White Paper

Single Sign-On for Mobile Apps

Mobile App Single Sign-On Overview

Employees in enterprises have to use several corporate mobile apps on a regular basis, such as an HR app to apply for time off, a travel app to book flights and hotels for business travel, and Box/SharePoint to share files among employees.

The process typically goes something like this:

1. Employee logs into the HR app to apply for time off.
2. But they also have a business trip coming up that they need to book, so they login to the travel app with completely different credentials.
3. After that's done, they wish to share their travel itinerary with their assistant via Sharepoint, which requires yet another login credential.
4. Employee gets frustrated because of multiple password prompts, and their tiny smartphone keyboard that makes it more difficult for them to type in their password with the required capital letters, numerals, and special symbols.

Mobile App Single Sign-On (SSO) can eliminate those frustrations by enabling users to login once with their user credentials and then authenticate to other apps automatically.

This provides enterprises the ability to allow employees to use multiple enterprise applications like HR, Expenses, and Travel using a single login.

Desktop and websites have provided SSO for a while, but a robust solution hasn't been easy to implement for mobile apps until now.

Implementing App SSO with Kony

Overview of the Kony Mobility Platform

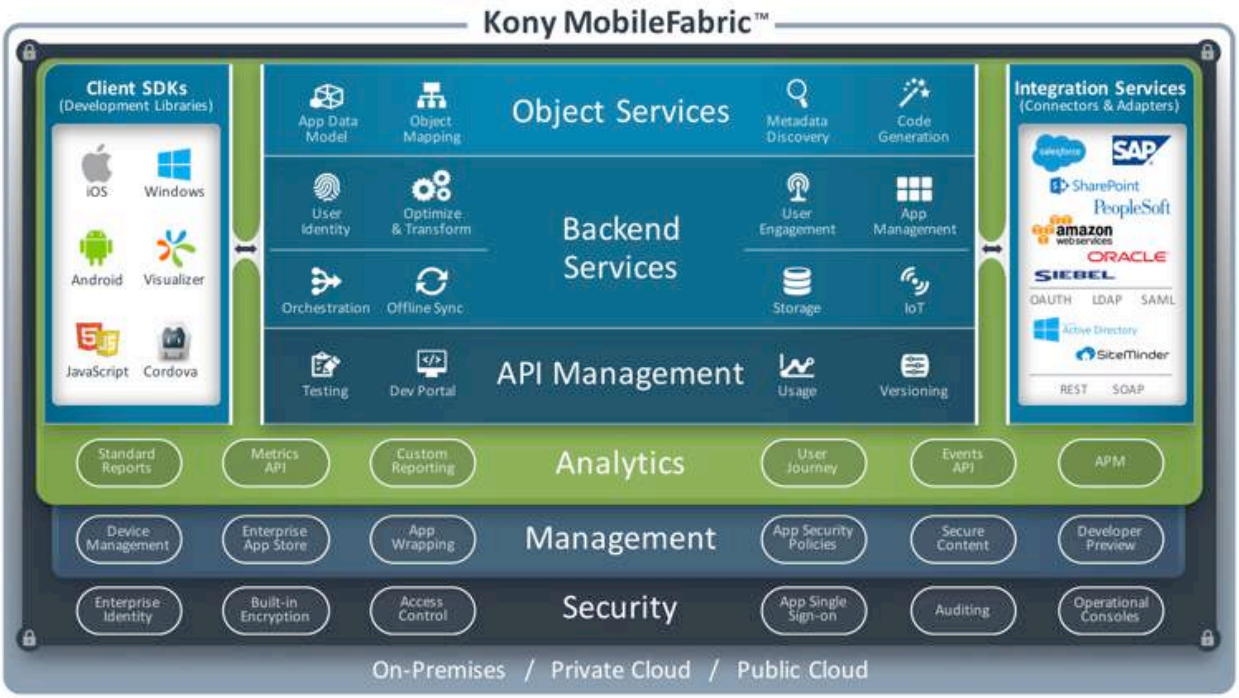
Top analysts such as Gartner and Forrester have recognized the Kony Mobility Platform for years as a leader in mobility.

Kony Visualizer® is the powerful enterprise-grade platform for front-end development for designing, developing, and deploying rapid, low-code, native mobile and web apps using open and standards-based tools with JavaScript.

