Android Workbook

Table of Contents

Session- Set Up the BlackBerry Dynamics Development Environment

Overview

This section contains step-by-step instructions for several common tasks for application developers using the BlackBerry Dynamics Software Development Kit (SDK) for Android.

The tasks covered include:

- Install the BlackBerry Dynamics SDK.
- Run a sample application.

BlackBerry Dynamics™ was previously Good Dynamics™. The earlier product name appears in some locations, and so does the abbreviation GD.

Compatibility

The instructions in this manual have been tested in the following environment:

Component	**Version**
Android Studio	Android Studio 3.1.4
Android SDK Build-tools	27.0.3
Android SDK Platform	23
Physical device running Android	6.0.1 & 7.1.2
BlackBerry Dynamics SDK for Android	4.2.0.69
BlackBerry UEM	12.9.0

Prepare for Application Development

To develop BlackBerry Dynamics applications for Android you will need to have installed the BlackBerry Dynamics SDK for Android.

The SDK can be downloaded from the application developer portal website. You must register on the portal to download the SDK. Instructions for the application developer portal, and for installing the SDK, are included below.

The SDK has several prerequisites for:

- Software, such as a supported version of the Android SDK Platform. See the Compatibility section, above.
- Hardware, as required by the above software.

Use of an integrated development environment (IDE) is not strictly necessary for Android application development. However, this manual assumes that you are using the Android Studio IDE. The Android Studio download includes the IDE and the Android SDK tools.

Application Developer Portal

The BlackBerry Dynamics application developer portal website is intended to provide the resources that you need to develop applications with the SDK.

You can access the application developer portal from the following URL: https://developers.blackberry.com

Some resources are available without registration, but you will need to register to download software, gain access to detailed technical material, or post on the developer community forum. Anybody can register and there is no charge.

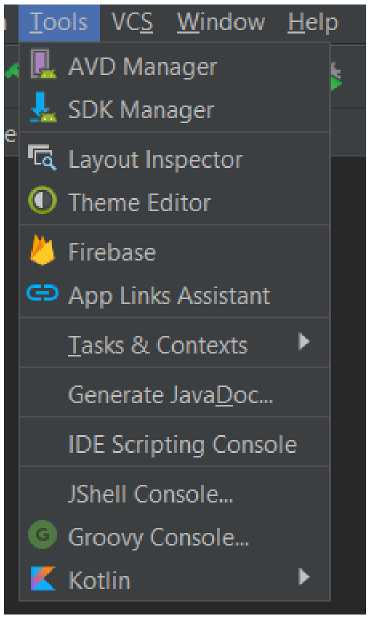
You will need an account on the application developer portal to download the BlackBerry Dynamics SDK for Android applications. If you don't already have one, you can register for a free account.

Download & Install the BlackBerry Dynamics SDK using the Android SDK Manager

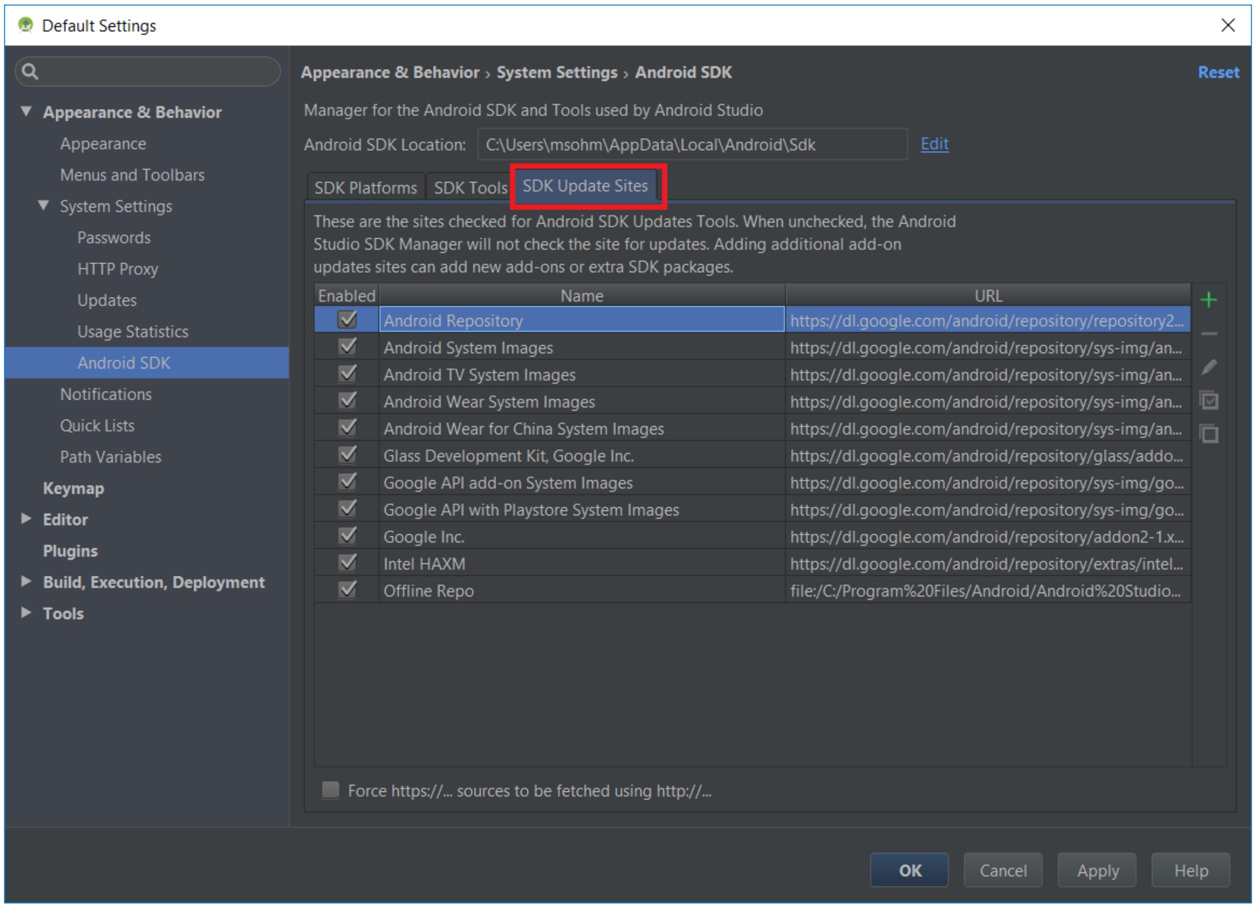
It is recommended to install the BlackBerry Dynamics SDK for Android using the Android SDK Manager. Doing so will allow you to update the SDK using the Android SDK Manager as new versions become available. If you are unable to install using the Android SDK Manager, the SDK can also be installed manually using the steps in the “Manually Download & Install the BlackBerry Dynamics SDK for Android” section.

The BlackBerry Dynamics SDK for Android can be installed using the Android SDK Manager tool using the following steps.

1. Click on the Tools menu and then click on SDK Manager.



2. Click on the SDK Update Sites tab.



Manager for the Android SDK and Tools used by Android Studio

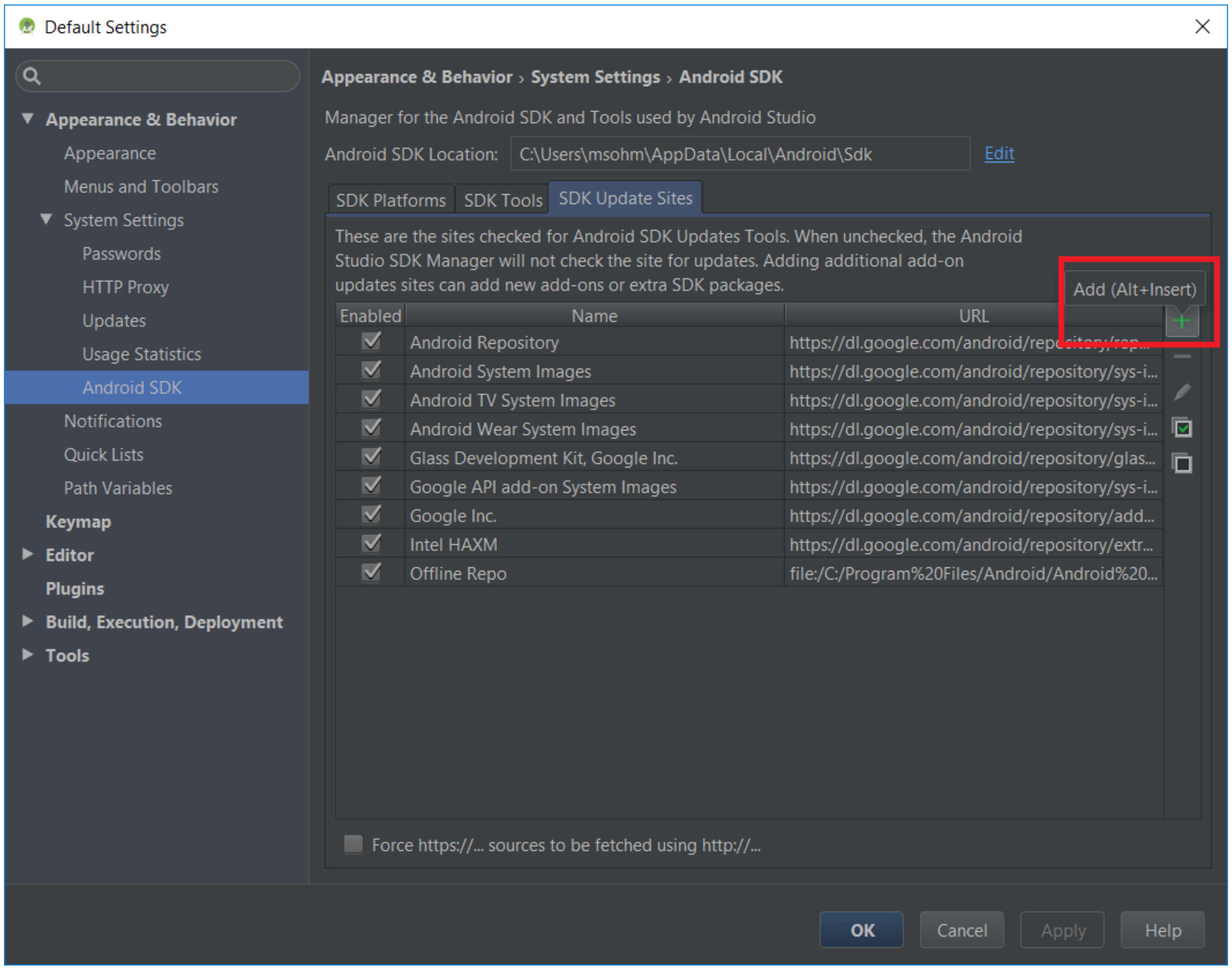
Android SDK Location: C:\Users\msohm\AppData\Local\Android\Sdk Edit

These are the sites checked for Android SDK Updates Tools. When unchecked, the Android Studio SDK Manager will not check the site for updates. Adding additional add-on updates sites can add new add-ons or extra SDK packages.

Enabled	Name	URL
✓	Android Repository	https://dl.google.com/android/repository/repository2...
✓	Android System Images	https://dl.google.com/android/repository/sys-img/an...
✓	Android TV System Images	https://dl.google.com/android/repository/sys-img/an...
✓	Android Wear System Images	https://dl.google.com/android/repository/sys-img/an...
✓	Android Wear for China System Images	https://dl.google.com/android/repository/sys-img/an...
✓	Glass Development Kit, Google Inc.	https://dl.google.com/android/repository/glass/addo...
✓	Google API add-on System Images	https://dl.google.com/android/repository/sys-img/go...
✓	Google API with Playstore System Images	https://dl.google.com/android/repository/sys-img/go...
✓	Google Inc.	https://dl.google.com/android/repository/addon2-1.x...
✓	Intel HAXM	https://dl.google.com/android/repository/extras/intel...
✓	Offline Repo	file:/C:/Program%20Files/Android/Android%20Studio...

☐ Force https://... sources to be fetched using http://...

