



**Tutorial: Android Object API Application
Development**

SAP Mobile Platform 2.3 SP02

DOCUMENT ID: DC01939-01-0232-01

LAST REVISED: May 2013

Copyright © 2013 by Sybase, Inc. All rights reserved.

This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.

Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.

Sybase trademarks can be viewed at the Sybase trademarks page at <http://www.sybase.com/detail?id=1011207>. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world.

Java and all Java-based marks are trademarks or registered trademarks of Oracle and/or its affiliates in the U.S. and other countries.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.

All other company and product names mentioned may be trademarks of the respective companies with which they are associated.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

SAP Mobile Platform Tutorials	1
Getting Started with SAP Mobile Platform (On-Premise)	
.....	3
Installing SAP Mobile Platform	3
Starting SAP Mobile Platform Services	3
Starting SAP Mobile WorkSpace	4
Connecting to SAP Control Center	4
Learning SAP Mobile WorkSpace Basics	5
Developing an Android Application	9
Installing the Android SDK	9
Installing ADT in SAP Mobile WorkSpace Eclipse	
Environment	10
Generating Java Object API Code	11
Creating the Android Project	14
Adding Compiler and Library Resources	19
Copying SAP Mobile Platform Files to Sample	
Project	20
Configuring Android Application Properties	20
Creating the User Interface	23
Creating a Launch Configuration for the Project	24
Testing the Device Application on the Android	
Emulator	26
Learn More About SAP Mobile Platform	29
Index	31

Contents

SAP Mobile Platform Tutorials

The SAP® tutorials demonstrate how to develop, deploy, and test mobile business objects, device applications, online mobile applications (native OData and REST services based), and Hybrid App packages. You can also use the tutorials to demonstrate system functionality and train users.

- Learn mobile business object (MBO) basics, and use this tutorial as a foundation for the Object API application development tutorials:

- *Tutorial: Mobile Business Object Development*

Note: For all Object API tutorials, if you opt to use the Mobile Business Object example project instead of performing the Mobile Business Object Tutorial, you must deploy the mobile application project to SAP Mobile Server as a prerequisite.

- Create native Object API mobile device applications:
 - *Tutorial: Android Object API Application Development*
 - *Tutorial: BlackBerry Object API Application Development*
 - *Tutorial: iOS Object API Application Development*
 - *Tutorial: Windows Object API Application Development*
 - *Tutorial: Windows Mobile Object API Application Development*
- Create a mobile business object, then develop a hybrid app package that uses it:
 - *Tutorial: Hybrid App Package Development*
- Create an OData mobile application with REST Services
 - *Tutorial: Android OData Application Development with REST Services*
 - *Tutorial: iOS OData Application Development with REST Services*

Getting Started with SAP Mobile Platform (On-Premise)

Install and learn about SAP Mobile Platform and its associated components.

Complete the following tasks for all tutorials, but you need to perform them only once.

Installing SAP Mobile Platform

Install SAP Mobile SDK and SAP Mobile Platform Runtime.

Before starting this tutorial, install all the requisite SAP Mobile Platform components. See the SAP Mobile Platform documentation at <http://sybooks.sybase.com/sybooks/sybooks.xhtml?id=1289&c=firsttab&a=0&p=categories>.

- *Release Bulletin*
 - *Installation Guide for SAP Mobile SDK*
 - *Installation Guide for Runtime*
1. Install these SAP Mobile Platform Runtime components:
 - Data Tier (included with single-server installation)
 - SAP Mobile Server
 2. Install SAP Mobile SDK, which includes:
 - Development support for native Object API and OData SDK applications, as well as HTML5/JS Hybrid Apps.
 - SAP Mobile WorkSpace, the Eclipse-based development environment for MBOs and Hybrid Apps.

Starting SAP Mobile Platform Services

Start SAP Mobile Server, SAP Control Center, the sample database, the cache database (CDB), and other essential services.

The way in which you start SAP Mobile Platform Services depends on the options you selected during installation. You may need to manually start SAP Mobile Platform Services. Select **Start > (All) Programs > SAP > Mobile Platform > Start SAP Mobile Platform Services**.

The following services will be started:

- SAP Control Center <Version>
- SAP Mobile Platform Cache DB

Getting Started with SAP Mobile Platform (On-Premise)

- SAP Mobile Platform SampleDB
- SAP Mobile Server

SAP Mobile Platform Services enable you to access the SAP Mobile Platform runtime components and resources.

Note: The SAP Mobile Platform installer creates the Windows service (SAP Mobile Platform Sample DB) that runs the `sampledb` server only when you install SAP Mobile Server with a Personal or Enterprise Development license. If you installed SAP Mobile Server with an Enterprise Server (production) license, you must create this service using the `sampledb.bat` command line utility. See *Create or Remove the Windows Service for `sampledb` Server (`sampledb`) Utility* in *System Administration* for more information about using this command line utility.

Starting SAP Mobile WorkSpace

Start the development environment, where you can create mobile business objects (MBOs), create connection profiles and manage SAP Mobile Server connections, develop Hybrid Apps, and generate Object API code.

Select **Start > (All) Programs > SAP > Mobile Platform > Mobile WorkSpace 2.3**.

The SAP Mobile WorkSpace opens in the Mobile Development perspective. The Welcome page displays links to the product and information.

Next

To read more about SAP Mobile WorkSpace concepts and tasks, select **Help > Help Contents**.

Connecting to SAP Control Center

Open SAP Control Center to manage SAP Mobile Server and its components.

From SAP Control Center, you can:

- View servers and their status
- Start and stop a server
- View server logs
- Deploy a mobile application package
- Register application connections
- Set role mappings
- Assign/Unassign a hybrid application to a device

For information on configuring, managing, and monitoring SAP Mobile Server, click **Help > Help Contents**.

1. Select **Start > (All) Programs > SAP > SAP Control Center**.

Note: If SAP Control Center does not launch, make sure that the SAP Control Center service is started in the Windows Services dialog.

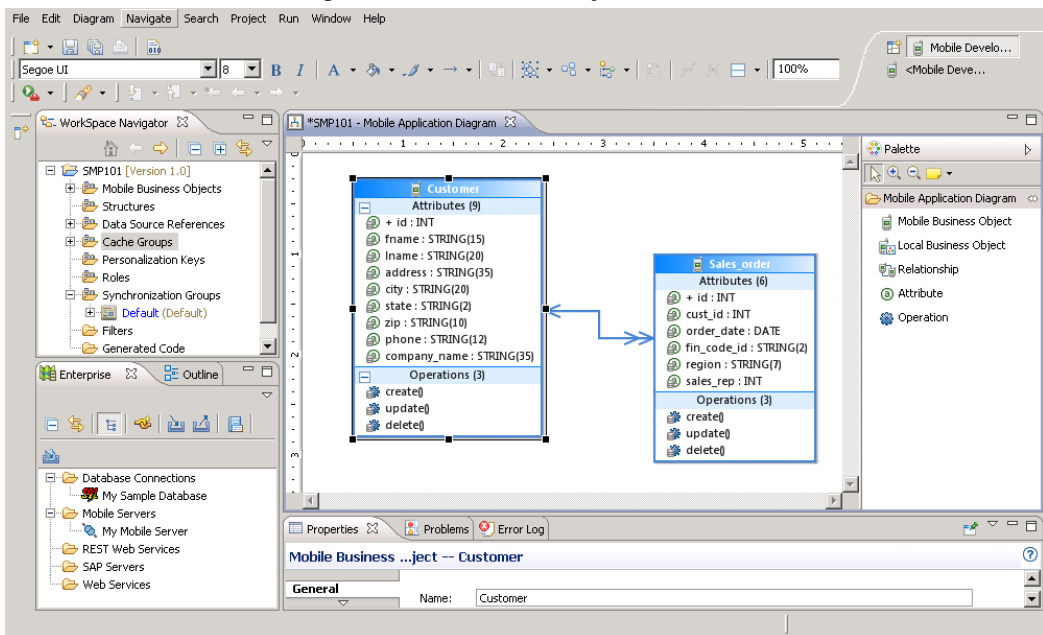
2. Log in by entering the credentials set during installation.

