ATID Co.,Ltd

RFID Programming Guide

Android Developer Guide

SW Team

2021-07-14

Revision History

Ver.	Date	Reason[1]	Description[2]	Writer
V1.0	2021-07-14	Draft	Initial draft	SW Team

[1] Revision : Define the contents are addition/modification/deletion

[2] Description: Describe revised page number and contents

Contents

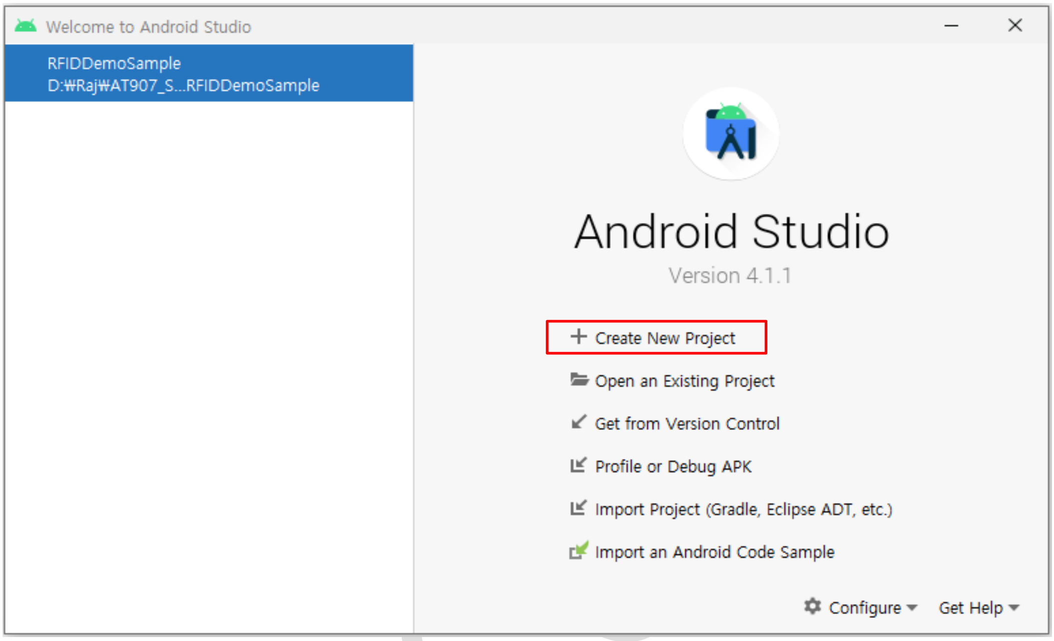
1. Intro

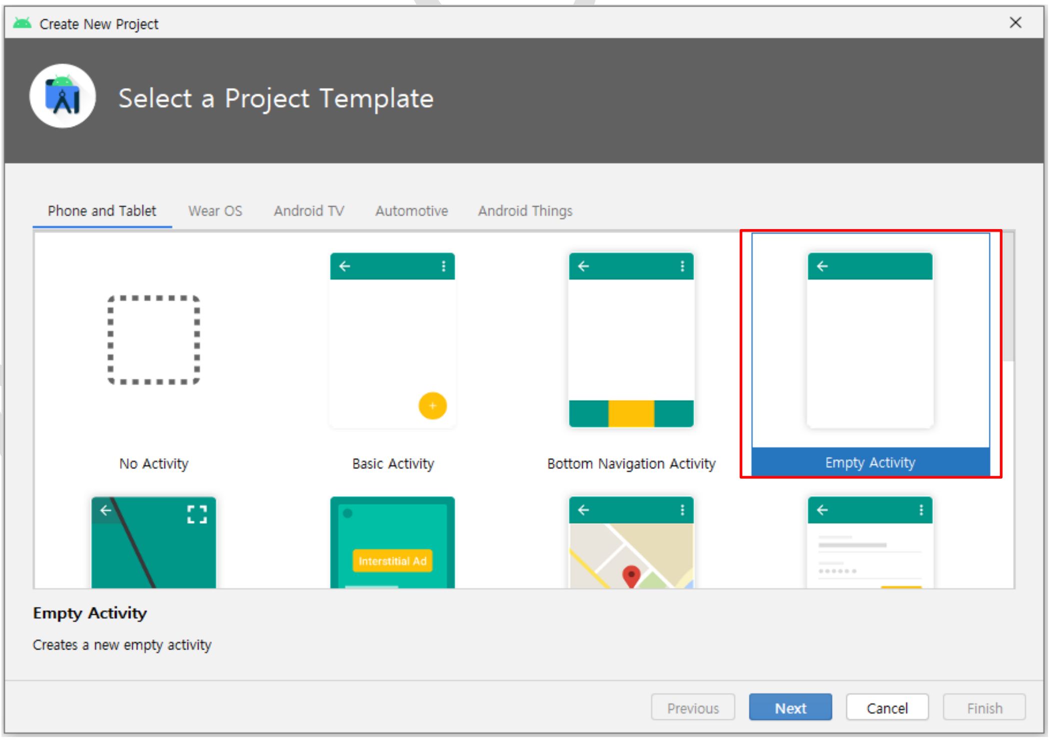
This Programming Guide explains how to use AT907 RFID SDK Library to develop Android application program. The development tool used here is Android Studio and the target platform supports Android 10.