3. Click on the Add button.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

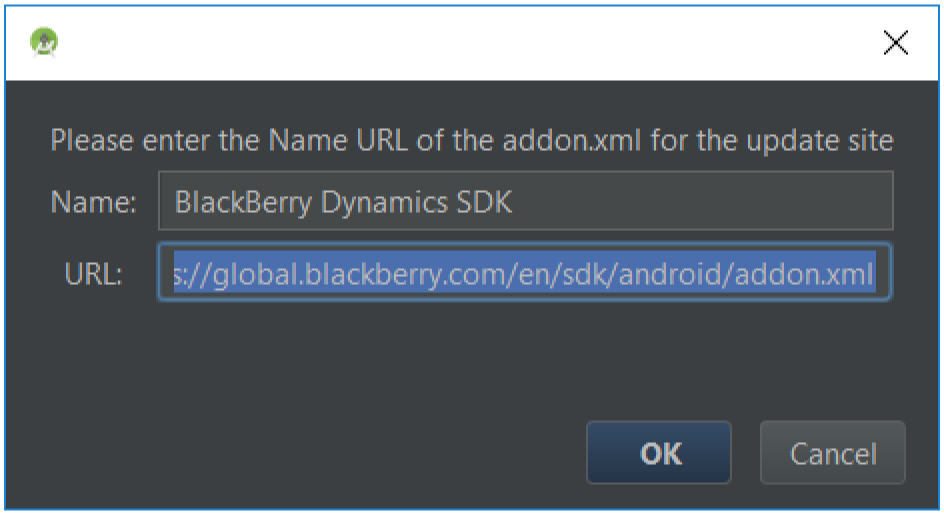
Android SDK Location: C:\Users\msohm\AppData\Local\Android\Sdk Edit

These are the sites checked for Android SDK Updates Tools. When unchecked, the Android Studio SDK Manager will not check the site for updates. Adding additional add-on updates sites can add new add-ons or extra SDK packages.

Enabled	Name	URL
✓	Android Repository	https://dl.google.com/android/repository/rep...
✓	Android System Images	https://dl.google.com/android/repository/sys-i...
✓	Android TV System Images	https://dl.google.com/android/repository/sys-i...
✓	Android Wear System Images	https://dl.google.com/android/repository/sys-i...
✓	Glass Development Kit, Google Inc.	https://dl.google.com/android/repository/glas...
✓	Google API add-on System Images	https://dl.google.com/android/repository/sys-i...
✓	Google Inc.	https://dl.google.com/android/repository/add...
✓	Intel HAXM	https://dl.google.com/android/repository/extr...
✓	Offline Repo	file:/C:/Program%20Files/Android/Android%20...

Force https://... sources to be fetched using http://...

4. Create a new update site with the following information:
   - Name: BlackBerry Dynamics SDK
   - URL: https://global.blackberry.com/en/sdk/android/addon.xml



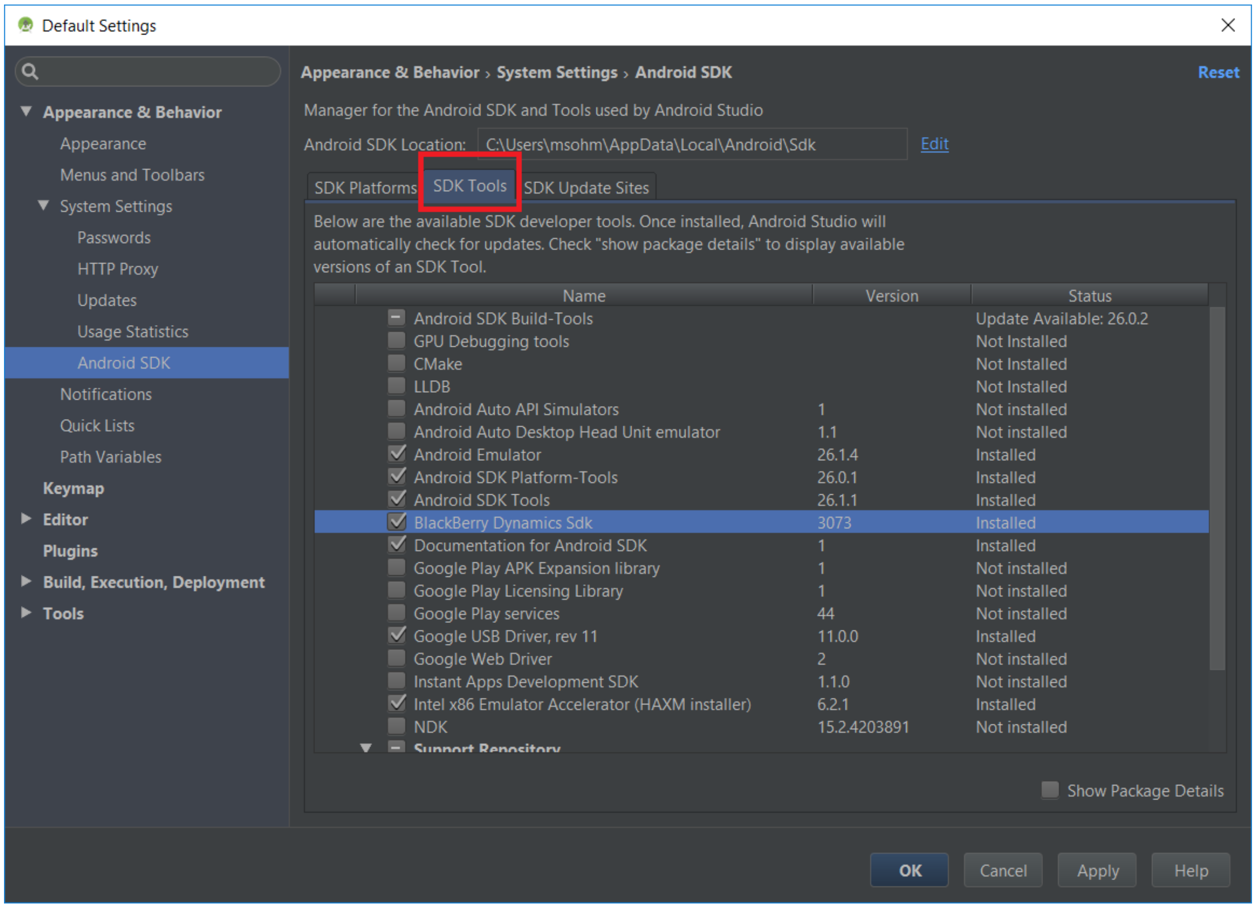
Please enter the Name URL of the addon.xml for the update site

Name: BlackBerry Dynamics SDK

URL: s://global.blackberry.com/en/sdk/android/addon.xml

5. Click OK.

6. Click on the SDK Tools tab.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

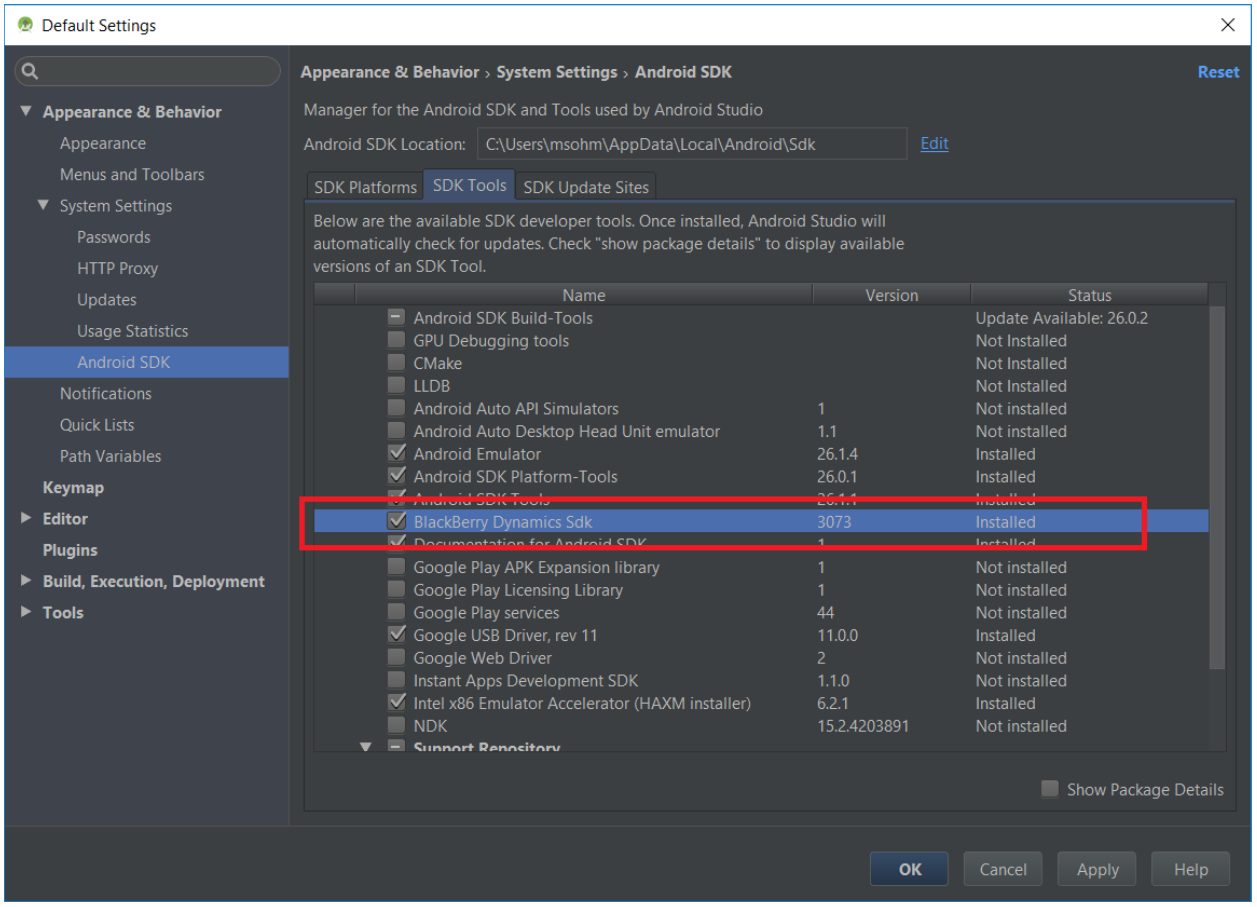
Android SDK Location: C:\Users\msohm\AppData\Local\Android\Sdk

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
Android SDK Build-Tools		Update Available: 26.0.2
GPU Debugging tools		Not Installed
CMake		Not Installed
LLDB		Not Installed
Android Auto API Simulators	1	Not installed
Android Auto Desktop Head Unit emulator	1.1	Not installed
✓ Android Emulator	26.1.4	Installed
✓ Android SDK Platform-Tools	26.0.1	Installed
✓ Android SDK Tools	26.1.1	Installed
✓ BlackBerry Dynamics Sdk	3073	Installed
✓ Documentation for Android SDK	1	Installed
Google Play APK Expansion library	1	Not installed
Google Play Licensing Library	1	Not installed
Google Play services	44	Not installed
✓ Google USB Driver, rev 11	11.0.0	Installed
Google Web Driver	2	Not installed
Instant Apps Development SDK	1.1.0	Not installed
✓ Intel x86 Emulator Accelerator (HAXM installer)	6.2.1	Installed
NDK	15.2.4203891	Not installed
Support Repository		

Show Package Details

7. Check BlackBerry Dynamics Sdk.



Default Settings

Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: C:\Users\msohm\AppData\Local\Android\Sdk

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
Android SDK Build-Tools		Update Available: 26.0.2
GPU Debugging tools		Not Installed
CMake		Not Installed
LLDB		Not Installed
Android Auto API Simulators	1	Not installed
Android Auto Desktop Head Unit emulator	1.1	Not installed
✓ Android Emulator	26.1.4	Installed
✓ Android SDK Platform-Tools	26.0.1	Installed
✓ Android SDK Tools	26.1.1	Installed
✓ BlackBerry Dynamics Sdk	3073	Installed
✓ Documentation for Android SDK	1	Installed
Google Play APK Expansion library	1	Not installed
Google Play Licensing Library	1	Not installed
Google Play services	44	Not installed
✓ Google USB Driver, rev 11	11.0.0	Installed
Google Web Driver	2	Not installed
Instant Apps Development SDK	1.1.0	Not installed
✓ Intel x86 Emulator Accelerator (HAXM installer)	6.2.1	Installed
NDK	15.2.4203891	Not installed
Support Repository		

8. Click Apply.

9. Click OK when prompted to install the SDK.

10. Accept the license agreement and click Next.

11. Click Finish once the SDK has completed installing.

The BlackBerry Dynamics SDK is now ready to use in your Android projects.