SAP Control Center gives you access to the SAP Mobile Platform administration features that you are authorized to use.

Learning SAP Mobile WorkSpace Basics

SAP Mobile WorkSpace features are well integrated in the Eclipse IDE. If you are unfamiliar with Eclipse, you can quickly learn the basic layout of SAP Mobile WorkSpace and the location of online help.

- To access the online help, select **Help > Help Contents**. Some documents are for SAP Mobile WorkSpace, while others are for the Eclipse development environment.
- The Welcome page provides links to useful information to get you started.
 - To close the Welcome page, click **X** in the upper right corner of the page.
 - Reopen the Welcome page by selecting **Help > Welcome**.
 - To learn about tasks you must perform, select the **Development Process** icon.
- In SAP Mobile WorkSpace, look at the area (window or view) that you will use to access, create, define, and update mobile business objects (MBOs).



Window	Description
<p>WorkSpace Navigator view</p>	<p>Use this view to create Mobile Application projects, and review and modify MBO-related properties.</p> <p>This view displays mobile application project folders, each of which contains all project-related resources in subfolders, including MBOs, datasource references to which the MBOs are bound, personalization keys, and so on.</p>
<p>Enterprise Explorer view</p>	<p>A view that provides functionality to connect to various enterprise information systems (EIS), such as database servers, SAP® back ends, and SAP Mobile Server.</p>
<p>Mobile Application Diagram</p>	<p>The Mobile Application Diagram is a graphical editor where you create and define mobile business objects.</p> <p>Use the Mobile Application Diagram to create MBOs (including attributes and operations), then define relationships with other MBOs. You can:</p> <ul style="list-style-type: none"> • Create MBOs in the Mobile Application Diagram using Palette icons and menu selections – either bind or defer binding to a datasource, when creating an MBO. For example, you may want to model your MBOs before creating the datasources to which they bind. This MBO development method is sometimes referred to as the top-down approach. • Drag and drop items from Enterprise Explorer to the Mobile Application Diagram to create the MBO – quickly creates the operations and attributes automatically based on the datasource artifact being dropped on the Mobile Application Diagram. <p>Each new mobile application project generates an associated mobile application diagram.</p>

Window	Description
Palette	The Palette is accessed from the Mobile Application Diagram and provides controls, such as the ability to create MBOs, add attributes and operations, and define relationships, by dragging and dropping the corresponding icon onto the Mobile Application Diagram or existing MBO.
Properties view	Select an object in the Mobile Application Diagram to display and edit its properties in the Properties view. While you cannot create an MBO from the Properties view, most development and configuration is performed here.
Outline view	Displays an outline of the active file and lists structural elements. The contents are editor-specific.
Problems view	Displays validation errors or warnings that you may encounter in addition to errors in the Diagram editor and Properties view. Follow warning and error messages to adjust MBO properties and configurations to avoid problems, and use as a valuable source for collecting troubleshooting information when reporting issues to Customer Service and Support.
Error Log view	Displays error log information. This is a valuable source for collecting troubleshooting information.

Developing an Android Application

Generate code for the Android platform, develop an Android device application using that code and sample files, and test the application's functionality on an emulator.

Prerequisites

Note: This tutorial was created using SAP Mobile Platform 2.3, Android SDK r21.0.1, ADT Plugin for Eclipse 21.0.0, and run on an Android 4.1.2 - API Level 16 target emulator. If you use a different version, some steps may vary.

1. Complete the tasks in *Getting Started with Mobile Platform*.
2. Either:
 - create the MBO project by completing *Tutorial: Mobile Business Object Development*, or
 - download and deploy the MBO SMP101 example project (complete project files) from the SAP® Community Network: <http://scn.sap.com/docs/DOC-8803>.

Note: If you upgrade SAP Mobile SDK after completing the tutorial, you can convert the project to the current SDK by importing the earlier project into the SAP Mobile Workspace and then accepting the confirmation prompt.

3. (Optional) To use as a reference and copy source code when completing this tutorial, download the Android SMP 101 example project (source code only) from the SAP® Community Network: <http://scn.sap.com/docs/DOC-8803>.
4. Download the supported versions of the Android SDK and Android Development Tools (ADT).
See the *Supported Hardware and Software* guide at <http://sybooks.sybase.com/sybooks/sybooks.xhtml>. Select the appropriate version of the SAP Mobile Platform document set.

Task

Create an Android native application that communicates with the mobile business objects that are deployed to SAP Mobile Server.

Installing the Android SDK

Install the Android SDK.

1. Confirm that your system meets the requirements at <http://developer.android.com/sdk/requirements.html>.
2. Download and install the supported version of the Android SDK starter package.

See the *Google Android Versions* topic for your platform in *Supported Hardware and Software* at <http://sybooks.sybase.com/sybooks/sybooks.xhtml>?

id=1289&c=firsttab&a=0&p=categories. Select the appropriate version of the SAP Mobile Platform document set.

3. Launch the Android SDK Manager and install the Android tools (SDK Tools and SDK Platform-tools) and the Android API.
4. Launch the **Android Virtual Device Manager**, and create an Android virtual device to use as your emulator.

Installing ADT in SAP Mobile WorkSpace Eclipse Environment

Install the supported version of Android Development Tools (ADT) in the SAP Mobile WorkSpace Eclipse environment.

See the *Google Android Versions* topic for your platform in *Supported Hardware and Software* at <http://sybooks.sybase.com/sybooks/sybooks.xhtml?id=1289&c=firsttab&a=0&p=categories>.

Select the appropriate version of the SAP Mobile Platform document set.

1. Start SAP Mobile WorkSpace, then select **Help > Install New Software**.
2. In the Available Software window, click **Add**.
3. In the Add Repository window, enter ADT Plugin for the name and <https://dl-ssl.google.com/android/eclipse/> for the location. Click **OK**.
4. In the Available Software window, select **Developer Tools**, then click **Next**.
5. In the Install Details window, you see a list of downloadable tools. Click **Next**.
6. Accept the license agreements, then click **Finish**.

Note: If you see a security warning about the authenticity or validity of the software, click **OK**.

7. When the installation completes, restart SAP Mobile WorkSpace.
8. For first-time installations:
 - a) In Welcome to Android Development, select **Use existing SDKs**, then browse to where the Android SDK is installed, by default, `C:\Program Files\Android\android-sdk`.
 - b) Click **Next**.
9. Click **Finish**.

Generating Java Object API Code

Use the Generate Code wizard to generate object API code for the SMP101 mobile application project. Code generation creates the business logic, attributes, and operations for the mobile business objects (MBOs) in the project.