2. Development environment setup

2.1. Create a project.

Create a new project in Android studio.

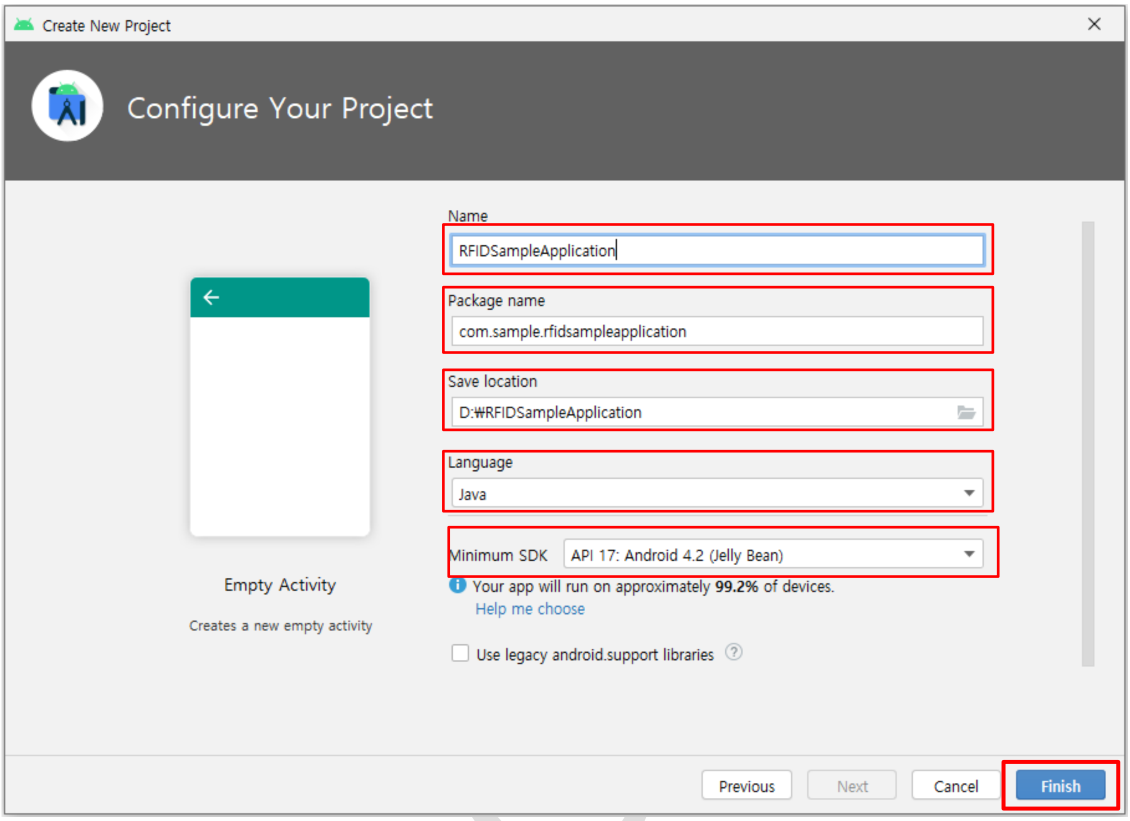




Empty Activity

Creates a new empty activity

Enter the application name, package name, save location, Language, and minimum SDK details.