Manually Download & Install the BlackBerry Dynamics SDK for Android

It is recommended to install the BlackBerry Dynamics SDK for Android using the Android SDK Manager. Doing so will allow you to update the SDK using the Android SDK Manager as new versions become available. If you successfully completed the steps above in the “Download &

Install the BlackBerry Dynamics SDK for Android using Android SDK Manager", you already have the SDK installed and can skip the steps in this section.

If you were unable to install using the Android SDK Manager using the steps in the previous section, the SDK can be manually installed using the following steps.

1. Download the BlackBerry Dynamics SDK for Android from the application developer portal. Open the URL below and select Android as your platform if not already selected. https://developers.blackberry.com/us/en/products/blackberry-dynamics/android.html

   You will retrieve a single file in zip format. The name will be similar to BlackBerry_Dynamics_for_Android_r####.zip

2. Open the Android home directory on your computer in a file manager application, or change to the directory in a terminal window. The Android home directory is the one named sdk, typically the following locations, which may be hidden by default.

   Windows: `C:\Users\`*`[username]`*`\AppData\Local\Android\sdk\`

   Mac: `/Users/`*`[username]`*`/Library/Android/sdk/`

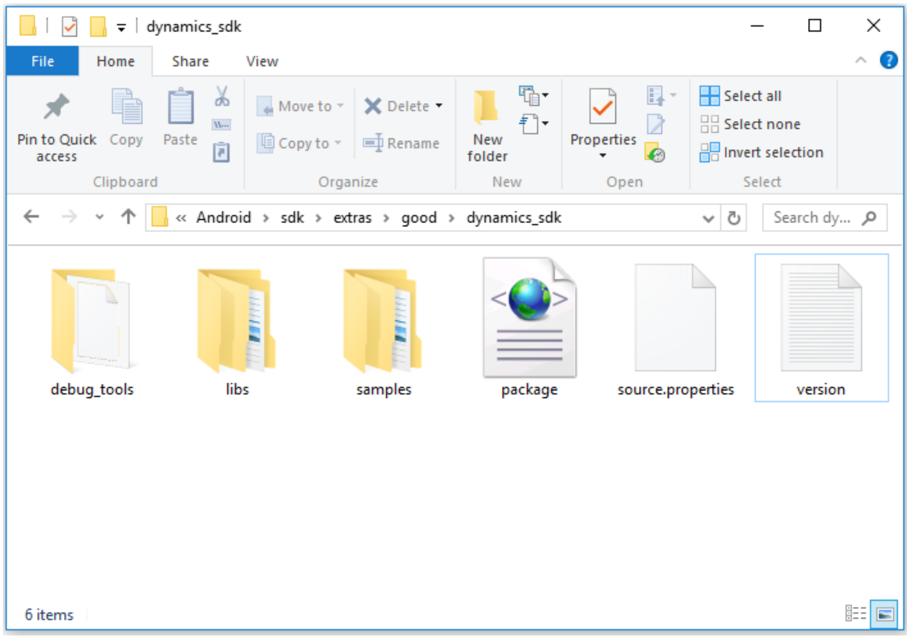
3. Navigate from this directory to the extras sub-directory.

4. If there isn't already a directory named "blackberry" here, create it now. Change to the blackberry sub-directory, which you might just have created.

5. If there is already a directory named "dynamics_sdk" here, then you may already have an installation of the SDK for Android. You can move or delete the current directory, or use a different name for the new directory. If there is no dynamics_sdk directory, then go directly to the next step.

6. Copy the SDK for Android zip file that you downloaded into this good directory and extract its files, or extract the files here directly. This will create a directory structure with a directory named sdk as its root.

7. Rename the new sdk directory to dynamics_sdk.



You have now installed the BlackBerry Dynamics SDK for Android. It is now ready to use in your Android projects.

Run a Sample Application from the BlackBerry Dynamics SDK for Android

Import a Sample Application into Android Studio

You can check your installation of the SDK by running one of the official sample applications.

1. Locate the sample applications directory.

   This will generally be a sub-directory of your Android home directory with the following path:

   Windows: `C:\Users\[username]\AppData\Local\Android\sdk\extras\blackberry\dynamics_sdk\sdk\samples\`

   Mac: `/Users/[username]/Library/Android/sdk/extras/blackberry/dynamics_sdk/sdk/samples/`

   In case you are unsure, the correct directory has a sub-directory for each sample application.

2. Choose a sample application.

For example, you could select the AppKinetics sample, which is in the AppKinetics sub-directory. Do not upgrade the Gradle plug-in version. If you do so you may see the following error.

****If you see an error message regarding the instrumentTest API, change the mention of 'instrumentTest' to 'androidTest' in the gd module's build.gradle file. This is a recently deprecated API and changes to the sample project will be applied in the next release of BlackBerry Dynamics. ****

3. **This step is optional.** Copy the application project sub-directory.

   If you like, you can make a copy of the sub-directory and then complete the remaining steps using your copy. Or, you can work in the sample sub-directory directly.

   You can make the update by editing the **settings.gradle** file. This could be done in Android Studio, or in any text editor.

   The file will define Gradle projects for libraries in the SDK. Each definition includes a path. The paths could be relative to the original location. Any that are must be changed to be absolute, or relative to the new location.

   For example, the AppKinetics sample has the following **settings.gradle** as installed:

```
include 'GDLibrary'

project(':GDLibrary').projectDir =

new File('../../libs/handheld/gd')

include 'GDLibrary_BackupSupport'

project(':GDLibrary_BackupSupport').projectDir =

new File('../../libs/handheld/gd_backup_support')
```

   This could be changed to the following:

```
// Absolute path to the libs directory.

def good_dynamics_libs =

"/Users/jahawkins/AndroidSDK/extras/good/dynamics_sdk/libs"

include 'GDLibrary'

project(':GDLibrary').projectDir =

new File(good_dynamics_libs + '/handheld/gd')

include 'GDLibrary_BackupSupport'

project(':GDLibrary_BackupSupport').projectDir =
new File(good_dynamics_libs + '/handheld/gd_backup_support')
```

4. Import the application into Android Studio.

   If Android Studio is already open and you have another project open, then, in the menu, select File, New, Import Project. Otherwise, open Android Studio now and select from the welcome screen: Import project (Eclipse ADT, Gradle, etc.).

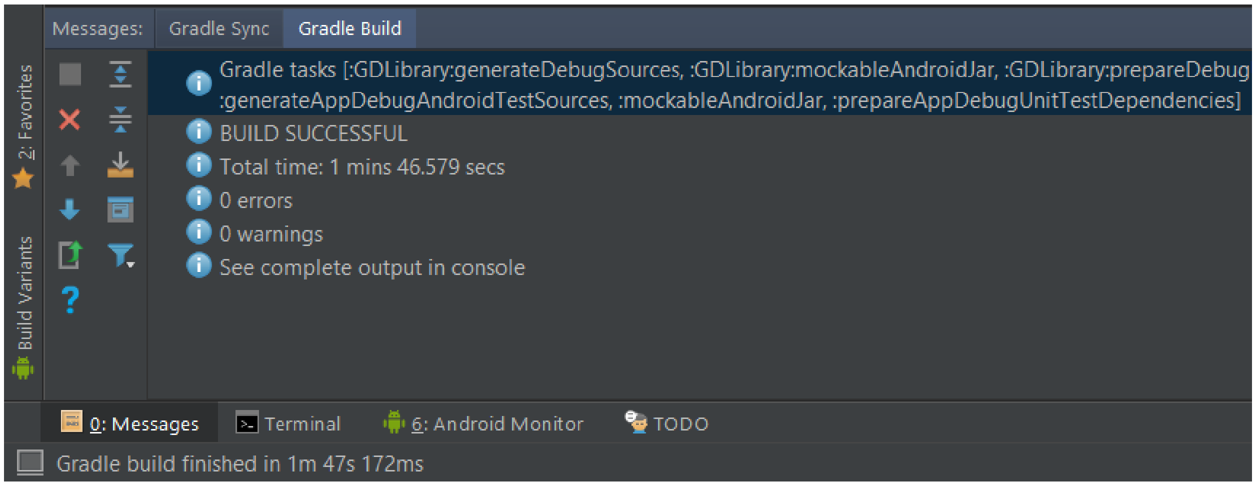
   A file chooser dialog will open. Navigate to and select the **settings.gradle** file in the application directory. If you just made a copy, then select the file you just edited.

   Android Studio will import and build the project.

   At this time, you might be prompted to update the Android Gradle Plugin in the project. **Don't update the plugin.** Instead, select the option: Don't remind me again for this project.

5. Check the build was successful.

   Open the Message tab, by default at the bottom of the Android Studio window. It should show a BUILD SUCCESSFUL message, as shown in the following screen capture.



```
Gradle tasks [:GDLibrary:generateDebugSources, :GDLibrary:mockableAndroidJar, :GDLibrary:prepareDebug
:generateAppDebugAndroidTestSources, :mockableAndroidJar, :prepareAppDebugUnitTestDependencies]
BUILD SUCCESSFUL
Total time: 1 mins 46.579 secs
0 errors
0 warnings
See complete output in console
Gradle build finished in 1m 47s 172ms
```

   Build failures here can sometimes be fixed by closing the project and then importing it again, or by restarting Android Studio.

6. Run the application.

   You can use an emulator, known as an Android virtual device, or a physical device if you have one connected and set up as for developer use.

   You can run the mobile application from the Android Studio menu, by selecting: Run, Run 'AppKinetics' or the name of whichever sample you chose to build.

A dialog on which you Select Deployment Target will open.

If you have no connected devices, and no available virtual devices, you must set one up now. There is an option on the Select Deployment Target dialog to Create New Virtual Device, which you could select.

Select a device and click OK. The mobile application will be installed on the device and then start. If you select a virtual device and it isn't running already, the install might fail if the emulator takes a long time to launch and initialize. In that case, just do the install again after the emulator has finished launching.

7. Test that the application reaches the activation screen.

   The application should show the following:

   a. Prompt for system permission, possibly.
      If the target device is running Android API level 23 or more, and the target SDK version of the application is also 23 or more, then a prompt will be shown for the Android system permission: Make and Manage phone calls.

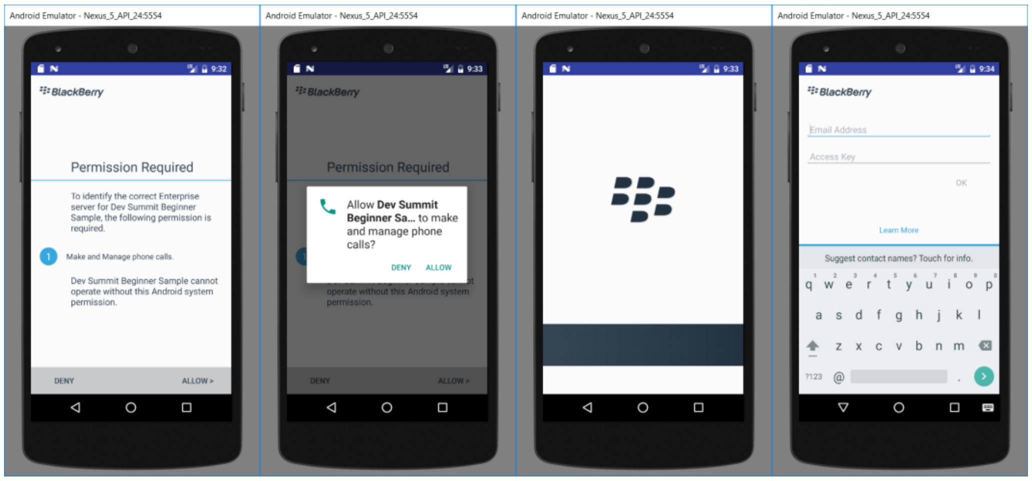
      You must allow the permission, which is required by all BlackBerry Dynamics applications.

      The screen is only shown after a new install, or if the permission has been revoked.

   b. Loading screen.

   c. Prompt for activation credentials.

These are shown in the following screen capture images.