For more info on Kony Visualizer visit: **http://www.kony.com/products/visualizer**

Kony MobileFabric® is the leading mobile middleware that provides a robust and secure mobile back-end platform and is available on both the cloud and on premises. Kony MobileFabric provides full app lifecycle management, mobile analytics for business insights and DevOps, API management, and key back-end services that allow developers to focus on creating exceptional mobile app experiences instead of developing complex back-end integration. MobileFabric's REST-based services architecture creates an agile framework for application development using the client tools developers are familiar with such as iOS native, Android native, HTML5, Cordova, Windows, and Kony Visualizer.

MobileFabric's core feature capabilities are shown in the below figure.



For more info on Kony MobileFabric visit: **http://www.kony.com/products/mobilefabric**

Configuring Apps for SSO

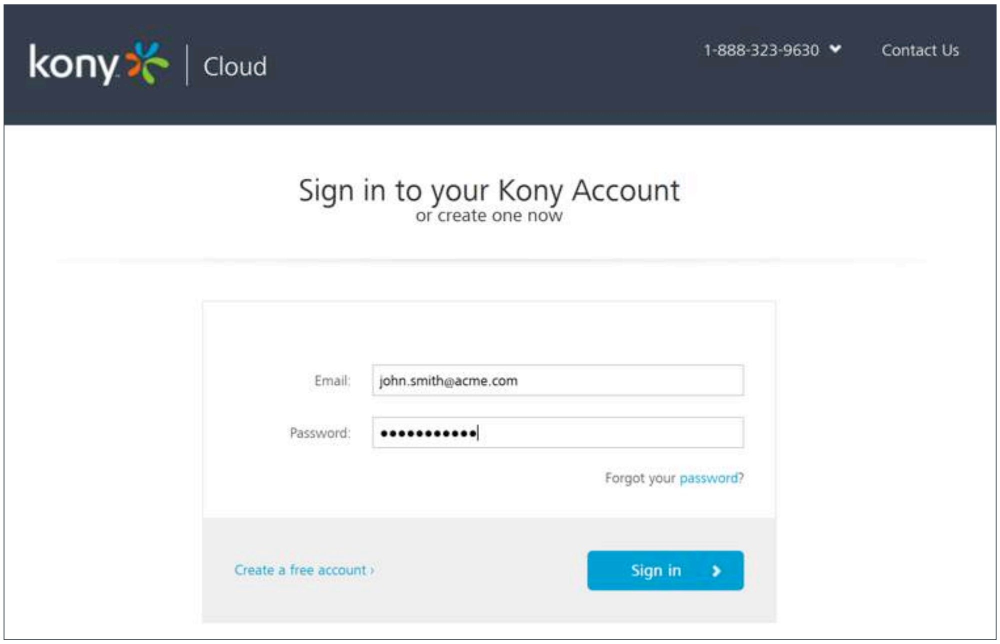
The following section provides a high-level walk through of configuring SSO using Kony MobileFabric and Kony Visualizer using two existing sample apps.

Back-end Configuration in Kony MobileFabric

MobileFabric's Identity Management services simplify the connectivity between data sources and their identity providers by mapping an identity provider to the data sources with which it interacts. When a user logs into an application, the application makes a request to MobileFabric to obtain an identity token over a secure connection. MobileFabric obtains the credentials for each of the back-end systems and stores them in the token manager. This allows MobileFabric to gather multiple identities while returning a single

Kony Identity Token back to the mobile application. Any tokens stored within MobileFabric are encrypted and stored in a secured credential vault. When the application requests data from back-end services, it sends the Kony Token as part of the request to MobileFabric. MobileFabric retrieves the back-end credentials from its credential vault in the token manager and uses the proper credentials to retrieve data from the back-end data source.

Step1: Login to MobileFabric console:



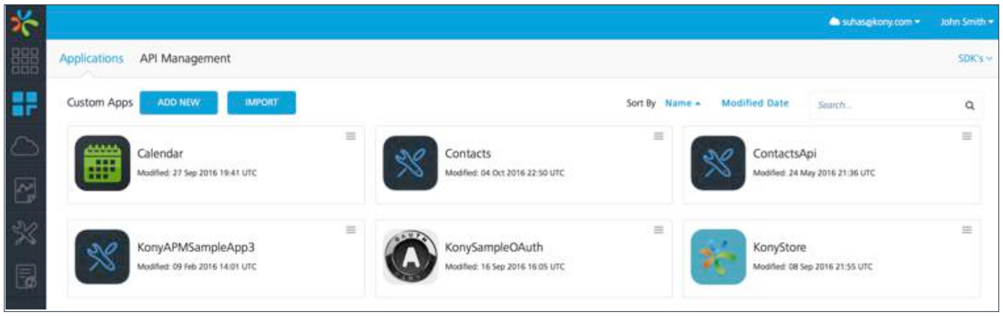
Sign in to your Kony Account
or create one now