Prerequisites

- In Enterprise Explorer, you must be connected to both My Sample Database and My Mobile Server. Code generation fails if the server-side (runtime) enterprise information system (EIS) datasources referenced by the MBOs in the project are not running and available to connect to when you generate object API code.
- In WorkSpace Navigator, verify the Java Compiler level is set correctly. Compiler levels must match or there will be errors.
 1. Select **Window > Preferences > Java > Compiler**.
 2. In the Compiler compliance level list, select **1.6** if it does not already appear.
 3. Click **Apply**, then **OK**.

Task

1. In SAP Mobile WorkSpace, open the **SMP101** mobile application project.
In WorkSpace Navigator, right-click the **SMP101** folder and select **Open in Diagram Editor**.
2. In WorkSpace Navigator, expand **SMP101**. Under **Generated Code**, add a folder named **Android**.

The `Generated Code` directory was created during the MBO tutorial.

3. Right-click anywhere in the SMP101 - Mobile Application Diagram and select **Generate Code**.
4. In the Generate Code wizard, click **Next** to continue without a configuration.
5. In the Select Mobile Business Objects window, select the **Customer** MBO, then click **Next**.

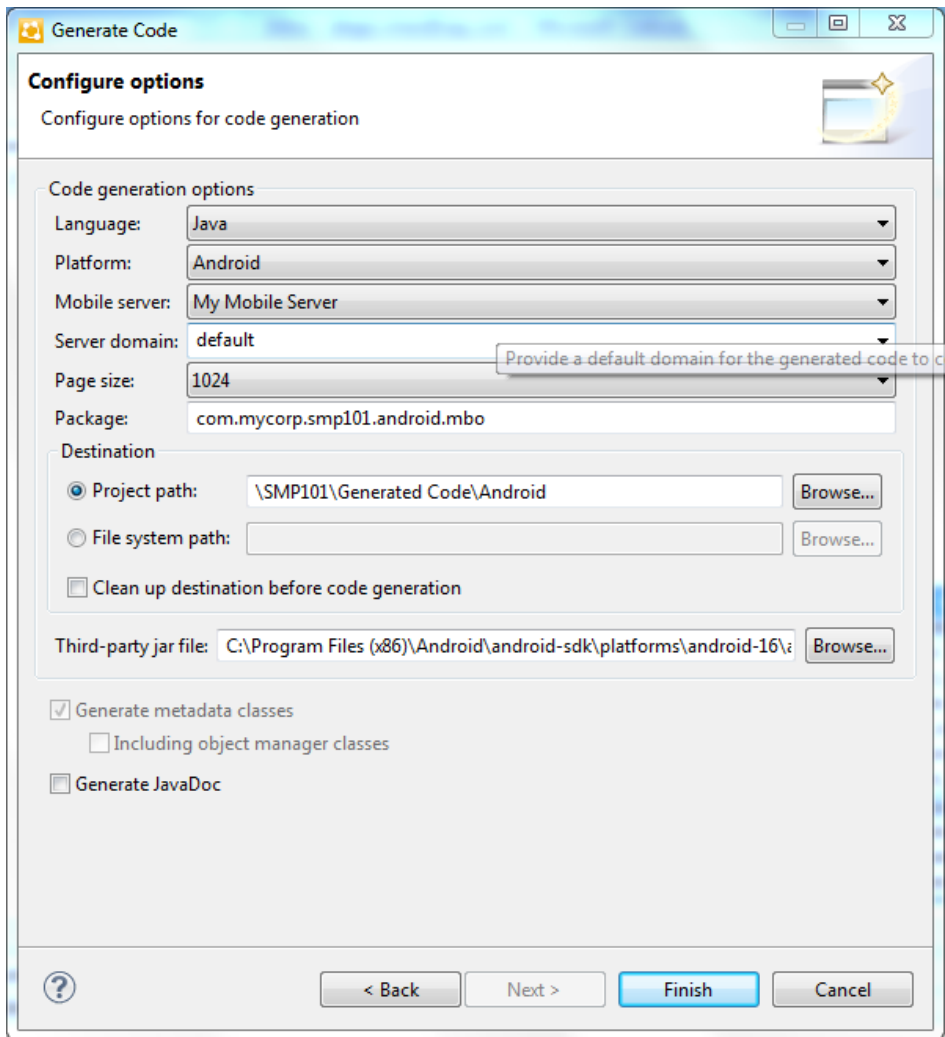
Ignore any warning about unresolved mobile business object dependencies. The warning appears because you selected the customer data only.

6. In the Configure options window, specify these values and click **Finish**.

Option	Description
Language	Select Java .
Platform	Select Android .

Option	Description
Mobile server	Select My Mobile Server (or the server to which you deployed the SMP101 project, if you used another).
Server domain	Select default .
Page size	Select 1024 . The page size should be larger than the sum of all attribute lengths for any MBO that is included with all the MBOs selected, and must be valid for the database. If the page size is not set, the default page size is 4KB at runtime.
Package	Enter <code>com.mycorp.smp101.android.mbo</code> . Bundling groups of related types into packages makes types easier to find and use, avoids naming conflicts, and controls access.
Project path	Enter <code>\SMP101\Generated Code \Android</code> .
Third-party jar file	Click Browse to open an <code>android.jar</code> , by default located in <code>C:\Program Files (x86)\Android\android-sdk\platforms\android-xx</code> . This is the Android system library.
Generate JavaDoc	Unselect for this tutorial. You are not creating automatic documentation from comments in the code.

Note: For more about the configuration options, see *Developer Guide: Android Object API Applications*.



7. In the Success dialog, click **OK**.
In the Generated Code directory, you see an \Android\src\ folder.

Creating the Android Project

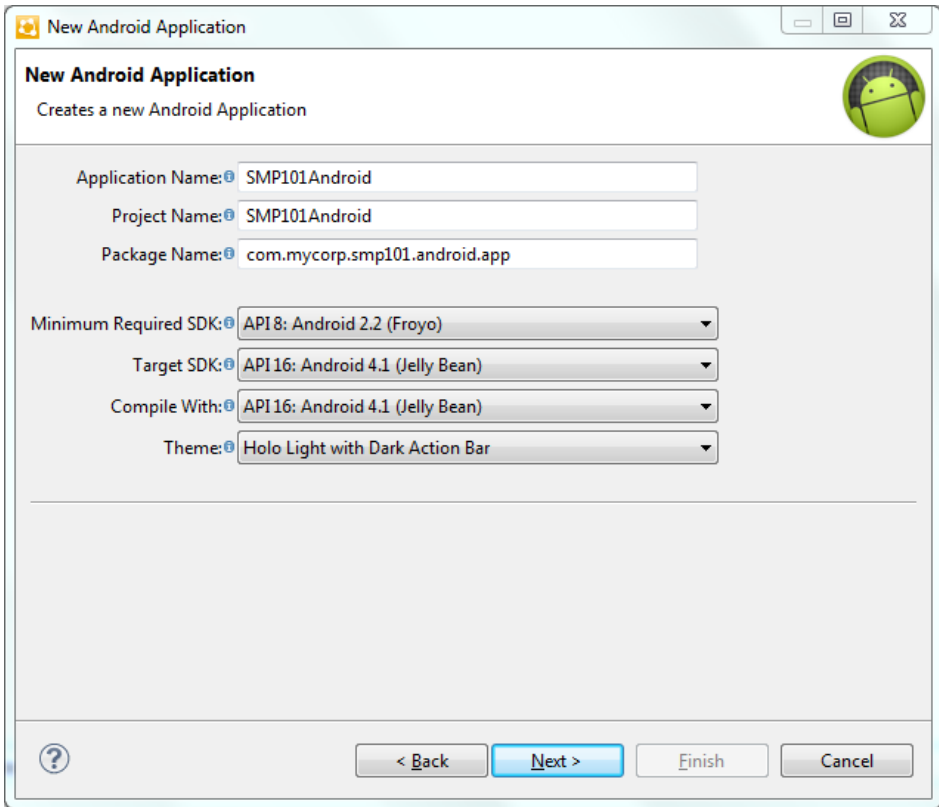
Create a new Android project in SAP Mobile WorkSpace. Add library resources to the project and set other application properties.