Create New Project

Configure Your Project

Empty Activity

Creates a new empty activity

Name: RFIDSampleApplication

Package name: com.sample.rfidsampleapplication

Save location: D:₩RFIDSampleApplication

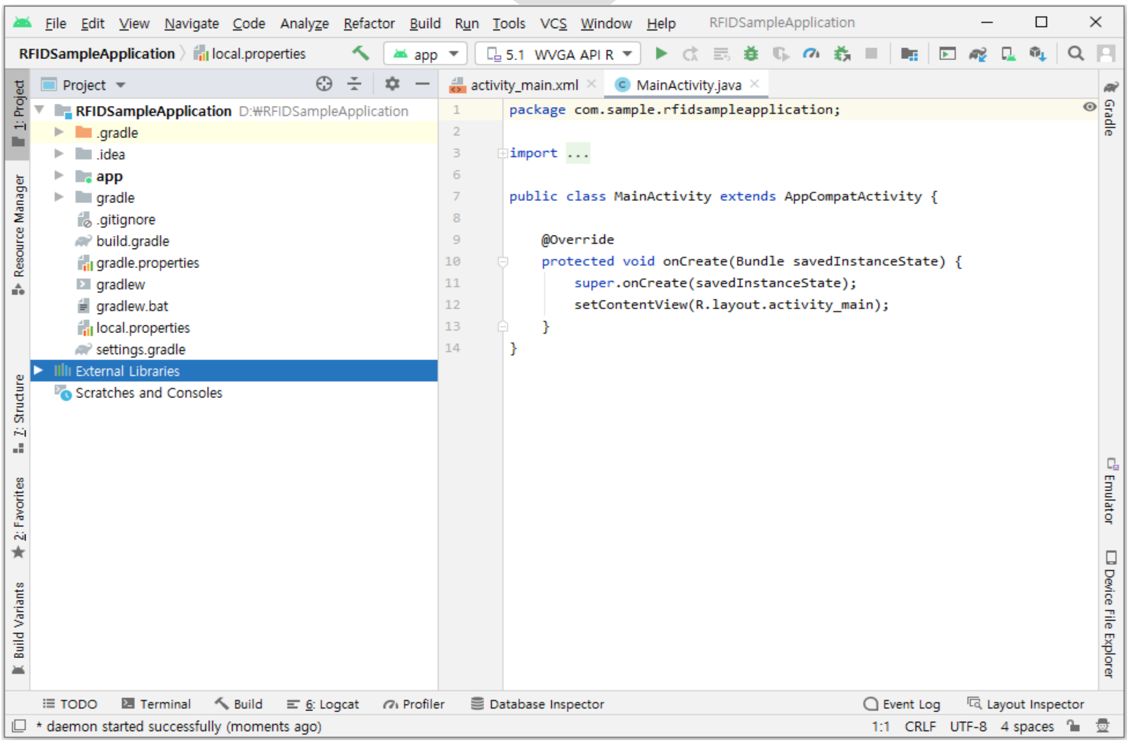
Language: Java

Minimum SDK: API 17: Android 4.2 (Jelly Bean)

Your app will run on approximately **99.2%** of devices.

Help me choose

Use legacy android.support libraries



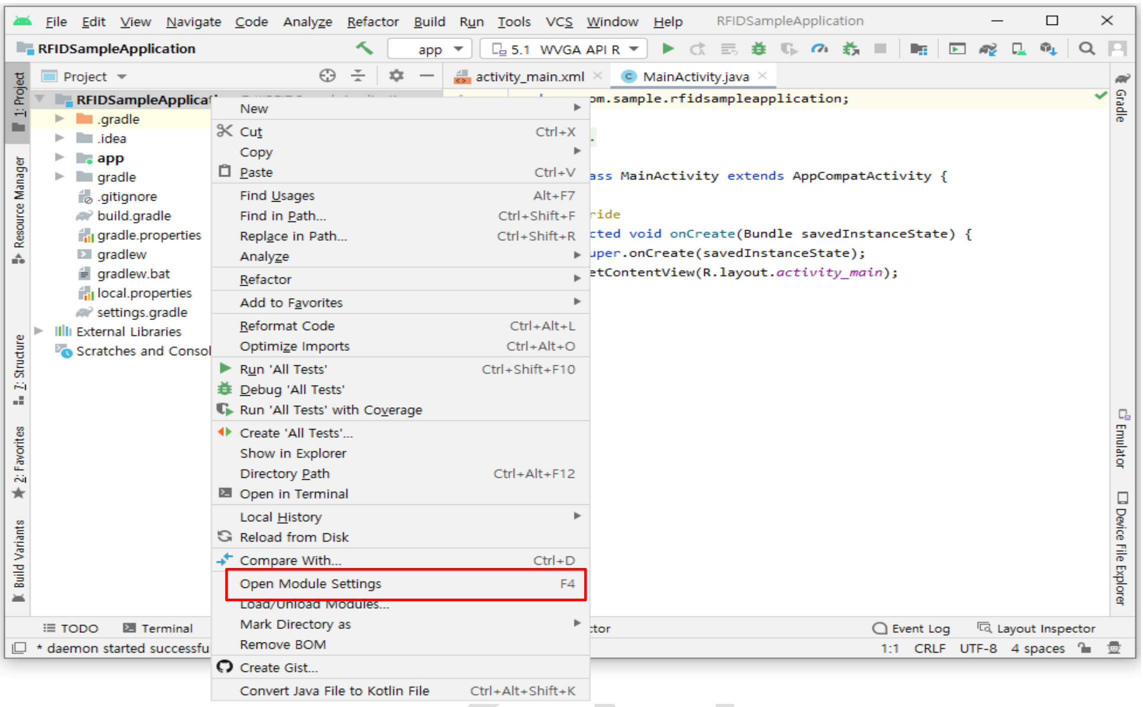
```
RFIDSampleApplication  D:₩RFIDSampleApplication
  .gradle
  .idea
  app
  gradle
  .gitignore
  build.gradle
  gradle.properties
  gradlew
  gradlew.bat
  local.properties
  settings.gradle
External Libraries
Scratches and Consoles
```