Android Emulator - Nexus_5_API_24:5554

Permission Required

To identify the correct Enterprise server for Dev Summit Beginner Sample, the following permission is required.

1 Make and Manage phone calls.

Dev Summit Beginner Sample cannot operate without this Android system permission.

Allow **Dev Summit Beginner Sa...** to make and manage phone calls?

If you already have a BlackBerry Dynamics application installed on your device, you may see the easy activation screen instead of the fourth screenshot shown above.

This test proves that your BlackBerry Dynamics SDK for Android is installed OK. You can then move on to run the application. Refer to Run an Application in Enterprise Mode below.

Run an Application in Enterprise Mode

Normal enterprise mode is the way in which BlackBerry Dynamics applications are generally run. In this mode, all features of BlackBerry Dynamics can be used.

Devices

Normal enterprise activation can be used:

- On a physical device.
- On a simulated device.

Note that it isn't possible to upgrade from Enterprise Simulation mode to normal enterprise mode. Uninstall the enterprise simulation mode application from the simulated device before installing normal enterprise mode.

Selection

Selection of mode is a build-time configuration.

Select normal enterprise mode in your application by editing the **assets/settings.json** file. Ensure the value for **GDLibraryMode** to **GDEnterprise**, as shown in the following:

```
{

  "GDApplicationID": "com.your.existing.value",

  "GDApplicationVersion": "1.0.0.0",

  "GDLibraryMode": "GDEnterprise"

}
```

Run the Application

Run the application from Android Studio. When it launches:

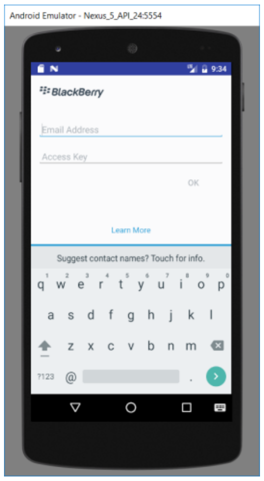
1. A Permission Required screen might be shown.

If the target device is running Android API level 23 or more, and the target SDK version of the application is also 23 or more, then a prompt will be shown for the Android system permission: Make and Manage phone calls.

If the screen is shown, you must select to allow this permission, which is required by all BlackBerry Dynamics applications.

The screen is only shown after a new install, or if the permission has been revoked.

2. The activation screen will be shown:


Android Emulator - Nexus_5_API_24:5554
9:34
BlackBerry
Email Address
Access Key
OK
Learn More
Suggest contact names? Touch for info.

3. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

Session- Secure First BlackBerry Dynamics App

Preparation

Ensure you have completed installing and configuring the BlackBerry Dynamics SDK before starting this section.

This section uses as its subject a sample application from the Android project: Data Layer. The Data Layer sample application includes a handheld application and a wearable application. You can follow the instructions in this workbook to add security to the handheld application only, or to add security to both the handheld and the wearable applications.

It is recommended that you build your application after each step in the adding security process to confirm the implementation, and to keep debugging manageable.

Import an Android Sample Project

The Android project has published several sample applications. You can download Android sample applications directly from GitHub using the Android Studio user interface using either of the following steps:

- In the application menu, select File, New, Import Sample.
- From the welcome screen, click Import an Android code sample.

Either of these actions opens the samples browser. Locate and select the **Data Layer** sample.

You can download the sample to a project location that you specify, for example your own project directory. After downloading the project, you might be prompted to install or update several components. It's OK to make all the prompted changes.

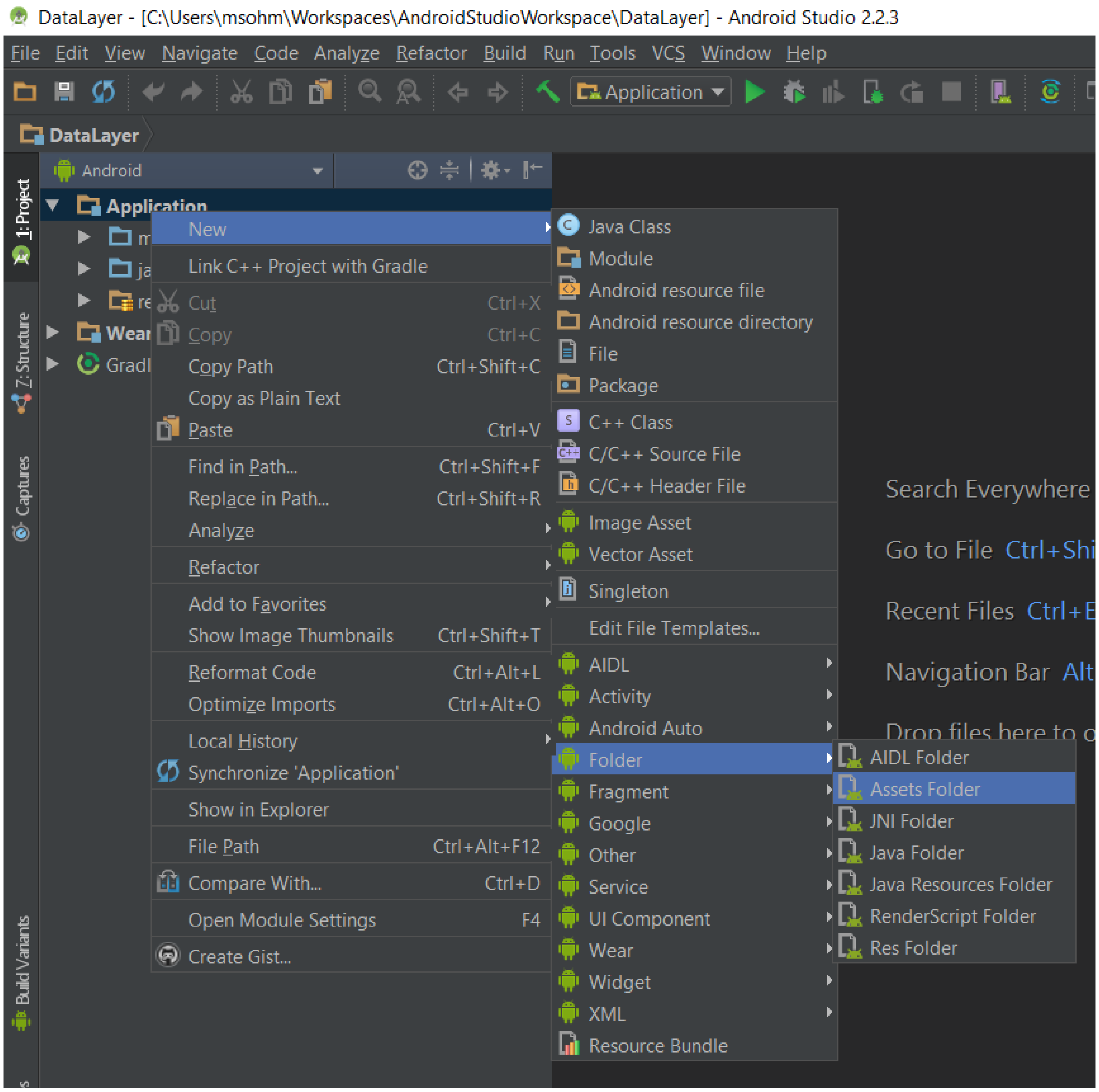
Integrate BlackBerry Dynamics into the Data Layer Sample

Create a Settings File

You will need to create a **settings.json** file within your project. It contains configuration information about your application, such as the library mode, and the BlackBerry Dynamics entitlement identifier and version.

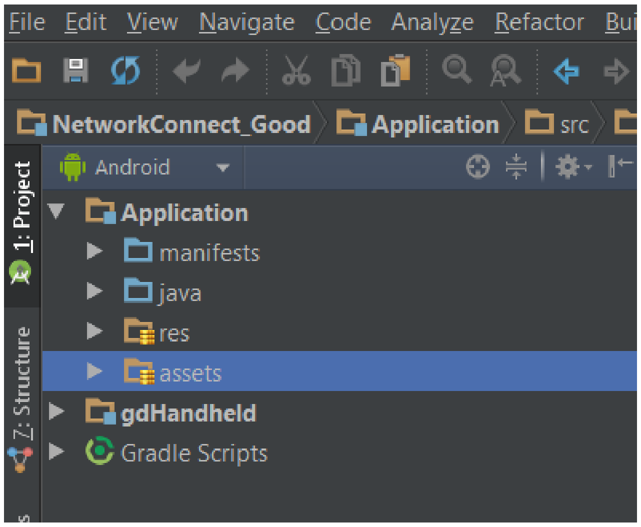
1. If you already have an assets folder in your project, skip to step #5. If not, proceed to step 2.

2. In Android Studio right click the top-level Application package and then select the following in the context menu that appears: New, Folder, Assets Folder. This opens a dialog whose title mentions activity customization, but is a confirmation for where to create "a source root for assets".



3. Leave the default options selected and click Finish. This creates the folder and returns you to the project window.

4. Expand the package in which you just created the folder. The assets folder should appear under the package, as shown in the following screen capture.



5. Right click on the assets folder in the project window and select New, File in the context menu that appears. This opens the New File dialog.

6. Enter **settings.json** as the file name and click OK. This creates an empty **settings.json** file and opens it in the Android Studio editor.

7. Use the following example to build your own **settings.json** file. Modify the example to include your own GDApplicationID and GDApplicationVersion.

GDLibraryMode – Can be set to either GDEnterpriseSimulation or GDEnterprise. **For the BlackBerry Developer Event, use GDEnterprise.**

- You would use GDEnterpriseSimulation if you didn't have a BlackBerry UEM server, BlackBerry UEM Cloud account, Good Control server or a Good Control Cloud account, and are using an Android emulator.
- Use GDEnterprise to test with a BlackBerry UEM server, BlackBerry UEM Cloud account, Good Control server or Good Control Cloud account for production deployments.

A connection to the BlackBerry Dynamics Network Operation Center (NOC) is required in either mode.

GDApplicationID – A unique identifier for your application, which should be the same across all platforms (i.e. Android, iOS, etc…). Conventionally, this value is similar to the application's package name. No capital letters or spaces are allowed. This value is also registered in the BlackBerry UEM server. Use the GDApplicationID you were assigned at the event.

GDApplicationVersion – A version number for the entitlement. It need not match your native application version, which must change in every version of your application. The entitlement version should only be changed when an aspect of your application's BlackBerry Dynamics integration changes, such as when you change the shared services provided by your application. Updating your application entitlement version will result in additional work for both you the developer and all administrators that have deployed your application.

Sample settings.json that uses Enterprise Mode

```
{

  "GDLibraryMode": "GDEnterprise",

  "GDApplicationID": "com.bbdevsummit.devapp###",

  "GDApplicationVersion": "1.0.0.0"

}
```

Replace the ### in the GDApplicationID above with the number assigned to you when you arrived at the BlackBerry Developer event.

Linking the BlackBerry Dynamics Runtime Library

Start by linking the runtime into the application. The runtime will be added as a module. This requires change in several parts of the project.

Add Module

Next you need to add the runtime library to the project as a module. There are a number of ways to add a module to an Android Studio project.

It is recommended that you import the BlackBerry Dynamics SDK using its AAR file.

Open the project in Android Studio and proceed as follows.

1. Expand "Gradle Scripts" within your project tree.
2. Open the build.gradle for your project.
    a. If this file is empty, open the build.grade for the Application Module.
3. Within allprojects, locate the repositories section. If allprojects is not listed, use the repositories at the root level.
4. Add reference to the BD SDK AAR file as shown below.

```
def localProperties = new File(rootDir, "local.properties")
```

```
Properties properties = new Properties()
localProperties.withInputStream { instr ->
    properties.load(instr)
}
def sdkDir = properties.getProperty('sdk.dir')
maven { url sdkDir+'/extras/blackberry/dynamics_sdk/m2repository' }
```

The allprojects section should now look similar to the code below. Note that this sample assumes the BD SDK was installed to the default location. If you installed to a different location, you will need to update the path in the url.

```
allprojects {

    repositories {

        google()

        jcenter()



        def localProperties = new File(rootDir, "local.properties")

        Properties properties = new Properties()

        localProperties.withInputStream { instr ->

            properties.load(instr)

        }

        def sdkDir = properties.getProperty('sdk.dir')

        maven { url 
sdkDir+'/extras/blackberry/dynamics_sdk/m2repository' }

    }

}
```

Add Dependency

Now that the module is part of the project it can be added as a dependency.