Prerequisites

To help create your project—and in a subsequent topic, build the user interface—download the SMP101 Android Object API (2.3) example project from the SAP Community Network (SCN) Web site at <http://scn.sap.com/docs/DOC-8803>.

Task

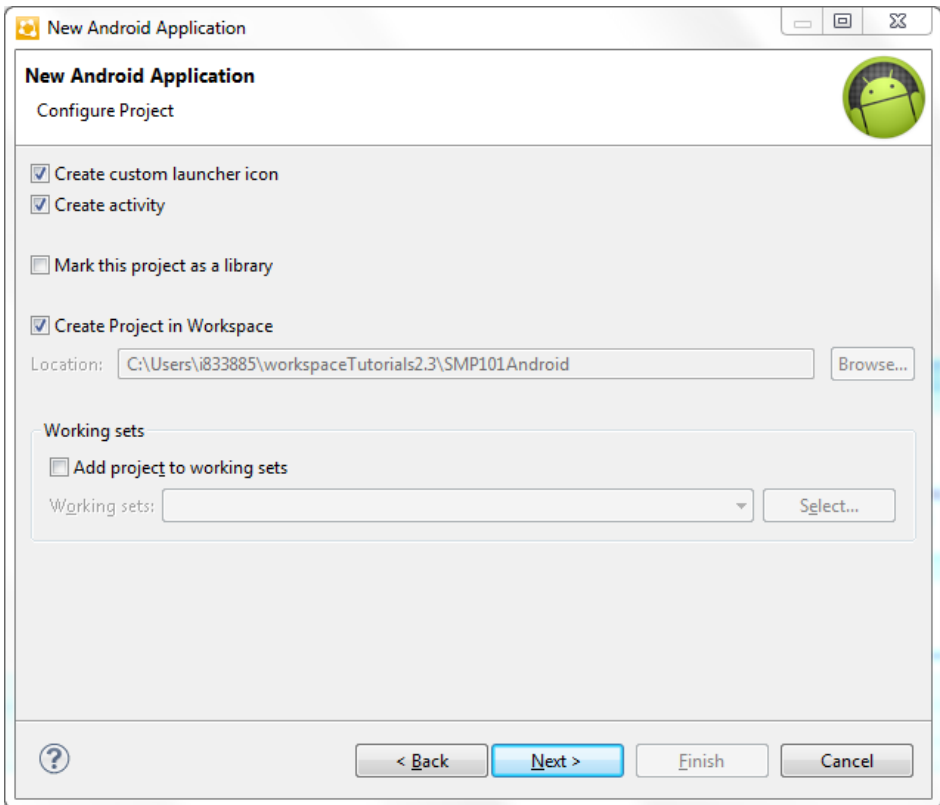
1. Start SAP Mobile WorkSpace.
2. In SAP Mobile WorkSpace Preferences, set the Android SDK location.
3. Select **File > New > Project**.
4. Select **Android > Android Application Project**, then **Next**.
If you have a version of Android other than the one used to design this tutorial, the screens you use to enter the information in the next several steps may be in different screens.
5. In the Creates a new Android Application page of the New Android Application wizard, use these values and then click **Next**.
 - Application Name – enter SMP101Android.
 - Project Name – enter SMP101Android.
 - Package Name – enter `com.mycorp.smp101.android.app`.
 - Minimum Required SDK – accept the default.
 - Target SDK – select the Android SDK used for the tutorial.
 - Compile With – select the Android SDK used for the tutorial.
 - Theme – accept the default.



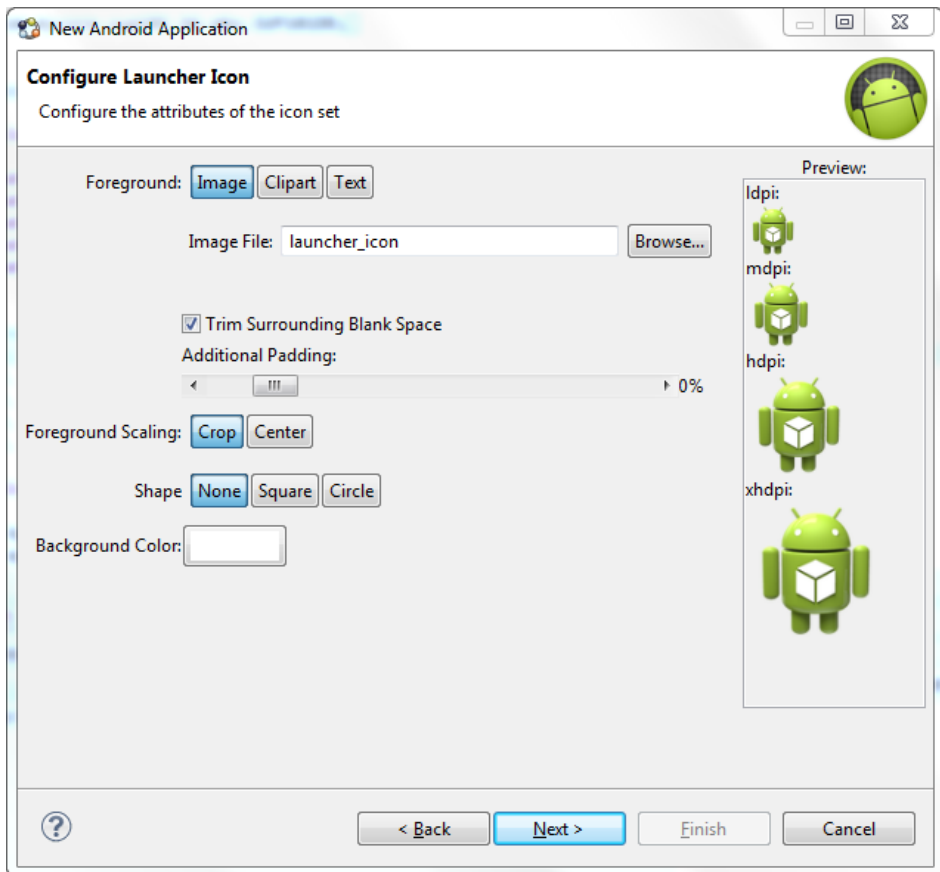
6. In the Configure Project page, use these values and click then **Next**.

Option	Description
Create custom launcher icon	Select this option to specify an icon.
Create activity	Select this option to choose a template to use as the basis for the project.
Create Project in Workspace	Select this option to create the project in the default workspace. You can unselect the option to create the project in another location that you specify.
Mark this project as a library	Unselect this option. You are not creating a library project.