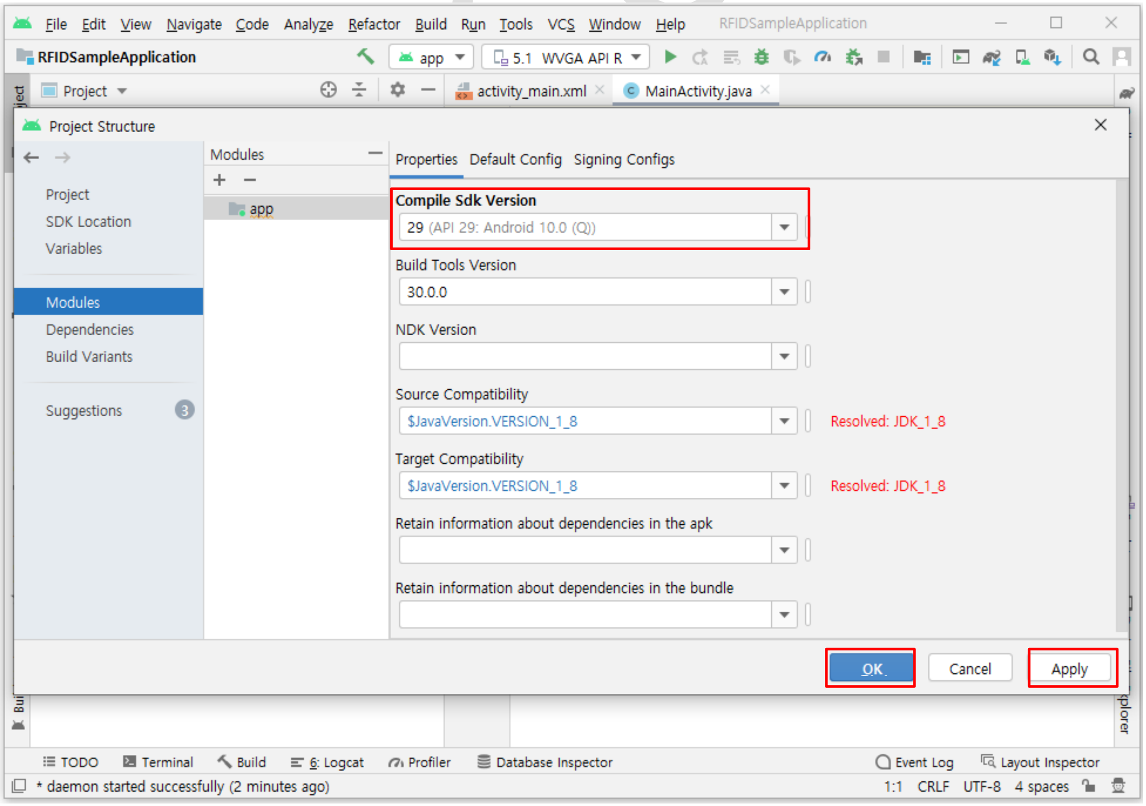
```java
package com.sample.rfidsampleapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

* daemon started successfully (moments ago)

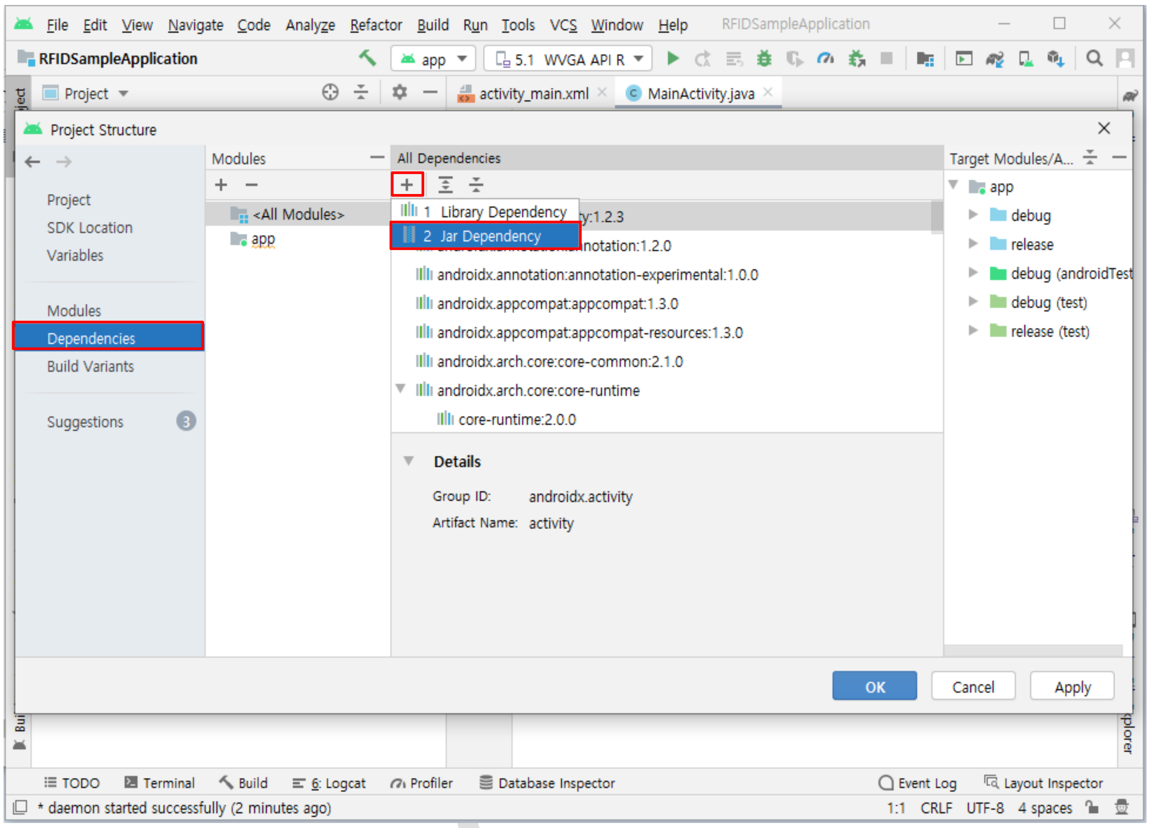
1:1 CRLF UTF-8 4 spaces

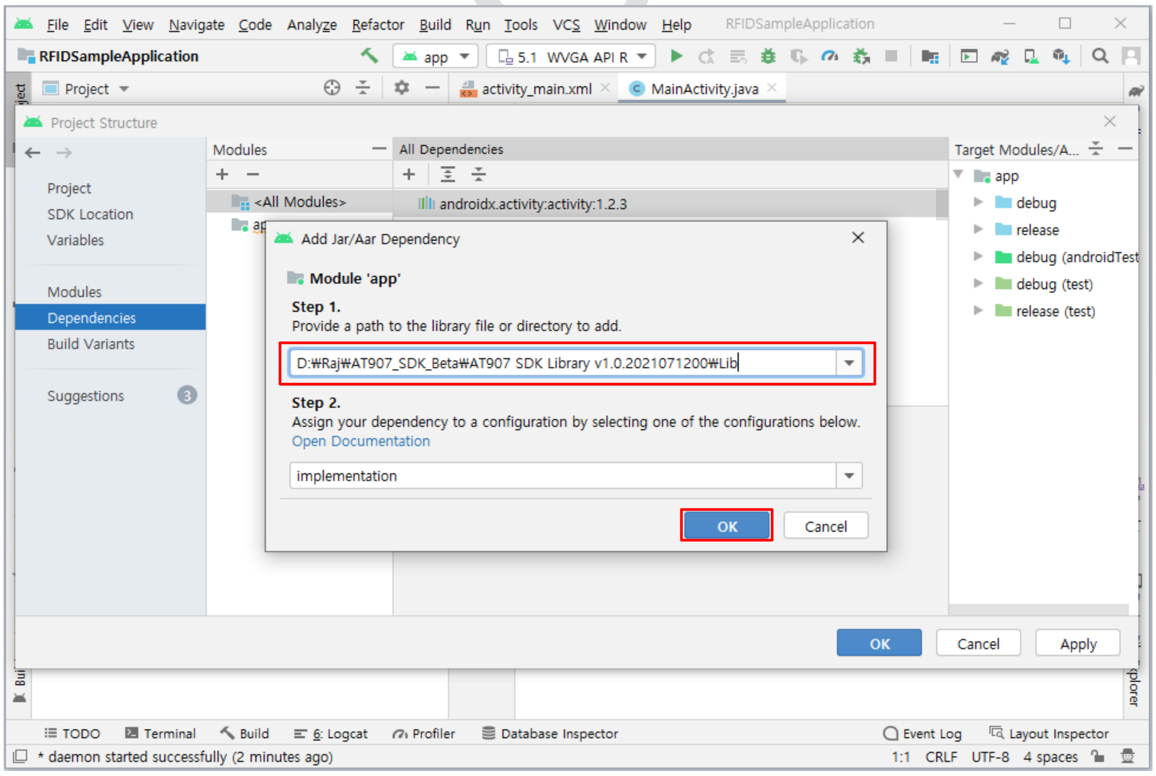
2.1.1 SDK setup and add RFID dependency libraries (ex: jar/aar files)





Compile Sdk Version: 29 (API 29: Android 10.0 (Q))
Build Tools Version: 30.0.0
NDK Version:
Source Compatibility: $JavaVersion.VERSION_1_8 Resolved: JDK_1_8
Target Compatibility: $JavaVersion.VERSION_1_8 Resolved: JDK_1_8
Retain information about dependencies in the apk:
Retain information about dependencies in the bundle:



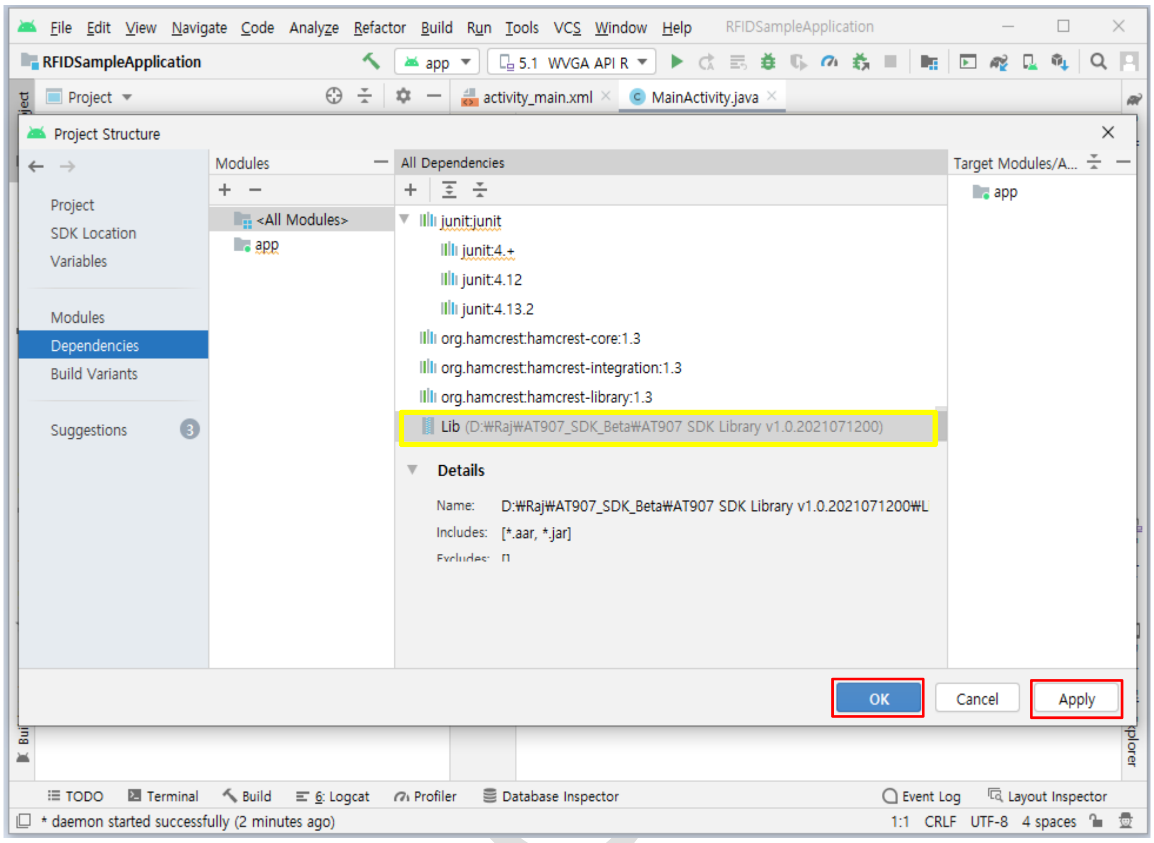


Step 1.
Provide a path to the library file or directory to add.

D:₩Raj₩AT907_SDK_Beta₩AT907 SDK Library v1.0.2021071200₩Lib

Step 2.
Assign your dependency to a configuration by selecting one of the configurations below.