Email: john.smith@acme.com

Password: •••••••••••

Step2: Navigate to apps tab and select app

Navigate to the apps tab and click on the app for which SSO is to be enabled



Custom Apps

Calendar
Modified: 27 Sep 2016 19:41 UTC

Contacts
Modified: 04 Oct 2016 22:50 UTC

ContactsApi
Modified: 24 May 2016 21:36 UTC

KonyAPMSampleApp3
Modified: 09 Feb 2016 14:01 UTC

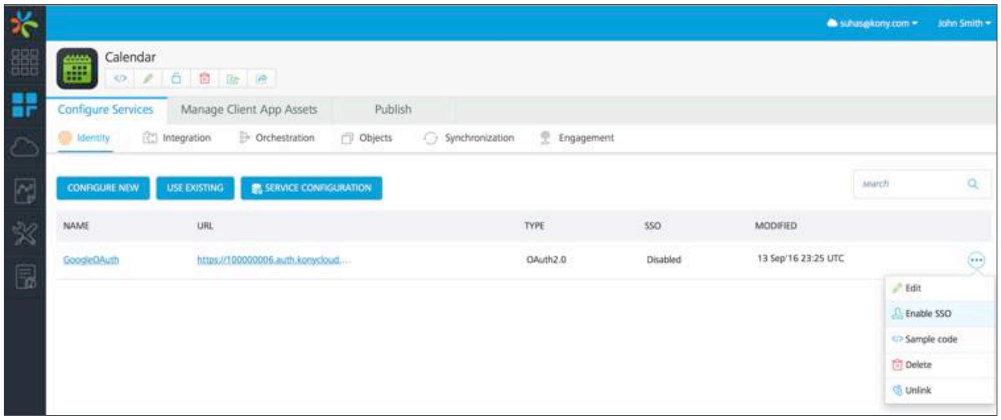
KonySampleOAuth
Modified: 16 Sep 2016 16:05 UTC

KonyStore
Modified: 08 Sep 2016 21:55 UTC

Step3: Enable SSO in Identity Service

In the Identity section of the services ensure that the Identity Service that you wish to use for authentication to the app is present.

Click on the button at the end of the service name and select Enable SSO.

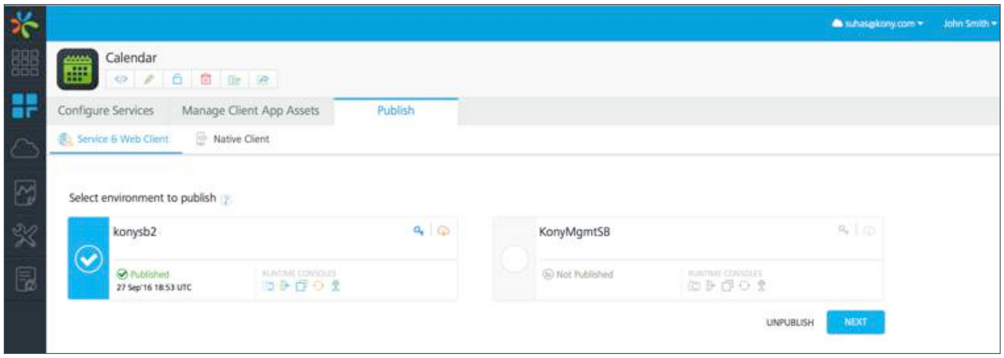


NAME	URL	TYPE	SSO	MODIFIED
GoogleOAuth	https://100000006.auth.konycloud....	OAuth2.0	Disabled	13 Sep'16 23:25 UTC

Step 4: Republish services

Once SSO has been enabled for a service the service needs to be republished to the server for it to take effect.

Click on the Publish tab and select the server to which the app was deployed and click Next.