Option	Description
Add project to working sets	Unselect this option. Working sets limit the resources shown in the WorkSpace Navigator and Enterprise Explorer. This project is small and so working sets are not needed. See <i>SAP Mobile WorkSpace: Mobile Business Object Development</i> to learn more about working sets.

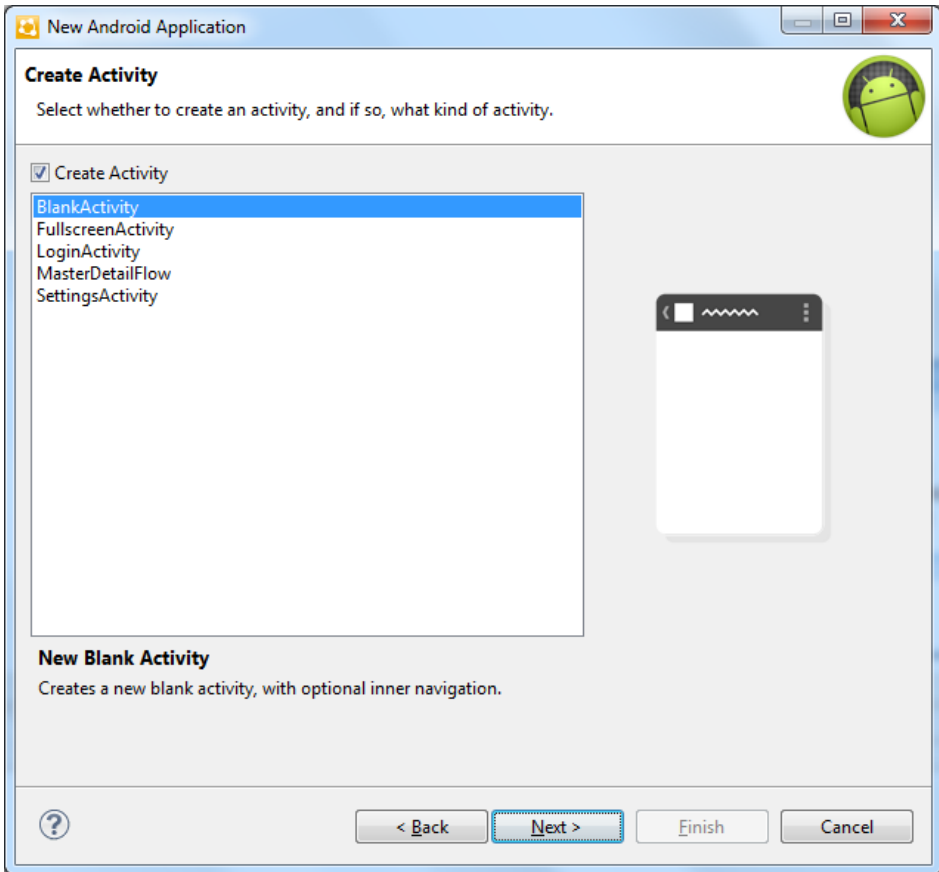


7. In the Configure Launcher Icon page you can define a custom image file to use as the launcher icon for the application. Accept the default settings and click **Next**.



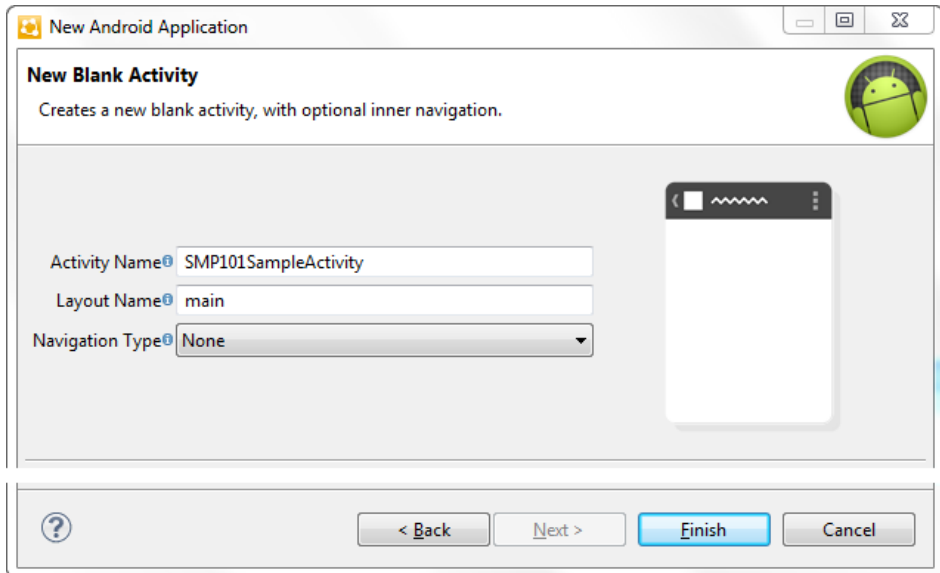
8. In the Create Activity window select **Create Activity**, then select **BlankActivity**, and click **Next**.

Choosing a template creates a simple Android application that you can use for your project.



9. In the New Blank Activity window, use these values and click **Finish**.

Option	Description
Activity Name	Enter SMP101SampleActivity.
Layout Name	Enter main.
Navigation Type	Accept the default of None.



The BlankActivity template with Navigation Type of None creates an application that follows the Android Design guidelines. It includes a title bar, an options menu, and a basic layout.

The left pane of the Workspace Navigator should list the SMP101Android project. In the `src` folder, a default Sample Activity class was automatically generated when you created the project.

Tip: To correct a misspelled package name, right-click the package and select **Refactor** > **Rename** to change the name and update all references.

Adding Compiler and Library Resources

Add compiler and library resources to the Android project.

1. Add a compiler resource to the root directory of the project:
 - a) In Windows Explorer, browse to `SMP_HOME\MobileSDK23\ObjectAPI\Android` (where `SMP_HOME` is the directory where SMP is installed, for example, `C:\SAP\UnwiredPlatform`) and copy the `armeabi` (Ultralite native library) folder and these JAR files: `AfariaSSL.jar` (for Afaria users), `ClientLib.jar` (messaging), `sup-client.jar` (ObjectAPI client), and `UltraLiteJNI12.jar` (database access).
 - b) In Workspace Navigator, expand **SMP101Android**, select the `libs` folder, and paste the `armeabi` folder and JAR files into it.
2. Add library resources to the project:

- a) In Workspace Navigator, right-click the **SMP101Android** project, click **Properties**, and select **Java Build Path**.
- b) Click the **Libraries** tab and select **Add JARs**.
- c) In the JAR Selection window, expand the `SMP101Android\libs` folder and select **AfariaSSL.jar**, **ClientLib.jar**, **sup-client.jar**, and **UltraLiteJNI12.jar**.
- d) Click the **Order and Export** tab and select those four JARs.
- e) Click **OK**.

Copying SAP Mobile Platform Files to Sample Project

Copy the object API code you generated using the Generate Code wizard for Android.