Open Documentation

implementation



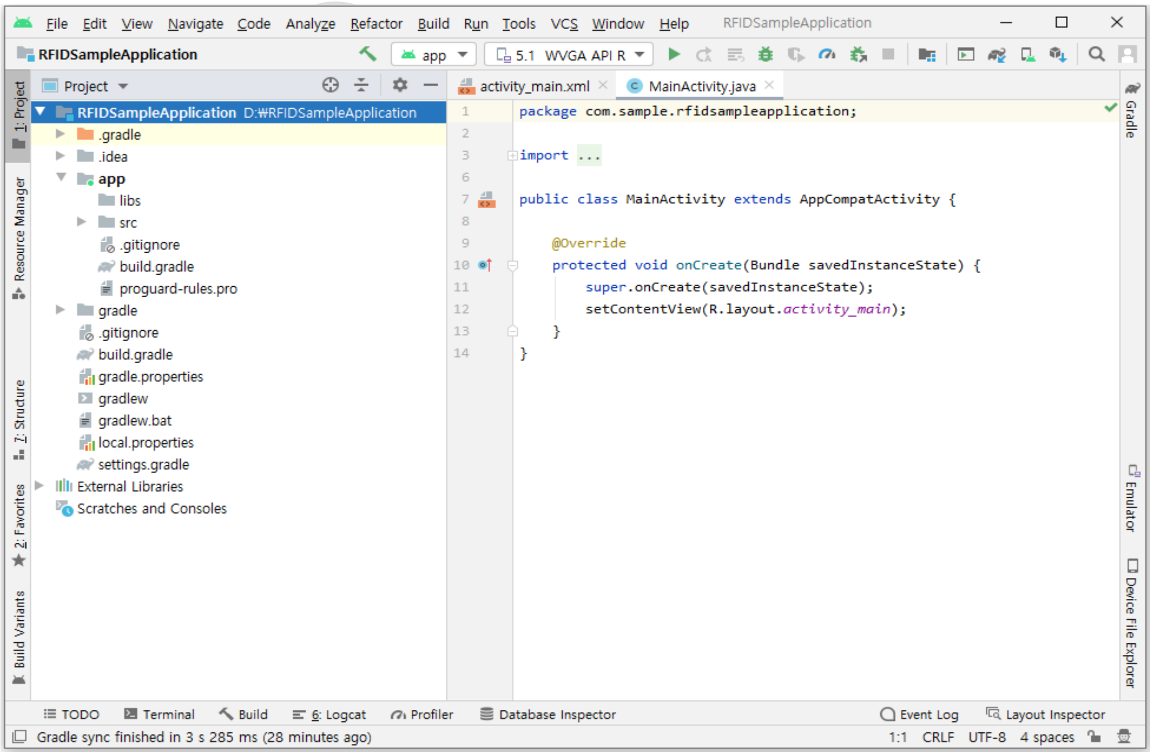
All Dependencies:

- junit:junit
  - junit:4.+
  - junit:4.12
  - junit:4.13.2
- org.hamcrest:hamcrest-core:1.3
- org.hamcrest:hamcrest-integration:1.3
- org.hamcrest:hamcrest-library:1.3
- Lib (D:₩Raj₩AT907_SDK_Beta₩AT907 SDK Library v1.0.2021071200)

Details

Name:	D:₩Raj₩AT907_SDK_Beta₩AT907 SDK Library v1.0.2021071200₩L
Includes:	[*.aar, *.jar]
Excludes:	[]