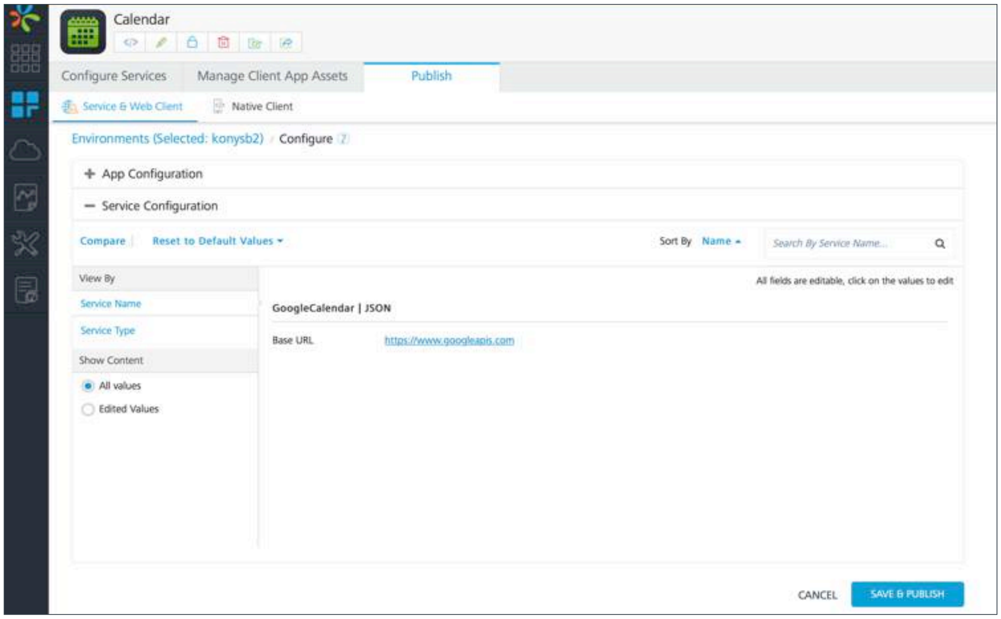


Select environment to publish

konysb2
Published
27 Sep'16 18:53 UTC
RUNTIME CONSOLES

KonyMgmtSB
Not Published
RUNTIME CONSOLES

In the subsequent screen click on Save & Publish for the update to the service with SSO enabled to be published to server.



GoogleCalendar | JSON

Base URL https://www.googleapis.com

Repeat the steps for each app that needs to participate in SSO. Ensure that all Identity Services that need to participate in the SSO are a part of each app.

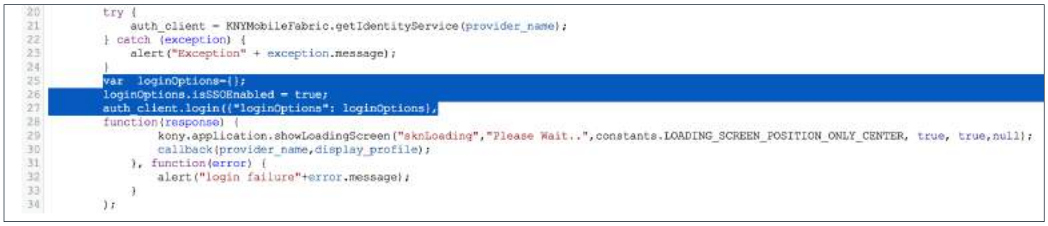
Client app configuration in Kony Visualizer

The client application is built on Kony Visualizer and it interacts with the Kony MobileFabric server using the Kony MobileFabric SDK.

The MobileFabric SDK has APIs to connect and use the various features of the MobileFabric server such as Identity Services for authentication, integration services for connecting to the backend, and metrics services to send app data for analytics.

Step 1: Enable SSO Option for Identity Service in client code

The SSO option has to be enabled on the client while the identity service is invoked for authentication. Sample code invocation snippet:



```
        try {
            auth_client = KNYMobileFabric.getIdentityService(provider_name);
        } catch (exception) {
            alert("Exception" + exception.message);
        }
        var  loginOptions={};
        loginOptions.isSSOEnabled = true;
        auth_client.login({"loginOptions": loginOptions},
        function(response) {
                kony.application.showLoadingScreen("sknLoading","Please Wait..",constants.LOADING_SCREEN_POSITION_ONLY_CENTER, true, true,null);
                callback(provider_name,display_profile);
            }, function(error) {
                alert("login failure"+error.message);
            }
        );
```

The highlighted section of the snippet enables SSO while invoking the Identity Service.