1. Open the build.grade file for module app.
2. Add the following lines within the dependencies of your app:

```
implementation
'com.blackberry.blackberrydynamics:android_handheld_platform:+'

implementation
'com.blackberry.blackberrydynamics:android_handheld_backup_support:+'

implementation
'com.blackberry.blackberrydynamics:android_handheld_wearable_support:+'
```

Note: Depending on your application's needs, you may need to add further dependencies here. For example, if your application supports wearables, you will need to specify the appropriate BlackBerry Dynamics dependencies. For a full list of refer to the BlackBerry Dynamics Development Guide.

3. Press "Sync Now", shown in the upper right of Android Studio.

The BlackBerry Dynamics runtime is now linked to the handheld application. You are ready to begin the code changes to your application to integrate it with the runtime.

Ensure that the Application API Level is Compatible

The initial selections for Android API level in the application might be incompatible with the BlackBerry Dynamics SDK for Android. Check and change as follows.

1. Open the project in Android Studio.
2. Open the **AndroidManifest.xml** file.

   In the Data Layer sample application, the file is in the manifests folder in the Application package. You can open it by double-clicking.
3. If there is a uses-sdk declaration, then delete it.

   In the Data Layer sample application, there is a declaration that should be deleted. It looks like this:

   ```
   <uses-sdk android:minSdkVersion="18"

               android:targetSdkVersion="23" />
   ```
4. Open the **build.gradle** file for the application module.

In the Data Layer sample application, this is the Application module. The file is listed under the special Gradle Scripts folder in the Android view. You can open it by double-clicking.

5. Locate the android section, and within it the defaultConfig sub-section.

6. Fix the SDK version settings.

   This sub-section might contain several settings. The ones that relate to compatibility of the API are:

   - minSdkVersion
   - targetSdkVersion

   In the Data Layer sample application, the defaultConfig section looks like this:

```
defaultConfig {

      minSdkVersion 21

      targetSdkVersion 27

}
```

   BlackBerry Dynamics supports Android SDK version 21 and higher. The DataLayer sample specifies a minimum SDK version of 18. Update the minimum SDK version using the following steps. After updating, it will look like this:

```
defaultConfig {

      minSdkVersion 21

      targetSdkVersion 27

}
```

7. Press “Sync Now”.

Connect to the BlackBerry Dynamics Platform

There are several variants in the API for connecting to the platform. This workbook gives instructions for the variant that is easiest to add to the Data Layer sample application. Other variants are documented in the API Reference, in the GDAndroid class reference.

Add Activity Monitoring and Indirect Authorization Initiation

Connection to the platform requires authorization processing and requires that every activity in the application is monitored by the BlackBerry Dynamics runtime. Both requirements can be met by using the activityInit API as follows.

1. Open the project in Android Studio.
2. Open the Java source file for an Activity class. All Activity classes will be listed in the **AndroidManifest.xml** file.

   The Data Layer sample handheld application, which is represented by the Application module, has only a single class and it is an Activity class: **MainActivity**.
3. Locate the **onCreate** method in the source file. Within the method, locate the call to **super.onCreate** which is typically at the start of the method. After the **super.onCreate** call, insert a call to the **activityInit()** method of the **GDAndroid** singleton instance, which is accessed through its **getInstance()** method. Pass a reference to the current object as the parameter.

   The start of the **onCreate** method should now look something like this:

   ```
   @Override

   public void onCreate(Bundle savedInstanceState) {

         super.onCreate(savedInstanceState);


         GDAndroid.getInstance().activityInit(this);

         // Line above was just added.


         LOGD(TAG, "onCreate");
   ```

   Inserting the indicated line is the only manual change made at this time. The editor should automatically insert an import statement for the **com.good.gd.GDAndroid** class. If it doesn't, then add it manually.
4. Repeat the above steps for every Activity class in the application. There is only one in the DataLayer sample.

This completes the addition of activity monitoring and indirect authorization initiation.

Implement Authorization Life Cycle Handling

The application must handle transitions in the BlackBerry Dynamics authorization life cycle. Every activity can be its own handler for these transitions. This can be implemented as follows.

1. Open the project in Android Studio.
2. Open the Java source file for an Activity class. All Activity classes will be listed in the **AndroidManifest.xml** file.

   The Data Layer sample handheld application has only a single class and it is an Activity class, **MainActivity**.
3. Locate the class declaration in the source file. It will be near the top, after the import statements.
4. Add to the declaration that the class implements the **GDStateListener** interface.

   If the class doesn't already implement any interfaces then you will have to add the **implements** keyword, and then the name of the interface. Otherwise, you will have to add a comma separator, and then the name of the interface. Example of a changed declaration that did not already implement any interfaces:

   ```
   public final class ContactManager extends Activity implements
   GDStateListener

   {
   ```

   The text highlighted in bold in the above is what would be inserted.

   Example of a changed declaration that already implemented one or more interfaces:

   ```
   public class MainActivity extends Activity implements

        CapabilityApi.CapabilityListener,MessageApi.MessageListener,

        DataApi.DataListener, ConnectionCallbacks,

        OnConnectionFailedListener, GDStateListener {
   ```

   The text highlighted in bold in the above is what was inserted.

   The editor might or might not automatically insert an import statement for the com.good.gd.GDStateListener interface. If it doesn't, then add it manually.
5. Add a minimal implementation of the interface that was just declared.

   The Android Studio editor facilitates this addition. Click on the Code menu and choose "Implement Methods…".

All the methods will be selected by default. Click OK to add minimal implementations.

Code like the following will have been inserted into the source file:

```
@Override
public void onAuthorized() {

}
@Override
public void onLocked() {

}
@Override
public void onWiped() {

}
@Override
public void onUpdateConfig(Map<String, Object> map) {

}
@Override
public void onUpdatePolicy(Map<String, Object> map) {

}
@Override
public void onUpdateServices() {

}
@Override
```

```
public void onUpdateDataPlan() {


}

@Override

public void onUpdateEntitlements() {


}
```

6. Repeat the above steps for every Activity class in the application. If you are using the DataLayer sample, there is only one activity.

This completes minimal life cycle handling implementation in the application.

Switch off Android Auto Backup

The Android operating system has a feature that automatically backs up application data from the mobile device to an Internet service operated by Google. This feature, Auto Backup, is configurable at build-time. Android applications that target API level 23 or later will be configured, by default, to back up all application data automatically.

The default Auto Backup configuration interferes with the normal secure functioning of BlackBerry Dynamics and can't be used in BlackBerry Dynamics applications. Instead, applications must include an explicit Auto Backup configuration that is compatible with normal secure functioning or must switch off the Auto Backup feature.

In this workbook, Android Auto Backup will be switched off, as follows.

1. Open the project in Android Studio.
2. Open the **AndroidManifest.xml** file.

   In the Data Layer sample application, the file is in the manifests folder in the Application package. You can open it by double-clicking.

3. Locate the opening tag of the application declaration.

   In the Data Layer sample application, it looks like this:

```
<application

        android:allowBackup="true"
```

```
    android:icon="@drawable/ic_launcher"

    android:label="@string/app_name">
```

4. Ensure that Android Auto Backup is switched off.

   If there is no **android:allowBackup** attribute, then insert one and set the value to **"false"**. If the attribute is already present, then change its value.

   After modification, the opening tag of the Data Layer application declaration looks like this:

   ```
   <application

       android:allowBackup="false"

       android:icon="@drawable/ic_launcher"

       android:label="@string/app_name">
   ```

Running the Application

You can now run the application.

Warning If you are running the Data Layer sample application from Android Studio then you may have to select the handheld application.

Select the handheld application as follows:

1. In the application menu, select Run, Debug... (or Run, Run...)
2. In the floating menu that appears, select Application.
3. If another floating menu appears, select Debug or Run again.

Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

After entering a BlackBerry Dynamics password you should see the user interface for the Android sample you have chosen.

Session- Principal BlackBerry Dynamics Features (Secure Communication)

These exercises don't depend on completion of any other exercises. These exercises can't be done with an application running in Enterprise Simulation mode.

Exercise One: Send an HTTP REST Request to a Configured Server

This exercise doesn't depend on completion of any other exercises.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project uses Google Android APIs that require modification to use the BlackBerry Dynamics APIs. It also has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

UEM Administration Actions

The sample requires a BlackBerry Dynamics Connectivity profile be configured in the BlackBerry UEM Administration console. This allows the administrator - which is you today – to remotely configure the server and port the application connects to. In production setting, this takes the burden of configuring the application out of an end user's hands.

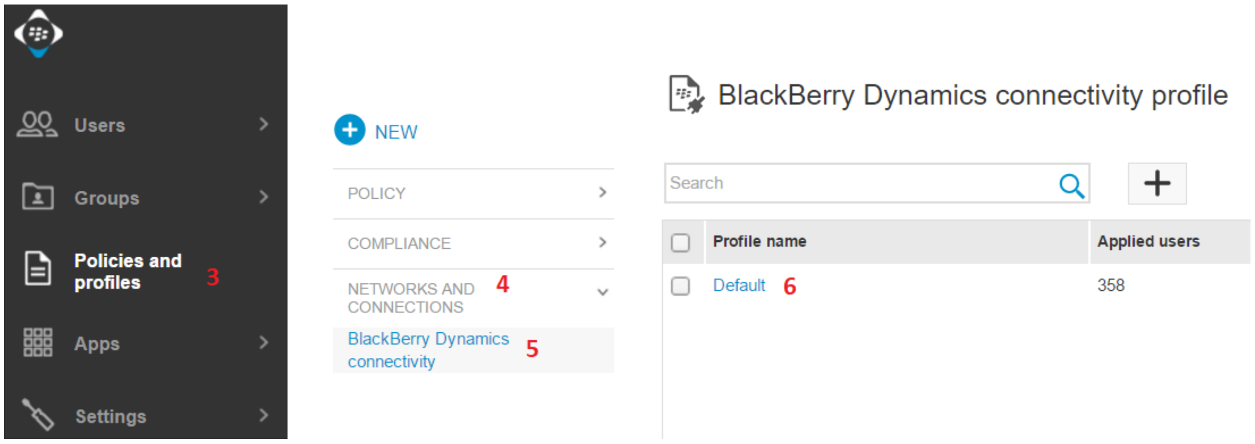
Perform the following steps to configure a BlackBerry Dynamics Connectivity profile you this sample application.

1. Within BlackBerry Access, navigate to: https://uem02.blackberrydevsummit.com/admin/ If you do not yet have BlackBerry Access installed, refer to Appendix A. Alternatively, you can access the same site using https://uem.blackberrydevsummit.com:10443/admin/ in a regular browser if connections over port 10443 are allowed in your environment.
2. Log in using the following credentials:
    - User Name: DevUser###

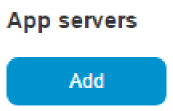
- Replace ### with the number assigned to you when you arrived at the BlackBerry Developer event.

- Password: DevUser!2018
- Domain: blackberrydev

3. Click on Policies and profiles.
4. Click on NETWORK AND CONNECTIONS.
5. Click on BlackBerry Dynamics connectivity.
6. Click on the Default BlackBerry Dynamics Connectivity profile. The screenshot below illustrates steps 3, 4, 5 and 6.

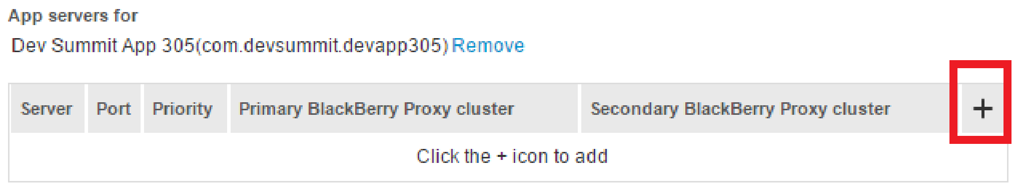


7. Click on the Pencil icon near the top right are of the screen to edit the connectivity profile.
8. Scroll to the bottom of the page and click on the Add button under App Servers.



9. In the window that appears, search for “Dev Summit App ###” where ### is the number assigned to you when you arrived at the BlackBerry Developer event.
10. Click on your app name in the search results and click the Save button.