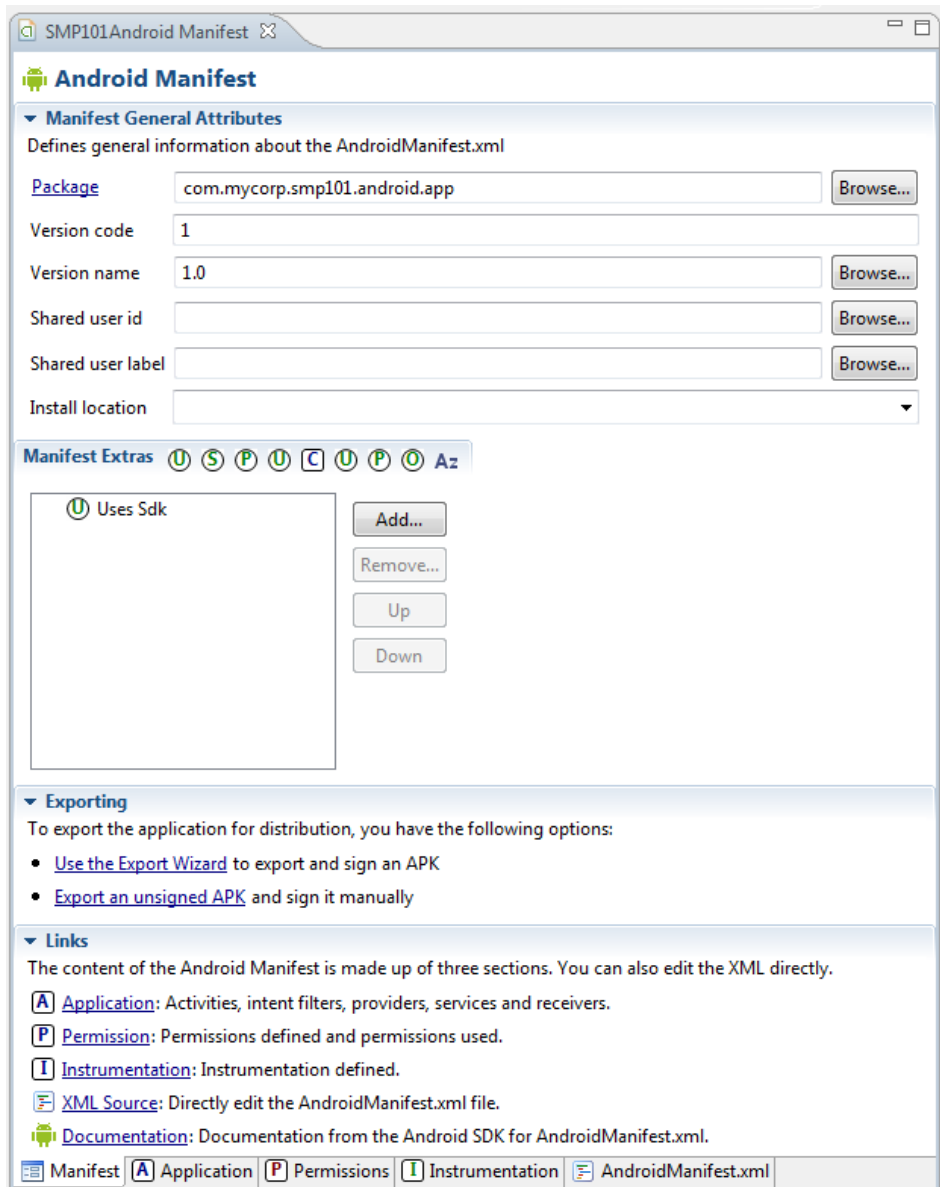
1. In Workspace Navigator, go to the **SMP101** project and copy the `com` folder in `\Generated Code\Android\src\`.
2. Go to the **SMP101Android** project and paste the `com` folder in to the `src` directory. Select **Yes to All** to copy over existing folders.

Configuring Android Application Properties

(Optional) Review the Android Manifest window, where you define the general Android properties used in an application.

1. In Workspace Navigator, expand the **SMP101Android** project.
2. Double-click the `AndroidManifest.xml` file.
3. Select the **Manifest** tab.
4. Review the options in the Android Manifest window, where you can change the general attributes, export options, and content of the `AndroidManifest.xml` file.

Tip: In **Manifest Extras**, you can click **Uses Sdk** to indicate the API level for the minimum Android SDK version on which you want to run the application.



5. Select **File > Save**.

Next

Modify the Android manifest file to add a Detail Activity class.

Adding User Permissions and a Class to the Android Manifest File

Add user permissions to the Android project. Also add a Detail Activity class to the `AndroidManifest.xml` file. This declaration launches a customer detail screen where you can make changes when you test the application.

1. If needed, open the Android manifest file.
2. Select the **AndroidManifest.xml** tab.
3. Replace the code with the source code from the `AndroidManifest.xml` file you downloaded from the SAP Community Network (SCN) Web site, also provided below:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/
android"
    package="com.mycorp.smp101.android.app"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.READ_PHONE_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".SMP101SampleActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".DetailActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

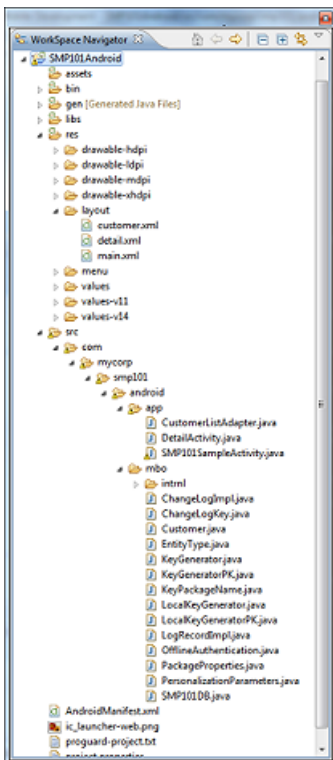
4. Select **File > Save**.

Creating the User Interface

Copy the Java code files, which provide the functionality and layout of the user interface, from the SMP101 Android Project example project archive to the SMP101Android project.

1. In Windows Explorer, browse to the directory where you saved the SMP101 Android Project example project.
2. Copy these Java files: `CustomerListAdapter.java`, `DetailActivity.java`, and `SMP101SampleActivity.java`.
3. In Workspace Navigator, go to **SMP101Android** and expand `\src\com\mycorp\smp101\android\app`, then paste the copied Java files, copying over any existing files.
4. Modify the host IP address in the `SMP101SampleActivity.java` file to point to the SAP Mobile Server.
 - a) In Workspace Navigator, expand the **SMP101Android** project.
 - b) Under the `\src\com\mycorp\smp101\android\app` folder, double-click the `SMP101SampleActivity.java` file.
 - c) Modify the host IP address, and verify the username and password are valid.
5. Browse to the directory where you saved the SMP101 Android Project example project.
6. Copy the sample layout XML files: `customer.xml`, `detail.xml`, and `main.xml`.
7. In the SMP101Android project folder, go to the `res\layout` directory and paste the copied XML files, copying over any existing files.

