiran  Hashim  Leslie | All | New Release Features:   * Secure Storage * Tagging * Logging * PRX Client * LocaleMatch * TimeSync * Internationalization * AppIdentity * Service Discovery * App Configuration * REST Client * Language Pack * App Update |

|  |  |  |  |
| --- | --- | --- | --- |
| Administrative Information | | | |
| Project Name | AppInfra | Project Identification | NA |
| Project Leader |  | Date | 05-Jul-2017 |
| Quality Leader |  | Form Filled by | Ravi Kiran HR |
| Name  Address  Phone  Fax |  | | |

|  |  |
| --- | --- |
| Author | Ravi Kiran HR |
| Approved by |  |
| Email ID | [Ravi.kiran@philips.com](mailto:Ravi.kiran@philips.com) |

**Code:**

Product: AppInfra

Release Version: 2.1.0

Release Date: 28-April-2017

Integration Document: MobileAppInfra\_PI17.2\_Integration\_IOS.docx

**Release Description:**

AppInfra is a horizontal library project developed both on Android and IOS platform. It offers common functionalities related to vertical app which can be used in all Philips applications.

Find **documents**  at <https://atlas.natlab.research.philips.com/bitbucket/projects/MAIL/repos/app-infra_ios/browse/Documents>

Find the **demo app** at <https://atlas.natlab.research.philips.com/bitbucket/projects/MAIL/repos/app-infra_ios/browse/Source/DemoAppInfra>

Find the **framework** at <https://atlas.natlab.research.philips.com/bitbucket/projects/MAIL/repos/app-infra_ios/browse/Framework>

**Library** :**pod 'AppInfra', '2.2.0-SNAPSHOT’**

**Source Code Links:**

http://tfsemea1.ta.philips.com:8080/tfs/TPC\_Region24/CDP2/\_git/ail-ios-appinfra

**System Requirement / Configuration:**

IOS Version: IOS 9 and above

## Latest Version PI17.3

\* new feature appupdate

\* logs component name is now TLA

\* appinfra code snippets added to repo

\* locale special handling for zh-Hans(zh\_CN) & zh-Hant(zh\_TW)