```
* daemon started successfully (2 minutes ago)        1:1  CRLF  UTF-8  4 spaces
```



File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help RFIDSampleApplication
RFIDSampleApplication
app
5.1 WVGA API R
Project
activity_main.xml
MainActivity.java
1: Project
Resource Manager
7: Structure
2: Favorites
Build Variants
Gradle
Emulator
Device File Explorer
TODO Terminal Build 6: Logcat Profiler Database Inspector Event Log Layout Inspector

```
RFIDSampleApplication D:₩RFIDSampleApplication
  .gradle
  .idea
  app
    libs
    src
    .gitignore
    build.gradle
    proguard-rules.pro
  gradle
  .gitignore
  build.gradle
  gradle.properties
  gradlew
  gradlew.bat
  local.properties
  settings.gradle
External Libraries
Scratches and Consoles
```

```java
package com.sample.rfidsampleapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

```
Gradle sync finished in 3 s 285 ms (28 minutes ago)        1:1  CRLF  UTF-8  4 spaces
```

3. Programing Guide

3.1. Initialization

3.1.1 Create Reader Object

Initially create the instance of ATRfidReader. For clear undersatdning refer MainActivity.java in the provided sample source.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

        // TODO

    if ((mReader = ATRfidManager.getInstance()) == null) {
    // ERROR
    }
    // TODO
}
```

3.1.2 Register / Unregister Event Listener.

To receive a response from an ATRfidReader instance, the RfidReaderEventListerner interface must be executed in the user activity. Register the event listener using SetEventListener() method and un-register/cancel it by using removeEventlistener(). If need to perform read operations in the multiple activities follow the similar approach.

For clear understanding please refer to the below sample code,

```
@Override
protected void onResume() {
        super.onResume();

        if (mReader != null)
                mReader.setEventListener(this);

        ATLog.d(TAG, "INFO onResume()");
}
@Override
protected void onPause() {
        if (mReader != null)
                mReader.removeEventListener(this);

        ATLog.i(TAG, "INFO. onPause()");
        super.onPause();
}
```

3.2. Module power management.

To reduce the power consumption, it is recommended to use the ATRfidManager.wakeUp() in the onStart of activity and ATRfidManager.sleep() in the onStop.

Make sure that for every wakeup call there must be a sleep call. (***note:*** wrong handling of sleep/wakeup calls could lead to module abnormal behavior).