The SMP101Android project directory should look like this:



Creating a Launch Configuration for the Project

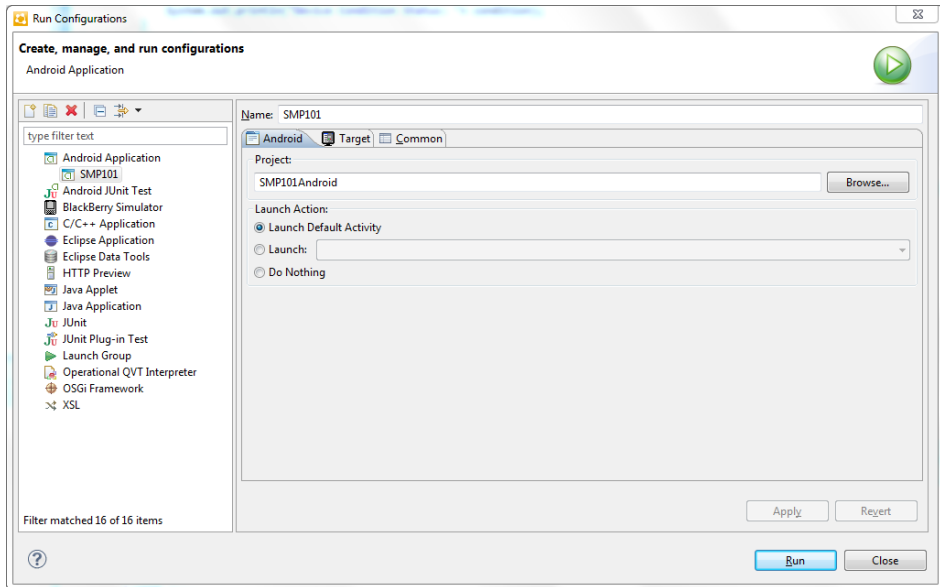
Create a new launch configuration for the SMP101Android project. The configuration specifies how the application launches, and defines the target Android platform.

Prerequisites

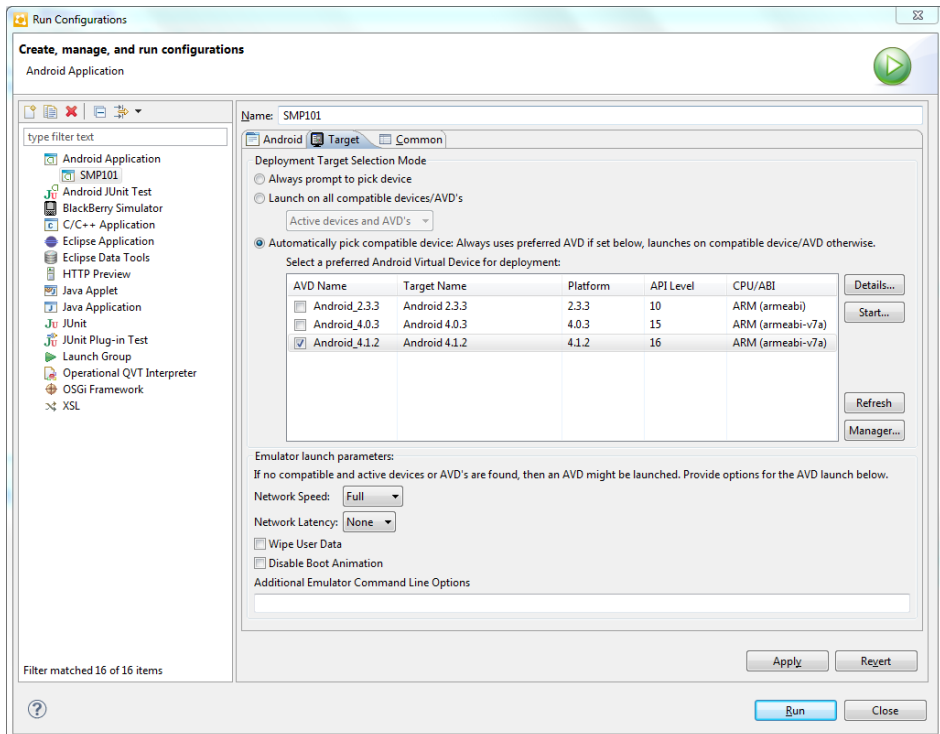
In the SAP Mobile WorkSpace, use the AVD Manager to add a new target Android Virtual Device (AVD).

Task

1. In Workspace Navigator, right-click the **SMP101Android** project, and select **Run As > Run Configurations**.
2. Right-click **Android Application** and select **New**.
3. In the Name field, enter **SMP101**.
4. In the Android tab, click **Browse** and select **SMP101Android**. Click **OK**.
5. In the Launch Action area, select **Launch Default Activity**.



- In the Target tab, select a deployment target. For example, select Automatically pick compatible device, then specify an AVD for deployment. Accept all other default settings.

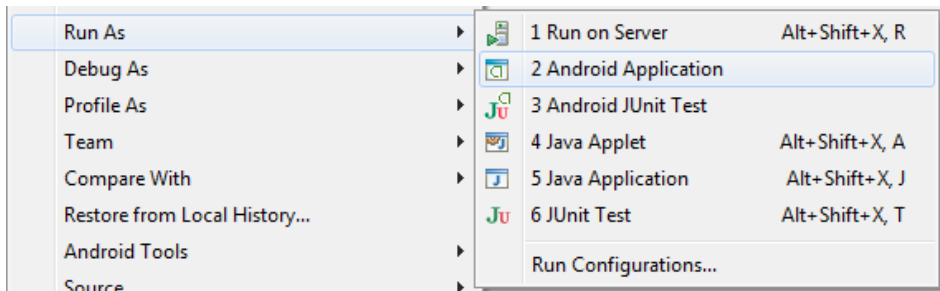


7. Click **Apply**, then **Close**.

Testing the Device Application on the Android Emulator

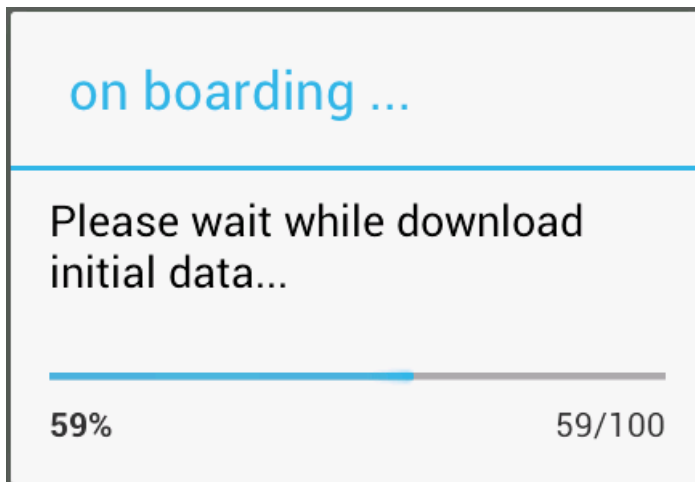
Run the SMP101Android application on the Android emulator, and change customer information to update the interface.

1. In WorkSpace Navigator, right-click **SMP101Android** and select **Run As > Android Application**.



Note: It may take several minutes for the Android emulator's home screen to appear.

The application activation (on boarding) image indicates that the application is registering data from SAP Mobile Server.