Enable permissions/capabilities per platform

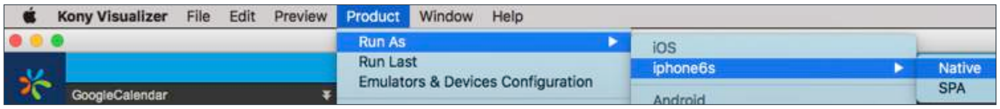
The final step of SSO configuration is specific to each platform.

Enabling SSO for iOS

To enable SSO in iOS we need to specify the unique location in OS that will be storing the SSO token and provide means to specific apps that need to participate in SSO to have access to the same.

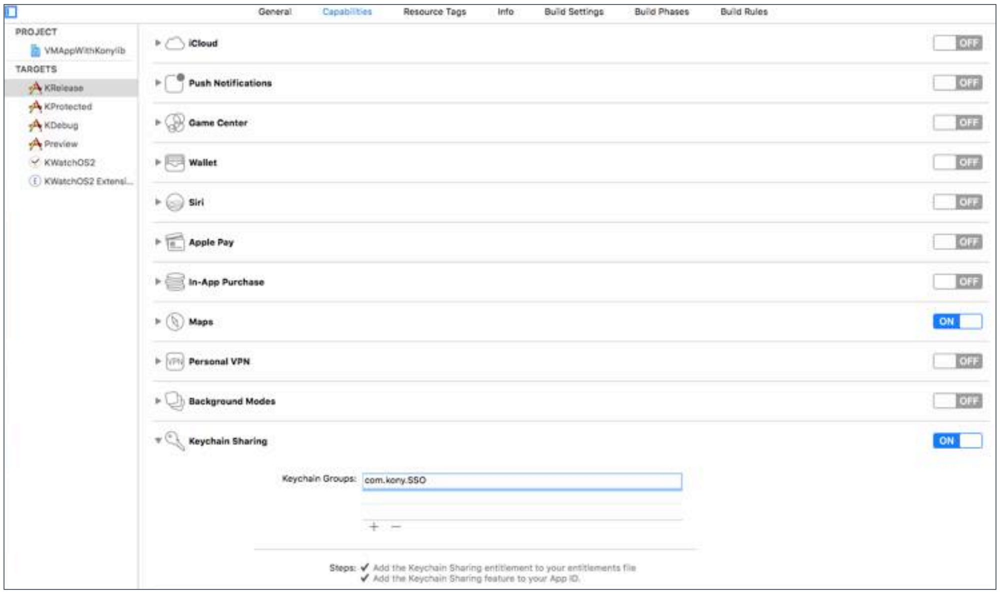
Step 2: Build the app for iOS

Navigate to Product from the top menu from Visualizer. Run As for iOS and open the Xcode project.



Step 3: Enable keychain sharing

Navigate to the capabilities tab in Xcode project.
Click on Keychain sharing to turn the capability ON.
Enter a name for the keychain group. Typical name suggested is com.orgname.SSO



iCloud OFF
Push Notifications OFF
Game Center OFF
Wallet OFF
Siri OFF
Apple Pay OFF
In-App Purchase OFF
Maps ON
Personal VPN OFF
Background Modes OFF
Keychain Sharing ON

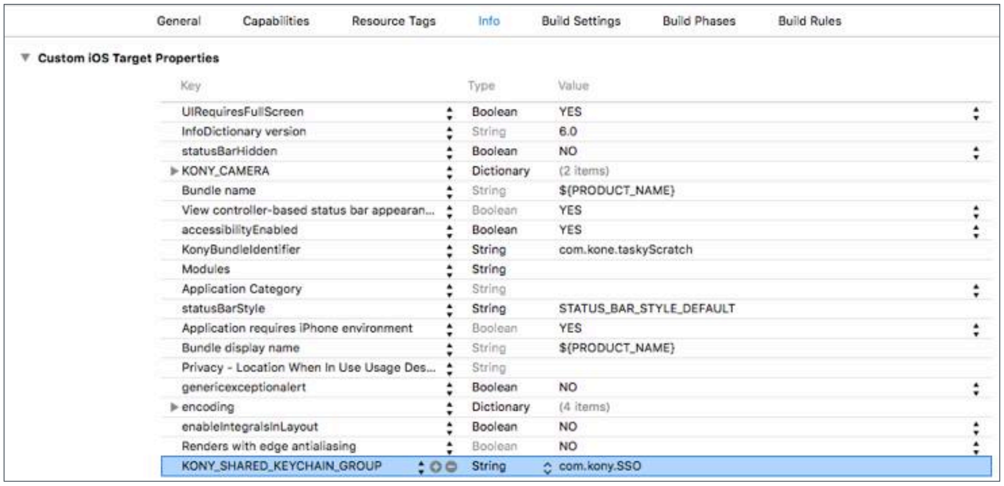
Keychain Groups: com.kony.SSO

+ −

Steps: ✓ Add the Keychain Sharing entitlement to your entitlements file
✓ Add the Keychain Sharing feature to your App ID.