\* new api in logging for dictionary

\* Demo with microapp template

\* Timesync issues fixed

\* Tagging pagename validations

\* Deletion of language pack if locale change

\* AppConfig reset api added

\* one log to show device name and os

**Changes in PI 17.2**

\* New feature language pack

\* Limiting service discovery environment

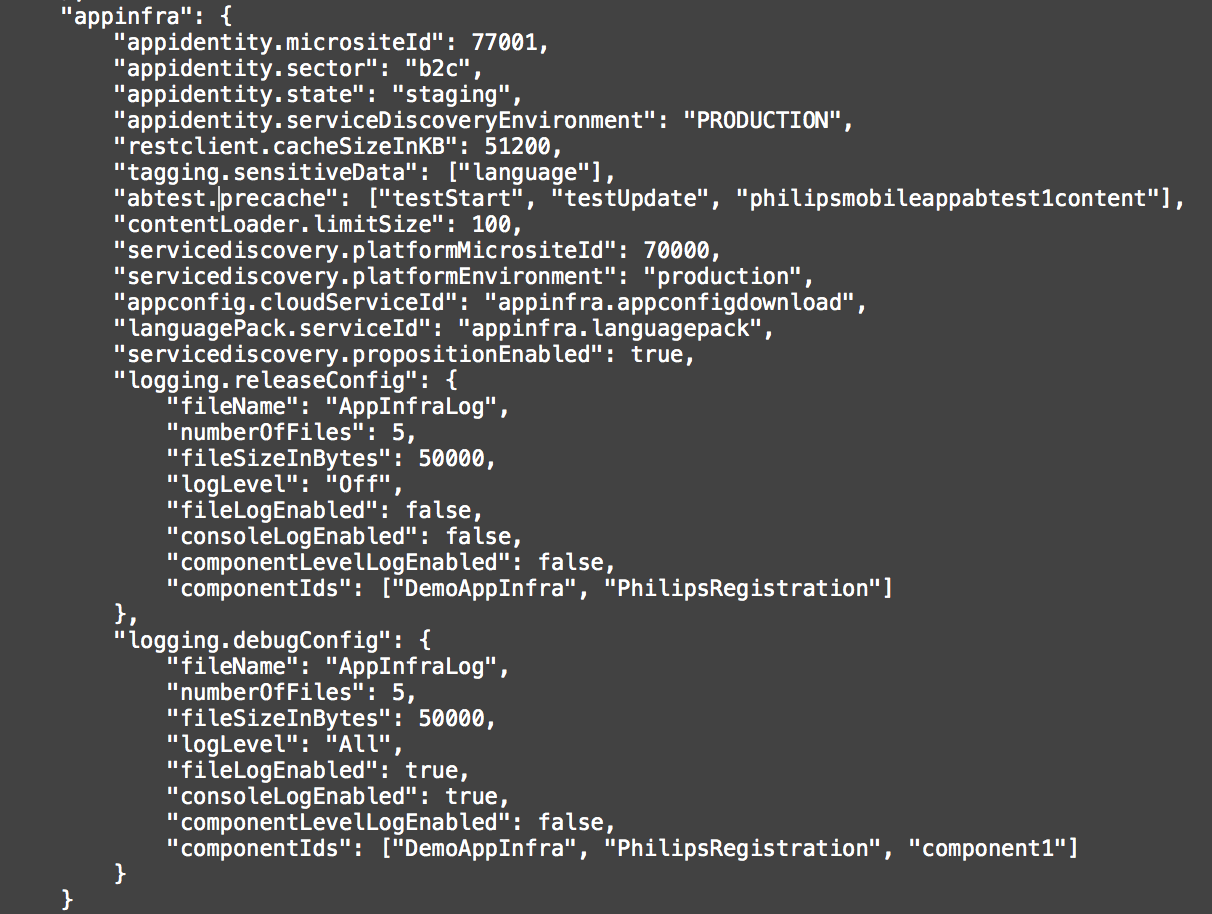
\* PSR defects fixes

\* Component version added

\* Mutiple URL configs merge in service discovery

component

\* In AppConfing file AppInfra section has been modified, appinfra section of Appconfig file looks like below sample

\* Logging configurations added to appconfig file

\* new how to pages in confluence

\* isSyncronised api in Time module, also few bugs were fixed

\* fixed REST client cache encryption bug

\* callback api for tagging

\* option to disable proposition microsite id

\* new api in tagging to get tracking identifier

\* api to detect device is jailbroken

\* sd improvements - caching home country

**Changes in PI 17.1**

\* US11828 - Service Discovery inject parameters in the URL

\* US13345 - Service Discovery refresh when home country is changed

\* App config keys are now case insensitive

\* App config cloud support

\* Service discovery data persistant across app launches

\* Service Discovery server call will be locked for 10 seconds

\* PRX dependency removed from AppInfra

\* Service Discovery alternate implementation using CSV available as seperate repo

\* Service discovery match by country result filtered based on user preferred languages in case of multiple locales for a country

\* Inject parameters to service url in service discovery

\* Multiple microsite id support one for proposition one for platform

\* Content loader data will be flushed for language change

\* content loader limit per instance

\* Network reachability is implemented in AppInfra as part of REST client

\* Internationalization now returnes the locale that app UI was rendered. Country part may not be there

**Supported Features:**

* **Secure Storage:** Secure Storage is used to store secret value in device storage with encrypted way using AES Encryption. It uses key value pair concept to store data inside the apps.
* **Tagging:** App tagging is used to track pages and button actions of the propositions or common components with page/action name and several other default values such as timestamps,device info,OS info etc.
* **Logging:** Logging is used to maintain the logs which user access of the propositions or common components with page/action name and several other default values such as UTC timestamps, Log type, Component ID, Event and Message.
* **PRX Client:** PRX Client is used to download any data related to product present on PRX. It can be used various Philips applications. It can be reused by other projects with minimal development changes as a generic network component as well.
* **Service Discovery:** Service Discovery reduces the hard dependency between app and cloud services. The main idea is that the list of URLs that are to be used by an application is maintained server side, at the service discovery server. The app only has to download this list from **one single global location**, this list tells the app where all other cloud services can be found.
* **Internationalization It** provided APIs to fetch Locale from Android settings.
* **Time Sync** feature provides an APIs to retrieve the UTC server time accurately.

It also perform synchronization for every 24hrs and whenever there is a Data and time change.

* **App Identity:** The App identity feature shall provide an API to get the app release status: development, test, acceptance, production.

The App identity feature shall obtain the technical app name, app version and app release status automatically from the build application build process.

* **App Configuration:** The app configuration module maintains configuration settings of the app and it’s included common components, in the form of key value pair.
* **Rest Client:** The REST client module simplifies communication with cloud services that use a REST based interface. Which is built upon AFNetworking Library
* **ABTesting** :The A/B test client module assists in performing A/B tests in the application. The client communicates with server which distributes users over the test experiences. A test is identified by its test name.when offline, the cached data is used if available instead of the default value to ensure a consistent user experience irrespective of network availability.
* **Content Loader:**This feature has a dedicated purpose of caching the articles from the **CQ5** server for the application. The articles could be in the form of text, images, document etc. This library will download all the articles available at the application specified URL and store it internally
* **Language Pack:** All apps contain text which is visualized to the user in some way, mainly this text is shown directly in the UI. A part of this text is more or less static and fundamental to the operation of the app. For that reason, this text is embedded according to the App UI internationalization guidelines. The text is shown in the locale as selected by the user on his device. This module enable to change these texts dynamically from cloud