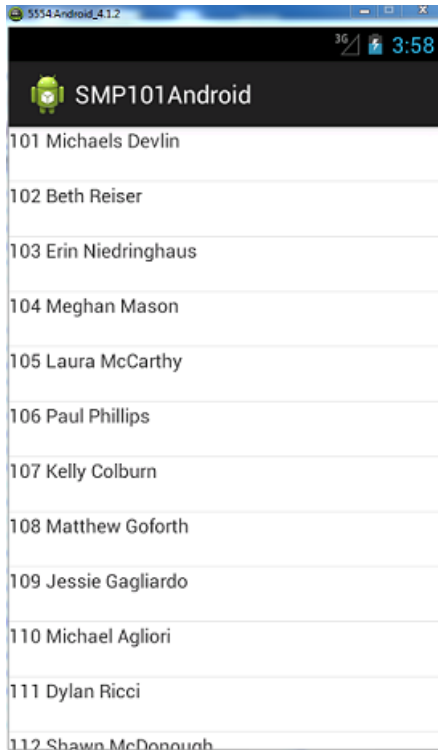


During initialization, the system enables the operation to target change notifications using:

```
SynchronizationGroup  
sg=SMP101DB.getSynchronizationGroup("default");  
sg.setEnableSIS(true);  
sg.save();
```

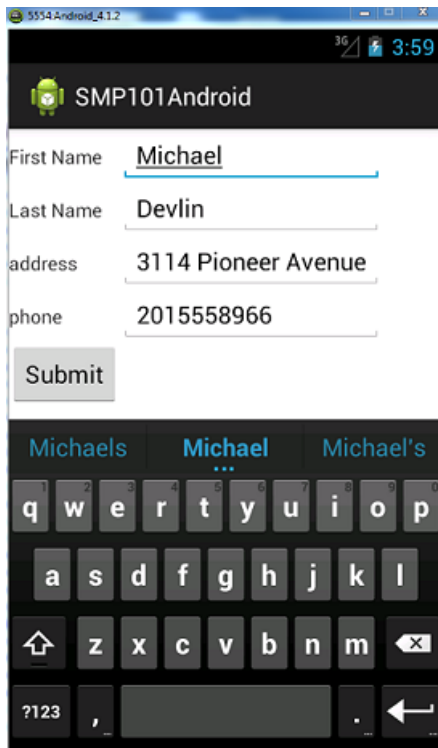
When the data finishes synchronizing, the device application shows the SMP101Android application with a list of customer data in a ListView control. You can scroll through the customer list to see more data and to make changes. The data loads from the database on demand.

When the application queries the customer list, it uses an `SMP101DB.executeQuery()` API to get only columns that are needed, such as (fname, lname...), instead of the entire customer object; this results in better performance.



2. Select the customer to update.
3. In the customer detail screen, change the first name of the customer and click **Submit**.

Developing an Android Application



The Submit button is mapped to the synchronize operation using `SMP101DB.beginSynchronize`. The synchronization occurs in the background, so the user interface is unaffected.

Any back-end changes initiate notifications from the server. The device application uses a ChangeLog API to query those managed items and update the user interface if needed.

```
GenericList<ChangeLog> changeLogs=SMP101DB.getChangeLogs(query);
```

4. Close the emulator to stop the SMP101Android application.

Learn More About SAP Mobile Platform

Once you have finished, try some of the other samples or tutorials, or refer to other development documents in the SAP Mobile Platform documentation set.

Check the Product Documentation Web site regularly for updates: <http://sybooks.sybase.com/sybooks/sybooks.xhtml?id=1289&c=firsttab&a=0&p=categories>, then navigate to the most current version.

Tutorials

Try out some of the other getting started tutorials available on the Product Documentation Web site to get a broad view of the development tools available to you.

Example Projects

An example project contains source code for its associated tutorial. It does not contain the completed tutorial project. Download example projects from the SAP® Community Network (SCN) at <http://scn.sap.com/docs/DOC-8803>.

Samples

Sample applications are fully developed, working applications that demonstrate the features and capabilities of SAP Mobile Platform.

Check the SAP® Development Network (SDN) Web site regularly for new and updated samples: <https://cw.sdn.sap.com/cw/groups/sup-apps>.

Online Help

See the online help that is installed with the product, or available from the Product Documentation Web site.

Developer Guides

Learn best practices for architecting and building device applications:

- *Mobile Data Models: Using Data Orchestration Engine* – provides information about using SAP Mobile Platform features to create DOE-based applications.
- *Mobile Data Models: Using Mobile Business Objects* – provides information about developing mobile business objects (MBOs) to fully maximize their potential.
- *SAP Mobile WorkSpace: Mobile Business Object Development* – provides information about using SAP Mobile Platform to develop MBOs and generate Object API code that can be used to create native device applications and Hybrid Apps.

Use the appropriate API to create device applications:

- *Developer Guide: Android Object API Applications*
- *Developer Guide: BlackBerry Object API Applications*

Learn More About SAP Mobile Platform

- *Developer Guide: iOS Object API Applications*
- *Developer Guide: Windows and Windows Mobile Object API Applications*
- *Developer Guide: Hybrid Apps*
- *Developer Guide: OData SDK*
- *Developer Guide: REST API Applications*

Customize and automate:

- *Developer Guide: SAP Mobile Server Runtime > Management API* – customize and automate system administration features.

Javadoc and HeaderDoc are also available in the installation directory.

Index

A

- ADT Plugin for Eclipse, installing 10
- Android application
 - attributes 20
- Android project 14
 - manifest file 22
 - src folder 20
- Android SDK 9
- AndroidManifest.xml 14, 20
 - Detail Activity 22
- application properties 20

B

- build path 14

C

- ClientLib.jar 14
- compiler resource 19
- customer.xml 23
- CustomerListAdapter.jar 23

D

- default_package.jar 11
- deployment target, launch 24
- deployment_unit.xml 11
- Detail Activity 22
- detail.xml 23
- DetailActivity.jar 23

E

- example projects 1

G

- Generate Code wizard 11
- generated object API code 11
 - using 20
- generating code 11

H

- Hybrid App package tutorial 1

J

- JAR files
 - ClientLib.jar 14
 - sup-client.jar 14
 - UltraLiteJNI2.jar 14
- Java class, creating 23
- Java files
 - CustomerListAdapter.jar 23
 - DetailActivity.jar 23
 - SMP101SampleActivity.jar 23
- Java object API code, generating 11
- Java perspective 23
- JDK 9

L

- launch configuration 24
- layout files
 - customer.xml 23
 - detail.xml 23
 - main.xml 23
- library resource 19

M

- main.xml 23
- manifest file 14, 22
- mobile business object tutorial 1

O

- Object API tutorials 1

P

- project build path 14
- properties, application 20

R

- resources
 - compiler 19

Index

library 19
run configurations 24

S

samples
 downloading 29
SAP Control Center
 connecting to 4
SAP Mobile Platform
 documentation resources 29
 getting started 3
 installing 3
SAP Mobile Platform Runtime
 installing 3
SAP Mobile Platform services 3
SAP Mobile SDK
 installing 3
SAP Mobile WorkSpace
 basics 5
 how to access online help 5
 starting 4
SAP Mobile WorkSpace basics 5
SAP Mobile WorkSpace, src folder 20
simulator 9
SMP101Android application, testing 26
SMP101SampleActivity.jar 23
SMP101SampleProject 14

src folder
 Android project 20
 SAP Mobile WorkSpace 20
sup-client.jar 14

T

troubleshooting information 5
tutorials 1
 downloading 29

U

UltraLiteJNI12.jar 14
user interface, creating 23

V

virtual devices 9

X

XML files
 customer.xml 23
 detail.xml 23
 main.xml 23