Step 4: Add keychain group

Navigate to Info tab in Xcode project.
Create a new property in Custom iOS Target Properties section called "KONY_SHARED_KEYCHAIN_GROUP" of type String.
Set the name of the keychain group created in previous step as value for the new property.



▼ Custom iOS Target Properties

Key	Type	Value
UIRequiresFullScreen	Boolean	YES
InfoDictionary version	String	6.0
statusBarHidden	Boolean	NO
▶ KONY_CAMERA	Dictionary	(2 items)
Bundle name	String	${PRODUCT_NAME}
View controller-based status bar appearan...	Boolean	YES
accessibilityEnabled	Boolean	YES
KonyBundleIdentifier	String	com.kone.taskyScratch
Modules	String	
Application Category	String	
statusBarStyle	String	STATUS_BAR_STYLE_DEFAULT
Application requires iPhone environment	Boolean	YES
Bundle display name	String	${PRODUCT_NAME}
Privacy - Location When In Use Usage Des...	String	
genericexceptionalert	Boolean	NO
▶ encoding	Dictionary	(4 items)
enableIntegralsInLayout	Boolean	NO
Renders with edge antialiasing	Boolean	NO
KONY_SHARED_KEYCHAIN_GROUP	String	com.kony.SSO

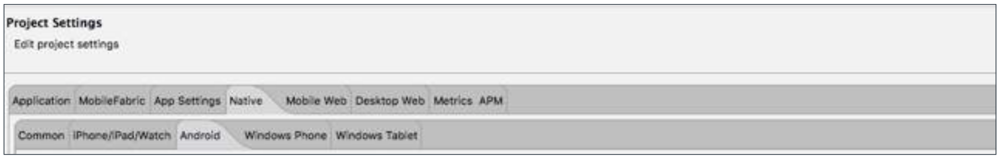
Repeat the steps for each app that needs SSO capability enabled.
The Xcode property settings are only needed the first time a project is opened in Xcode. For subsequent runs the values are saved and the app can be deployed to the device directly from Visualizer.

Enabling SSO for Android

Open the Project settings of the app in Kony Visualizer and set tags in application's manifest

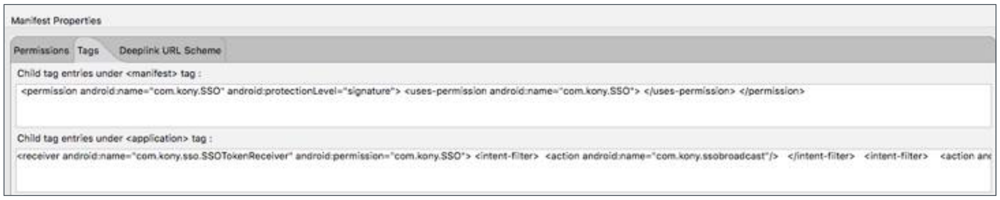
Step2: Navigate to Android properties

Open Project Settings -> Native tab -> Android


Project Settings
Edit project settings

Step 3: Add app permissions

Scroll to the bottom and click on Tags tab in Manifest properties section.



Child tag entries under <manifest> tag :

```
<permission android:name="com.kony.SSO" android:protectionLevel="signature"> <uses-permission android:name="com.kony.SSO"> </uses-permission> </permission>
```

Child tag entries under <application> tag :