11. Locate the "Dev Summit App ###" entry that was added to the profile screen and click the + icon.



App servers for

Dev Summit App 305(com.devsummit.devapp305) Remove

Server	Port	Priority	Primary BlackBerry Proxy cluster	Secondary BlackBerry Proxy cluster	+
Click the + icon to add					

12. Add the following app server information:
    - Server: anonymous.blackberrydevsummit.com
    - Port: 80
    - Primary BlackBerry Proxy cluster: First (select only option in list)
13. Click the Save button to add the app server.
14. Click the Save button to save the connectivity profile.

You have now configured the app server for the BlackBerry Dynamics connectivity profile that will be delivered to your application.

Programming Actions

Open the BeginnerSample_Incomplete-Android project and complete the steps below.

Note that this sample assumes the BD SDK was installed to the default location. **If you installed to a non-default location,** you will need to update the path in the maven url in the build.gradle file for the project as shown in bold below.

```
def localProperties = new File(rootDir, "local.properties")

Properties properties = new Properties()

localProperties.withInputStream { instr ->

    properties.load(instr)

}

def sdkDir = properties.getProperty('sdk.dir')

maven { url sdkDir+'/extras/blackberry/dynamics_sdk/m2repository' }
```

1. Open the **settings.json** file and update the **GDApplicationID**, replacing ### with the number assigned to you at the BlackBerry Developer event.
2. Retrieve the server details from the management console by completing the **B04-Exercise One** TODO section of **MainActivity.java**.

   For you to complete this task, there must be a server address and port number configured for your application in the management console. If you attempt the REST connection before configuring the server and port, a NullPointerException will be thrown.

   Use the **getApplicationConfig** method in the **GDAndroid** to retrieve a collection of configuration settings.
3. Send a REST request to the atomic REST service using BlackBerry Dynamics APIs by completing the **B04-Exercise One** TODO sections of **HttpFragment.java**. This web service accepts an atomic number as input and returns information about that element.

   Use the com.good.gd.net.GDHttpClient class.
4. Follow the steps in "Appendix B – Provision an Application in the BlackBerry Developer Lab" to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.
5. To test the application, enter an atomic number (1-118) and press Go.

Exercise Two: Authenticate the User at the Enterprise Server

This exercise builds on the previous exercise.

URLs internal to the BlackBerry Developer lab that can be used for this exercise:

Kerberos Constrained Delegation

http://kcd.blackberrydevsummit.com:80

Kerberos Log-in Authentication

http://kerberos.blackberrydevsummit.com:80

Anonymous

http://anonymous.blackberrydevsummit.com:80

Basic

http://basic.blackberrydevsummit.com:80

You can change the URL used in this exercise by removing the server configured in the BlackBerry Dynamics Connectivity profile and adding a new one. Do not include HTTP or HTTPS when adding to BlackBerry UEM. Refer to “UEM Administration Actions” in the previous exercise for steps to modify the BlackBerry Dynamics Connectivity profile. Ensure you click all Save buttons in the BlackBerry UEM administration console so that the changes are delivered to your application.

Code the following.

1. Receive a response that authorization is required.

   Change the configured application server to be one of the above URLs that:

   - Requires authorization of the end user. We recommend trying Kerberos Constrained Delegation (KCD) first and then Kerberos log-in authentication.
   - Observe that no changes to the application were required for KCD authentication. KCD authentication is handled by the BlackBerry Dynamics platform.

   Your code from the previous exercise should work for this step, without modification. The expected response is an HTTP 401 Unauthorized for all authenticated URLs except KCD.

2. Switch to the Kerberos URL and send user credentials. Locate the **B04-Exercise Two** TODO items in **HttpFragment.java**. Change your application to send a second HTTP request that includes credentials.

   Use the username and password supplied to you at the BlackBerry Developer event.

   - Create an object to hold the credentials. The object will be of the class: **com.good.gd.apachehttp.auth.Kerberos5Credentials**
   - Use the **GDHttpClient.getCredentialsProvider** method to obtain a **CredentialsProvider** instance. Then use its **setCredentials** method to put the credentials in place in the client object. Use **AuthScope.ANY** in the **setCredentials** method.

3. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

Additional Exercise – Open A Socket Connection

This exercise doesn’t depend on completion of any other exercises.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project uses Google Android APIs that require modification to use the BlackBerry Dynamics APIs. It also has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

Open the BeginnerSample_Incomplete-Android project and complete the steps below.

Like exercise one, send an HTTP request to a server, but open a socket connection instead of sending an HTTP request by completing the TODO sections of **SocketFragement.java**.

Once complete, follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

Session- Principal BlackBerry Dynamics Features (Secure Storage)

These exercises don't depend on completion of any other exercises. These exercises can be done with an application running in Enterprise Simulation mode, or in normal enterprise mode.

Exercise One: Secure Store File

This exercise doesn't depend on completion of any other exercises.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project uses standard Android APIs that require modification to use the

BlackBerry Dynamics APIs. It also has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

Open the BeginnerSample_Incomplete-Android project in Android Studio (File, Open menu) and complete the steps below.

Note that this sample assumes the BD SDK was installed to the default location. **If you installed to a non-default location,** you will need to update the path in the maven url in the build.gradle file for the project as shown in bold below.

```
def localProperties = new File(rootDir, "local.properties")

Properties properties = new Properties()

localProperties.withInputStream { instr ->

    properties.load(instr)

}

def sdkDir = properties.getProperty('sdk.dir')

maven { url sdkDir+'/extras/blackberry/dynamics_sdk/m2repository' }
```

1. Open the **settings.json** file and update the **GDApplicationID**, replacing ### with the number assigned to you at the BlackBerry Developer event.

2. Open **FileFragment.java**.

3. Complete the TODO areas in **FileFragment.java** with the **B05-Exercise One** heading.

4. Check that the data is encrypted.

    a. Get the native path of the file by using the **GDFileSystem.getAbsoluteEncryptedPath** method.

    b. Use the Android Debug Bridge (adb) tool to get access to the native file system of the device. Run a command like the following:

    ```
    adb exec-out run-as "com.example.blackberry.beginnersample"
    cat -vte
    "/data/user/0/com.example.blackberry.beginnersample/app_dat
    a/.AContainer/AsXe2amIdU49b4kxLwgi8OY="
    ```

5. Follow the steps in "Appendix B – Provision an Application in the BlackBerry Developer Lab" to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

Exercise Two: Secure Store Directory

Code the following directory management operations. This exercise builds

on the previous exercise.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project uses standard Android APIs that require modification to use the BlackBerry Dynamics APIs. It also has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

Open the BeginnerSample_Incomplete-Android project in Android Studio (File, Open menu) and complete the steps below.

1. Open the **settings.json** file and update the **GDApplicationID**, replacing ### with the number assigned to you at the BlackBerry Developer event.

2. Open **FileFragment.java**.
3. Complete the TODO areas in **FileFragment.java** with the **B05-Exercise Two** heading to:
    a. Move the file created in the previous exercise into the directory.
    b. Rename the file.
    c. Delete the file.
    d. Delete the directory.
4. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

Exercise Three: Secure SQL Database

This exercise doesn’t depend on completion of any other exercises.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project uses standard Android APIs that require modification to use the BlackBerry Dynamics APIs. It also has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

Open the BeginnerSample_Incomplete-Android project in Android Studio (File, Open menu) and complete the steps below.

1. Open the **settings.json** file and update the **GDApplicationID**, replacing ### with the number assigned to you at the BlackBerry Developer event.
2. Open **ColorDbHelper.java** and **SqlFragment.java**.
3. Complete the **B05-Exercise Three** TODO areas in the Java code of **ColorDbHelper.java** and **SqlFragment.java**.
    - Use classes in the com.good.gd.database package for the database actions.
    - Use the com.good.gd.file.File class to build the path of the database file.

4. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

If you have extra time, try inducing an error in each step and printing the error message.

Additional Exercise: Secure Shared Preferences for Android

Get and set values in an Android SharedPreferences file in the secure store.

1. Use the **GDAndroid.getGDSharedPreferences** method to obtain a **SharedPreferences** object.
2. Use the native Android SharedPreference API to get and set values in standard way.
3. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.

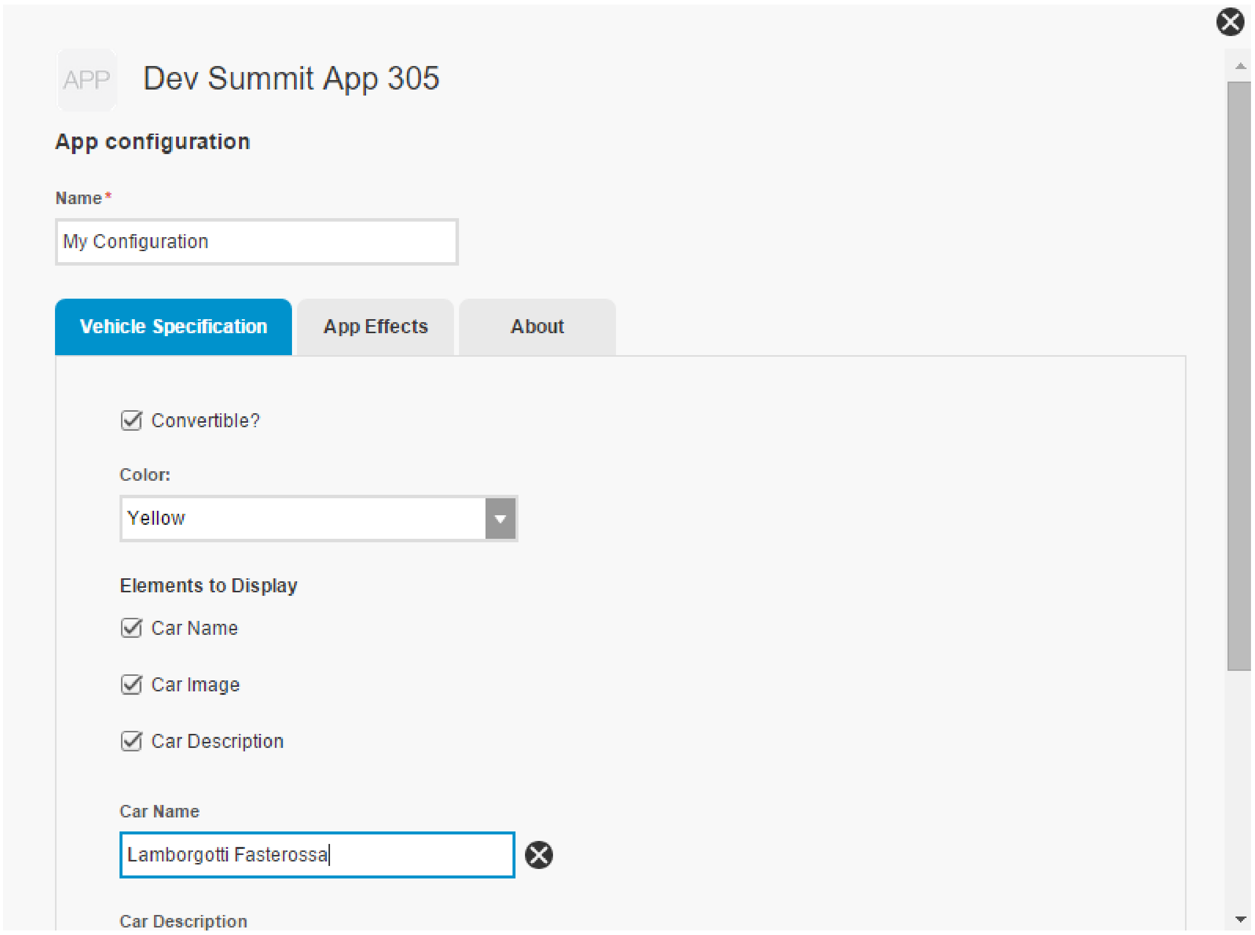
Session- Introduction to BlackBerry Dynamics Value Added Features

These exercises can't be done with an application running in Enterprise Simulation mode.

Application Configuration Exercises

This section uses the term Application Configurations, which is what is now used in BlackBerry UEM. In Good Control the same concept is known as Application Policies. Both terms may appear in online documentation and refer to the same thing.