```java
@Override
protected void onStart() {
       super.onStart();

       if (mReader != null) {
              ATRfidManager.wakeUp();
       }

       ATLog.i(TAG, "INFO. onStart()");
}
@Override
protected void onStop() {

       ATRfidManager.sleep();

       ATLog.i(TAG, "INFO. onStop()");

       super.onStop();
}
```

3.3. Event Handler

Once the EventListerner is registered, the interface function implemented by the user is executed whenever event occurs,

List of events,

Module Operation (`onReaderStateChanged`),

Inventory Operation (`onReaderReadTag`) and Access Operation (`onReaderResult`)

onReaderStateChange method is to receiving module's connection state events.

Refer to code below,

```
@Override
public void onReaderStateChanged(ATRfidReader reader, ConnectionState
state) {

       switch (state) {
       case Connected:
              // to do something
              break;
       case Disconnected:
              // to do something
              break;
       case Connecting:
              // to do something
              break;
       default:
              break;
       }

       ATLog.i(TAG, "EVENT. onReaderStateChanged(%s)", state);
}
```

The onReaderReadTag method is executed when the inventory starts and returns the read tag information such as tag data, RSSI and phase information.

```
 @Override
public void onReaderReadTag(ATRfidReader reader, String tag, float rssi,
float phase) {

   ATLog.i(TAG, "EVENT. onReaderReadTag([%s], %.2f, %.2f)", tag, rssi,
phase);
}
```

onReaderResult method is to check Read/Write/Lock/Kill and other access operation results.

```
@Override
public void onReaderResult(ATRfidReader reader, ResultCode code,
ActionState action, String epc, String data, float rssi, float phase) {
       ATLog.i(TAG, "EVENT. onReaderResult(%s, %s, [%s],
[%s], %.2f, %.2f", code, action, epc, data, rssi, phase);
}
```

3.4. Start and Stop Inventory

3.4.1 Start Inventory

To start inventory of tag, inventory6cTag, inventory6bTag, readEpc6cTag, readEpc6bTag methods are used.

Refer to the InventoryActivity.java for the sample implementation.

```
// Start Action
protected void startAction() {
        // 생략
        if(mReader.getModuleType() == RfidModuleType.I900MA) {
                mMAReader = (ATRfid900MAReader)mReader;
                if (chkContinuousMode.isChecked()) {

                        // Multiple Reading
                        if(tagType == TagType.Tag6B) {
                                mMAReader.inventory6bTag()
                        } else if(tagType == TagType.Tag6C) {
                                mMAReader.inventory6cTag()
                        } else if(tagType == TagType.TagRail) {
                                mMAReader.inventoryRailTag()
                        } else if(tagType == TagType.TagAny) {
                                mMAReader.inventoryAnyTag()
                        }
                } else {
                        // Single Reading
                        if(tagType == TagType.Tag6B) {
                                mMAReader.readEpc6bTag()
                        } else if(tagType == TagType.Tag6C) {
                                mMAReader.readEpc6cTag()
                        } else if(tagType == TagType.TagRail) {
                                mMAReader.readEpcRailTag()
                        } else if(tagType == TagType.TagAny) {
                                mMAReader.readEpcAnyTag()
                        }
                }
        } else {
                        // Multiple Reading
                if (chkContinuousMode.isChecked()) {
                        mReader.inventory6cTag()
                } else {
                        // Single Reading
                        mReader.readEpc6cTag()
                }
        }

        ATLog.i(TAG, "INFO. startAction()");
}
```

3.4.2 Stop Inventory

To stop inventory or access commands in-progress, call mReader.stop method.

Refer to the ActionActivity.java file.

```
protected void stopAction() {

        if(mReader.getAction() == ActionState.Stop) {
                ATLog.e(TAG, "ActionState is not busy.");
                return;
        }
        ResultCode res;
        enableWidgets(false);

        if ((res = mReader.stop()) != ResultCode.NoError) {
                ATLog.e(TAG, "ERROR. stopAction() - Failed to stop
operation [%s]", res);
                enableWidgets(true);
                return;
        }

        ATLog.i(TAG, "INFO. stopAction()");
}
```

3.5. Closing an application.

Finally, do not forget to clean-up/destroy the Reader Objects using ATRfidManager.onDestroy() method.

Refer to the MainActivity.Java's onDestroy () in the sample code.

```
@Override
protected void onDestroy() {

        // Deinitalize RFID reader Instance
        ATRfidManager.onDestroy();

        // Wake Unlock
        SysUtil.wakeUnlock();

        saveConfig();

        ATLog.d(TAG, "INFO. onDestroy");
        ATLog.shutdown();

        super.onDestroy();

}
```