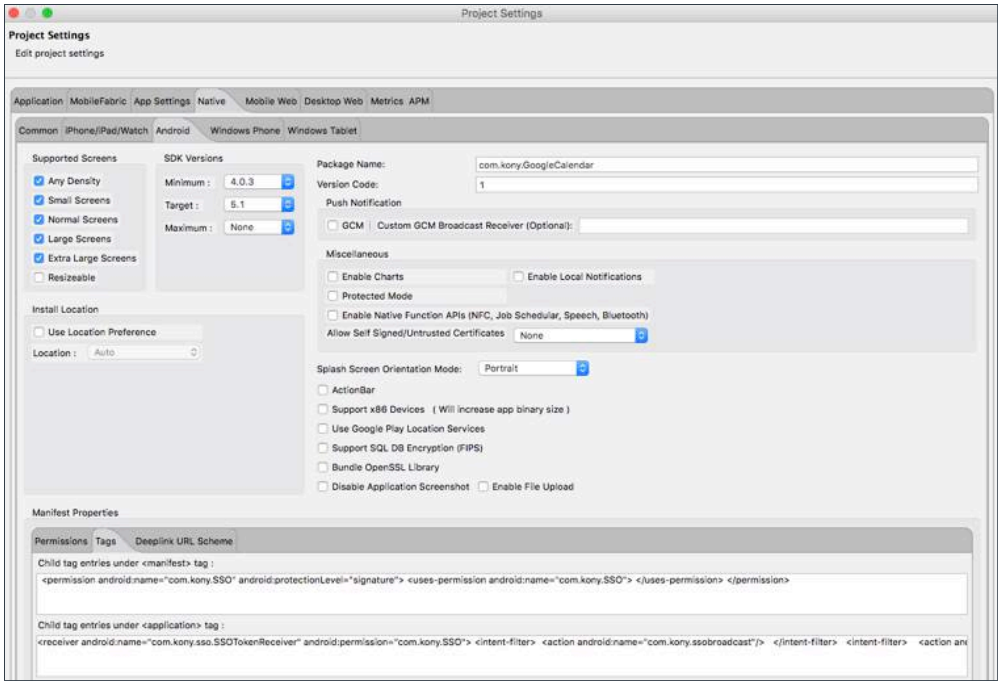
```
<receiver android:name="com.kony.sso.SSOTokenReceiver" android:permission="com.kony.SSO"> <intent-filter>  <action android:name="com.kony.ssobroadcast"/>  </intent-filter>  <intent-filter>  <action an
```

Enter the below snippet in manifest tag:

```
<permission android:name="com.kony.SSO" android:protectionLevel="signature"> <uses-permission
android:name="com.kony.SSO"> </uses-permission> </permission>
```

Enter the below snippet in manifest tag:

```
<receiver android:name="com.kony.sso.SSOTokenReceiver" android:permission="com.kony.SSO">
<intent-filter>  <action android:name="com.kony.ssobroadcast"/>
</intent-filter>
<intent-filter>    <action android:name="android.intent.action.PACKAGE_ADDED"/>
<data android:scheme="package"/>
</intent-filter>
</receiver>
```



Supported Screens

☑ Any Density
☑ Small Screens
☑ Normal Screens
☑ Large Screens
☑ Extra Large Screens
□ Resizeable

SDK Versions

Minimum : 4.0.3
Target : 5.1
Maximum : None

Install Location

□ Use Location Preference
Location : Auto

Package Name: com.kony.GoogleCalendar
Version Code: 1

Push Notification

□ GCM | Custom GCM Broadcast Receiver (Optional):

Miscellaneous

□ Enable Charts □ Enable Local Notifications
□ Protected Mode
□ Enable Native Function APIs (NFC, Job Schedular, Speech, Bluetooth)
Allow Self Signed/Untrusted Certificates None

Splash Screen Orientation Mode: Portrait
□ ActionBar
□ Support x86 Devices (Will increase app binary size)
□ Use Google Play Location Services
□ Support SQL DB Encryption (FIPS)
□ Bundle OpenSSL Library
□ Disable Application Screenshot □ Enable File Upload

Child tag entries under <manifest> tag :

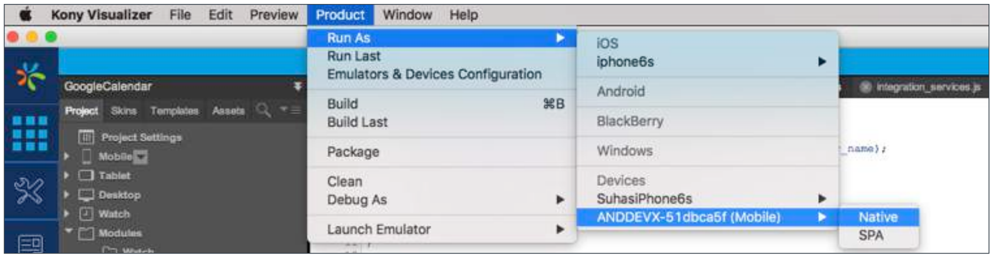
```
<permission android:name="com.kony.SSO" android:protectionLevel="signature"> <uses-permission android:name="com.kony.SSO"> </uses-permission> </permission>
```