By using a Custom Application Configuration, you provide an administrator with a user interface in the BlackBerry Dynamics management console to configure your application. An example is shown in the screenshot below. The Application Configuration definition that is loaded into the console is defined in an XML file. When the configuration settings are changed, the new values are pushed to all applications activated against the BlackBerry Dynamics deployment. The running applications are notified when the updated configuration settings arrive.



Name*: My Configuration

☑ Convertible?

Color: Yellow

Elements to Display

☑ Car Name

☑ Car Image

☑ Car Description

Car Name: Lamborgotti Fasterossa

Car Description

Screen capture of an Application Configuration in the UEM management console.

Exercise One: Upload Custom App Configuration to BlackBerry UEM

This exercise doesn’t depend on completion of any other exercises.

Sample projects have been created for this exercise that contain a complete and incomplete implementation called BeginnerSample_Complete-Android and BeginnerSample_Incomplete-Android. The complete sample can be used as a reference of the complete solution. The incomplete project has missing sections you will need to code from scratch. Access them using link on the following page:

http://www.blackberrydevsummit.com

If prompted to log in, use your BlackBerry account login credentials.

Open the BeginnerSample_Incomplete-Android project in Android Studio (File, Open menu) and complete the steps below.

Note that this sample assumes the BD SDK was installed to the default location. **If you installed to a non-default location,** you will need to update the path in the maven url in the build.gradle file for the project as shown in bold below.

```
def localProperties = new File(rootDir, "local.properties")

Properties properties = new Properties()

localProperties.withInputStream { instr ->

    properties.load(instr)

}

def sdkDir = properties.getProperty('sdk.dir')

maven { url sdkDir+'/extras/blackberry/dynamics_sdk/m2repository' }
```

Upload the custom App Configuration to the BlackBerry UEM Server

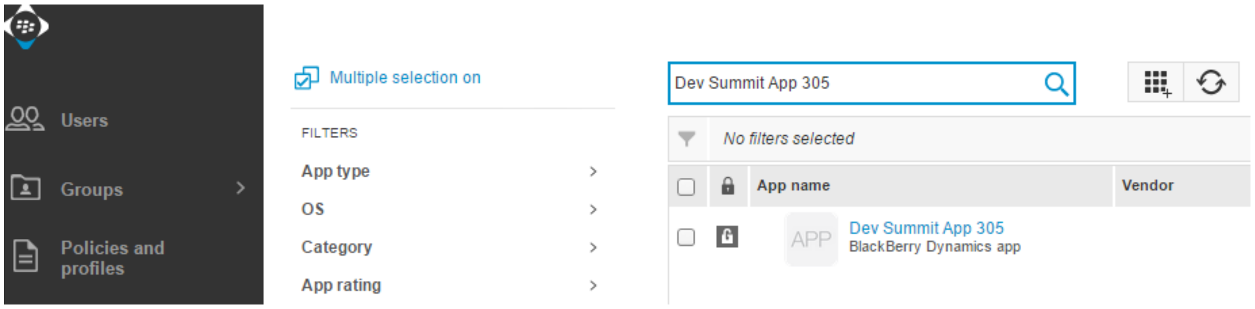
1. Open the app configuration definition file **The Configurator App Configuration.xml** in a text editor. This file is located in the root of the sample directory. Observe how the configuration is defined.
2. Within BlackBerry Access, navigate to: https://uem02.blackberrydevsummit.com/admin/ If you do not yet have BlackBerry Access installed, refer to Appendix A. Alternatively,

you can access the same site using https://uem.blackberrydevsummit.com:10443/admin/ in a regular browser if connections over port 10443 are allowed in your environment.

3. Log in using the following credentials:
    - User Name: DevUser###
        - Replace ### with the number assigned to you when you arrived at the BlackBerry Developer event.
    - Password: DevUser!2018
    - Domain: blackberrydev

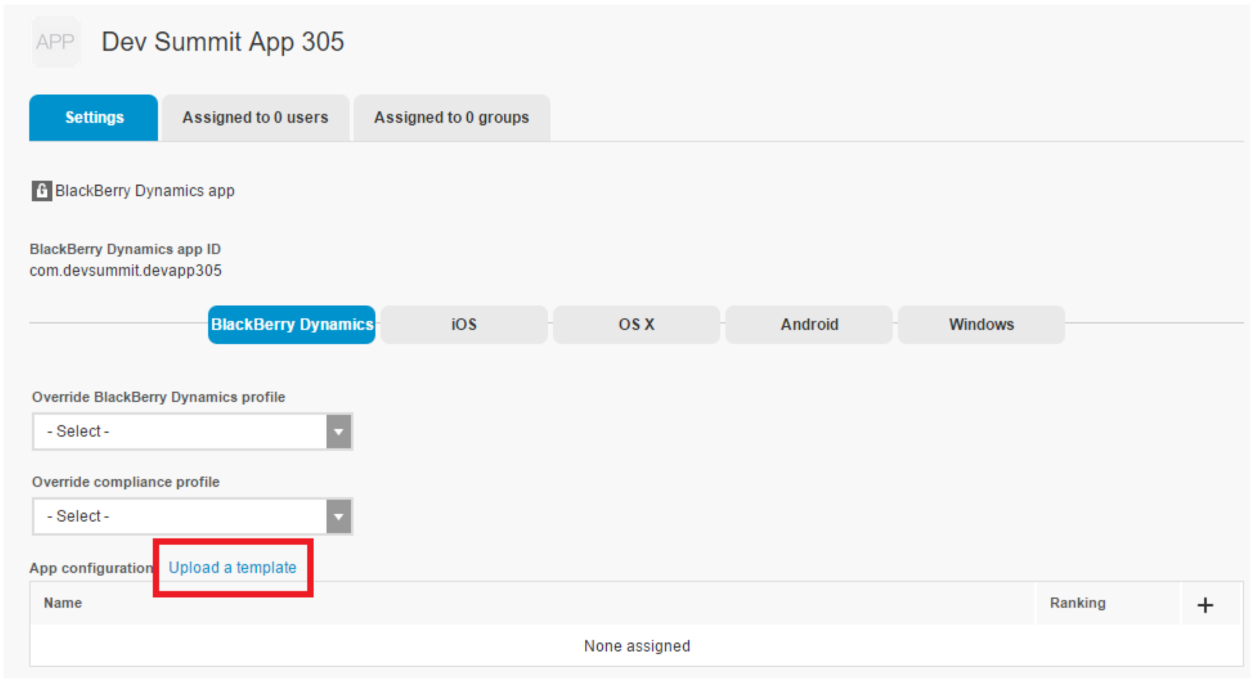
4. Click on Apps.

5. In the search bar, enter “Dev Summit App **###”,** replacing ### with the number assigned to you.



6. Click on your assigned application.

7. Click on “Upload a template” under App Configuration.


Dev Summit App 305

BlackBerry Dynamics app

BlackBerry Dynamics app ID
com.devsummit.devapp305

Override BlackBerry Dynamics profile
- Select -

Override compliance profile
- Select -

App configuration Upload a template

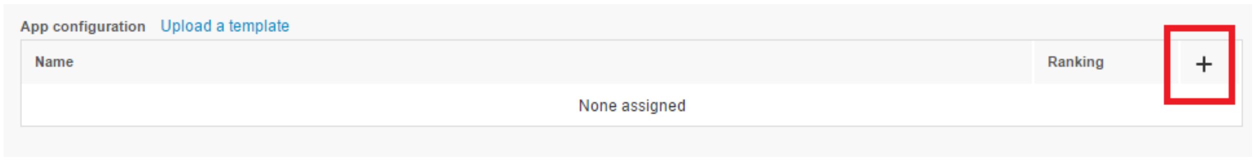
Name	Ranking
None assigned	

8. Click on the Browse button and choose **The Configurator App Configuration.xml** located in the root directory of the sample.

9. Click the Save button.

10. Read the important message displayed and then press OK.

11. Click on the + under App Configuration.


App configuration Upload a template

Name	Ranking
None assigned	

12. Enter an app configuration name.

13. Modify some configuration values.

14. Click Save.

15. If a default configuration appears under App Configurations, delete it by pressing the X at the right of its row.

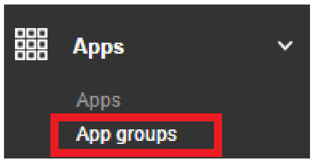
16. Click Save.

Apply the Application Configuration to Your Account

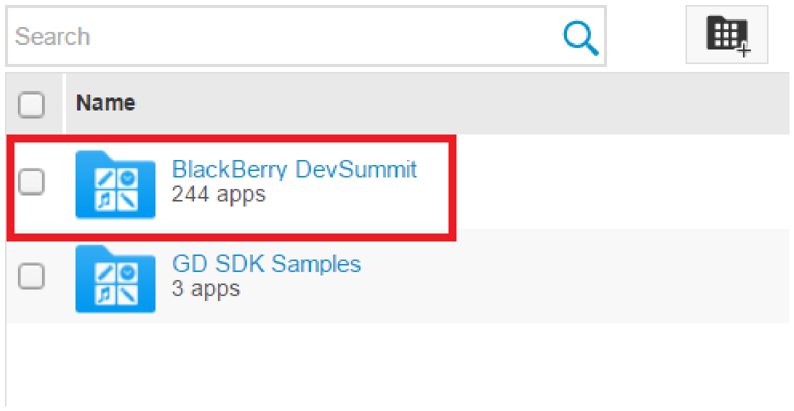
In a typical production deployment, application configurations are applied to groups of users. For this lab, each attendee has a unique application for their account, so the configuration is applied on an individual basis. These steps only need to be done once. You do not need to repeat these steps after changing application configuration settings (from the previous section).

Complete the following steps to apply the application configuration you just created to your account.

1. Skip step 2 and 3 if you are already logged in.
2. Within BlackBerry Access, navigate to: https://uem02.blackberrydevsummit.com/admin/ If you do not yet have BlackBerry Access installed, refer to Appendix A. Alternatively, you can access the same site using https://uem.blackberrydevsummit.com:10443/admin/ in a regular browser if connections over port 10443 are allowed in your environment.
3. Log in using the following credentials:
   - User Name: DevUser###
     - Replace ### with the number assigned to you when you arrived at the BlackBerry Developer event.
   - Password: DevUser!2018
   - Domain: blackberrydev
4. Click on Apps.
5. Click on App Groups.

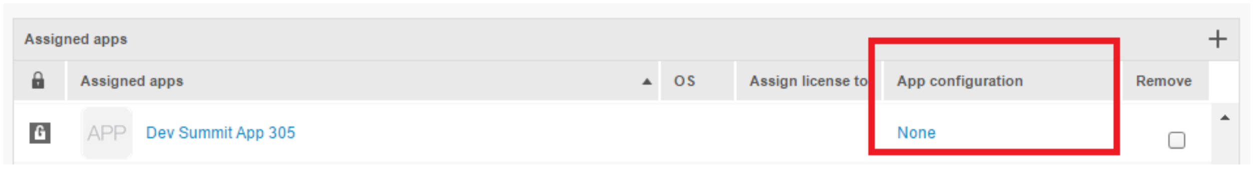


6. Click on “BlackBerry DevSummit”.



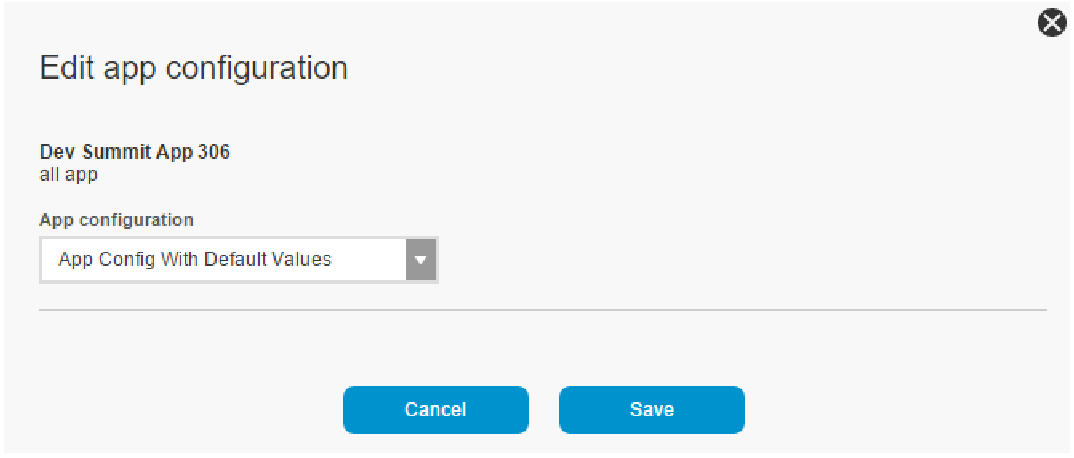
7. Scroll down and locate your Dev Summit App ###**,** where ### is the number you were assigned.