Child tag entries under <application> tag :

```
<receiver android:name="com.kony.sso.SSOTokenReceiver" android:permission="com.kony.SSO"> <intent-filter>  <action android:name="com.kony.ssobroadcast"/>  </intent-filter>  <intent-filter>  <action an
```

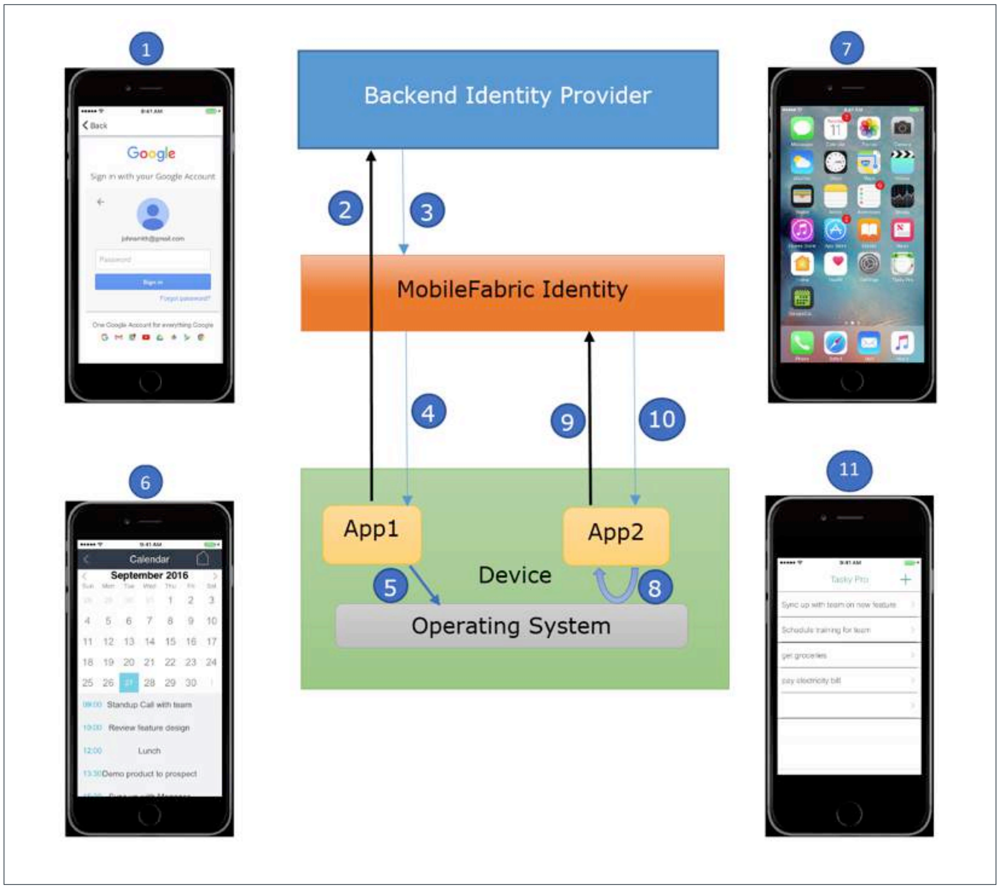
Step 4: Build and run the app from Visualizer on Android

Repeat the steps for each app that needs SSO capability enabled.



How it Works

The following diagram illustrates how Kony App SSO works on the client device at runtime using a sample Calendar and Task app.



1. User launches calendar app and enters user credentials
2. App sends credentials to backend identity provider
3. Backend Identity provider returns its authentication token to MobileFabric (MF) Identity
4. MF Identity sends an SSO token back to the app
5. App saves the SSO token securely in the mobile operating system
6. App loads the authenticated user's calendar
7. User launches Task app
8. The app fetches the SSO token from device
9. App sends SSO token to MF identity
10. MF identity validates the SSO token, authenticates user, and sends an MF token back to the app
11. User views his tasks without being prompted for credentials

For more info on SSO and MobileFabric check out our **video tutorials** and **documentation**.

For more information, please visit www.kony.com. Connect with Kony on Twitter, Facebook, and LinkedIn.
9225 Bee Cave Road, Building A, Suite 300, Austin, TX 78733 1.888.323.9630 | info@kony.com | kony.com