8. Click on None under App configuration.



Assigned apps	OS	Assign license to	App configuration	Remove
APP Dev Summit App 305			None	☐

9. Choose the app configuration you created and click Save.



Edit app configuration

Dev Summit App 306
all app

App configuration: App Config With Default Values

10. Click Save.

11. Click Save.

Exercise Two: Retrieve Initial Custom App Configuration Settings

This exercise builds on the previous exercise.

Open the BeginnerSample_Incomplete-Android project in Android Studio (File, Open menu) and complete the steps below.

1. Open the **settings.json** file and update the **GDApplicationID**, replacing ### with the number assigned to you at the BlackBerry Developer event.
2. Retrieve the configuration settings in your application by completing the **B06-Exercise Two** TODO sections in **MainActivity.java**.
    - Write code to retrieve the configuration settings that uses **getApplicationPolicy** within the **GDAndroid** class.
3. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.
4. Change a configuration setting in the UEM management console and restart the application to observe the changes. It may take a minute or two for the changed setting to become available in the mobile application. You can refer to the steps in the “Upload the custom App Configuration to the BlackBerry UEM Server” section of the previous exercise as a refresher to see where to make these changes. Note that you do not have to upload the XML file again.

Exercise Three: Be Notified of Changes to Custom App Configuration Settings

This exercise builds on the previous exercise.

Code the following.

1. Code a listener for changes to Application Configuration settings by completing the TODO sections with the **B06-Exercise Three App Configuration** heading in **MainActivity.java**.
2. Follow the steps in “Appendix B – Provision an Application in the BlackBerry Developer Lab” to activate the application. Note that this step is only required the first time you run this sample and does not need to be repeated as you update it.
3. Change a configuration setting in the management console. Refer to the previous exercise for instructions on how to do this in the BlackBerry Dynamics management console.

   Check that your listener is invoked. It might take a minute or two.

Exercise Four: Change the App Configuration Definition

This exercise builds on the earlier exercise to retrieve custom configuration settings.

1. Open the **The Configurator App Configuration.xml** configuration definition file.

   Make a copy of the Application Configuration file that you installed in the management console in the earlier exercise. Open the file in a suitable editor. The file format is XML.

   The format is documented in the online API References of the SDKs here: https://developers.blackberry.com/us/en/resources/api-reference/dynamics-android.html#/content/dam/developer-blackberry-com/api-reference/dynamics/dynamics-android/_app_policies.html

2. Add some more settings.

   Refer to the file itself, and to the API Reference, for details of the format.

   Every setting you add will have to be inserted in two places.

   - Add a **<setting>** element to define the setting. The definition includes the data type and how it is presented and labelled in the configuration editor user interface of the management console.

   - Add a **<pe>** element in the **<pview>** section of the definition to define where the setting appears in the structure.

   Tip: The easiest types of setting to add are the hidden and checkbox data type.

3. Install your modified definition in the management console. Follow the same instructions as you did for the initial installation.

4. Check that the modifications appear in the management console. Follow the same instructions as you did for changing configuration settings.

   It might take multiple attempts to get the definition working as you want it. It's a good idea to make an obvious visible change prior to every update installation. For example, the sample definition has an About tab that contains fixed text. Change the text before the upload and then check that the change is reflected in the configuration editor in the management console.

   If you don't see your changes you likely have a syntax error that is causing the updated configuration file to fail to load. In this scenario, the server continues to use the previous working version. You might find it useful to use an online XML validator to check your syntax against the Application Configuration XSD file found here:

https://developers.blackberry.com/us/en/resources/api-reference/dynamics-android.html#/content/dam/developer-blackberry-com/api-reference/dynamics/dynamics-android/_app_policies.html

5. Adapt **AppConfiguration.java** so that it parses the new configuration settings you added and then implement them so they appear in the application.

   Retrieve the configuration settings as before, in exercises three and four. Check that the extra settings are present.

Appendix A – Installing BlackBerry Access

The BlackBerry Developer Lab has been configured similar to a corporate network environment, in that the application servers used in the exercises are not publicly available over the internet. You will use the BlackBerry Dynamics secure connectivity and a BlackBerry UEM Server to access them.

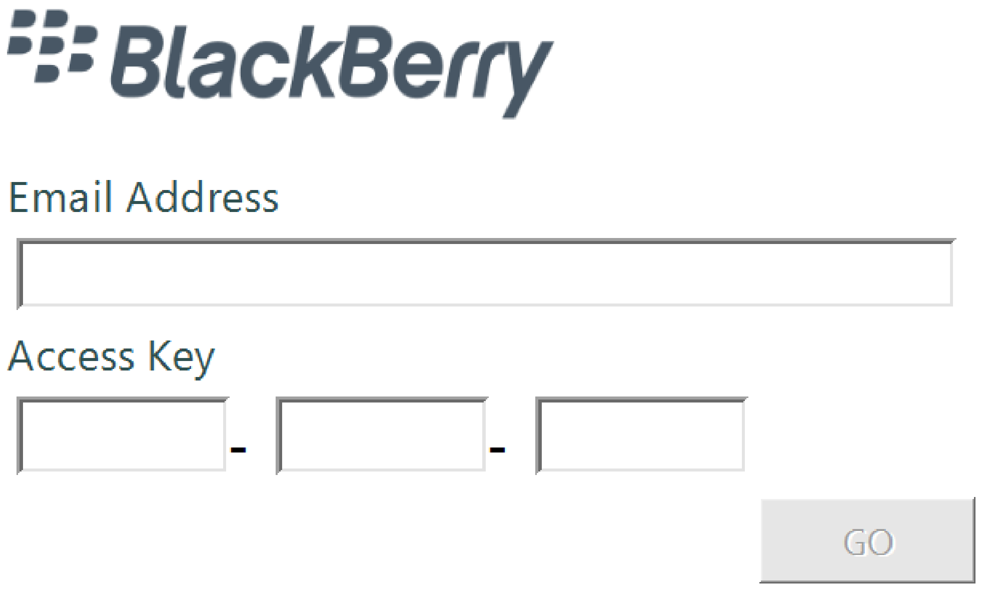
To access these servers and the BlackBerry UEM administration console, BlackBerry Access will be used.

Download BlackBerry Access

Use the following link to download BlackBerry Access:
https://ca.blackberry.com/support/business/enterpriseapps/blackberry-mobility-suites

Installing & Configuring BlackBerry Access

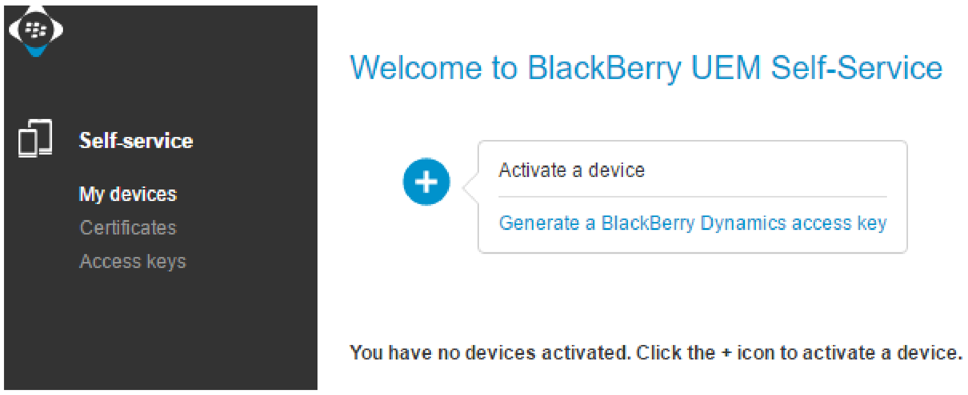
1. Install BlackBerry Access using the installation file you downloaded in the previous step. On first run it will prompt you for an email address and access key.
2. Enter an email address of DevUser###@BlackBerryDevSummit.com, replacing ### with the number assigned to you when you arrived at the BlackBerry Developer event.



3. Navigate to the BlackBerry UEM Self Service Portal to generate an access key: https://uem.blackberrydevsummit.com:10443/index.jsp

4. Log in using the following credentials:
   - Username: DevUser###
     - Replace ### with the number assigned to you when you arrived at the BlackBerry Developer event.
   - Password: DevUser!2018
   - Domain: BlackBerryDev

5. Once logged in, click on the + button and choose “Generate a BlackBerry Dynamics access key”.

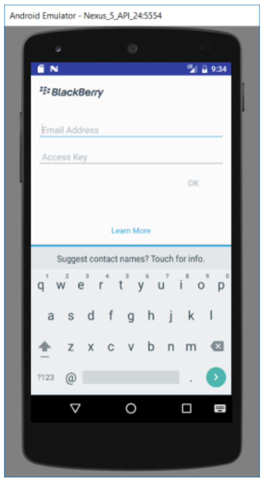

Self-service
My devices
Certificates
Access keys
Welcome to BlackBerry UEM Self-Service
Activate a device
Generate a BlackBerry Dynamics access key

You have no devices activated. Click the + icon to activate a device.

6. Return to BlackBerry Access and enter the access key you just generated.
7. Create a password that will be required to start BlackBerry Access.

Appendix B – Provision an Application in the BlackBerry Developer Lab

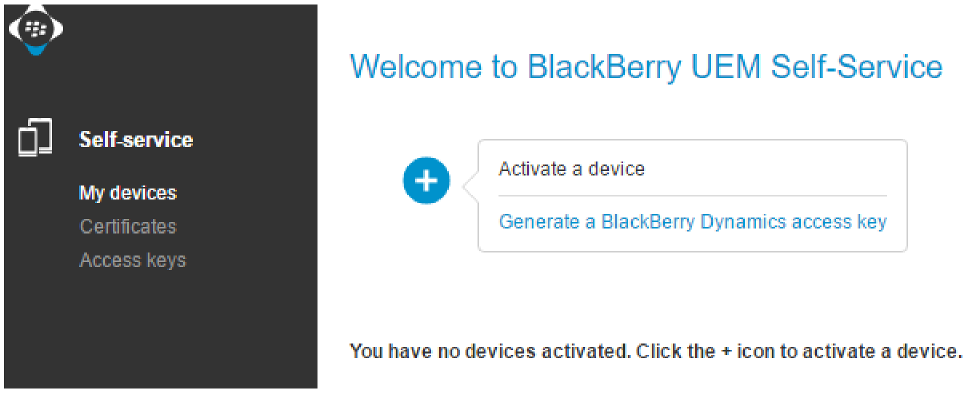
1. In activation screen of your sample application enter an email address of DevUser###@BlackBerryDevSummit.com, replacing ### with the number assigned to you when you arrived at the summit. The activation screen will appear as shown in the screenshot below.


Android Emulator - Nexus_5_API_24:5554
9:34
BlackBerry
Email Address
Access Key
OK
Learn More
Suggest contact names? Touch for info.

2. Navigate to the BlackBerry UEM Self Service Portal to generate an access key: https://uem.blackberrydevsummit.com:10443/index.jsp

3. Log in using the following credentials:
   - Username: DevUser###
     - Replace ### with the number assigned to you when you arrived at the summit.
   - Password: DevUser!2018
   - Domain: blackberrydev

4. Once logged in, click on the + button and choose “Generate a BlackBerry Dynamics access key”.



Welcome to BlackBerry UEM Self-Service

You have no devices activated. Click the + icon to activate a device.

5. Return to your sample application and enter the access key you just generated. The email address to use to activate is DevUser###@BlackBerryDevSummit.com. Replace ### with the number you were assigned.

6. Press the OK button.

7. The activation processing carousel will scroll through its stages.

8. Create a password that will be required to start the application.

The application user interface will then be displayed. Note that activation isn't necessary when you reinstall the application as an upgrade, without uninstalling.