



ATID Co.,Ltd

RFID Programming Guide

Android Developer Guide

SDK Team

2017-02-22

		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

Revision History

Ver.	Date	Reason ¹	Description ²	Writer
V1.0	2015-07-07	Draft	Draft Document	Y.H Park
V2.0	2017-02-22	Modify	Apply update information	Y.J Cho

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

² Description: Describe revised page number and contents

		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

Contents

Contents	3
1. Intro	4
2. Getting Started	5
2.1. Create Project for Android.....	5
2.2. Development Environment	11
3. Programing Guide.....	15
3.1. Create and Synchronization	15
3.1.1. Create Reader.....	15
3.1.2. EventListener register and release.....	15
3.1.3. To manager Sleep/Wakeup.....	16
3.2. Event Handler.....	17
3.3. Action Operation.....	18
3.3.1. Start Operation.....	18
3.3.2. Stop Operation	19
3.4. End.....	20

		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

1. Intro

This Programming Guide explains how to use AT911N RFID SDK Library to develop Android application program. The development tool used Eclipse for Java Developers and the target platform supports Android 4.2.2

		RFID Programming Guide					
All That Identification		Android Developer Guide				Company	ATID Co.,Ltd
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

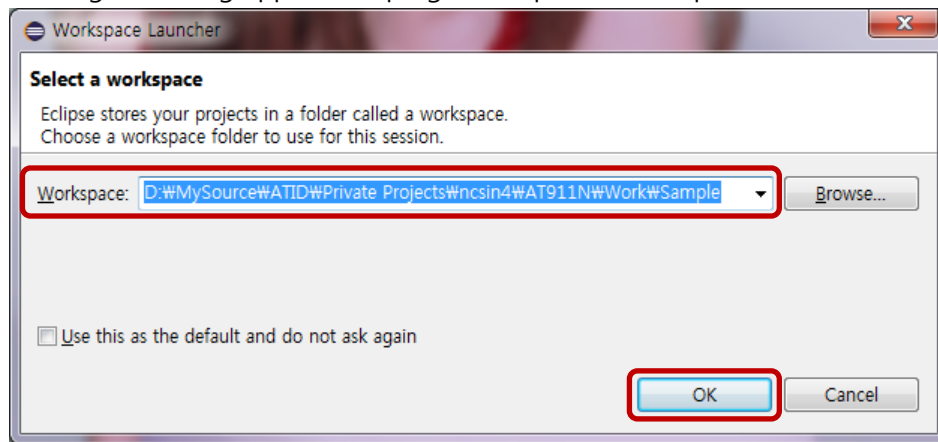
2. Getting Started

In Getting started, the steps in building development environment for using SDK Library, and Building application program are explained.

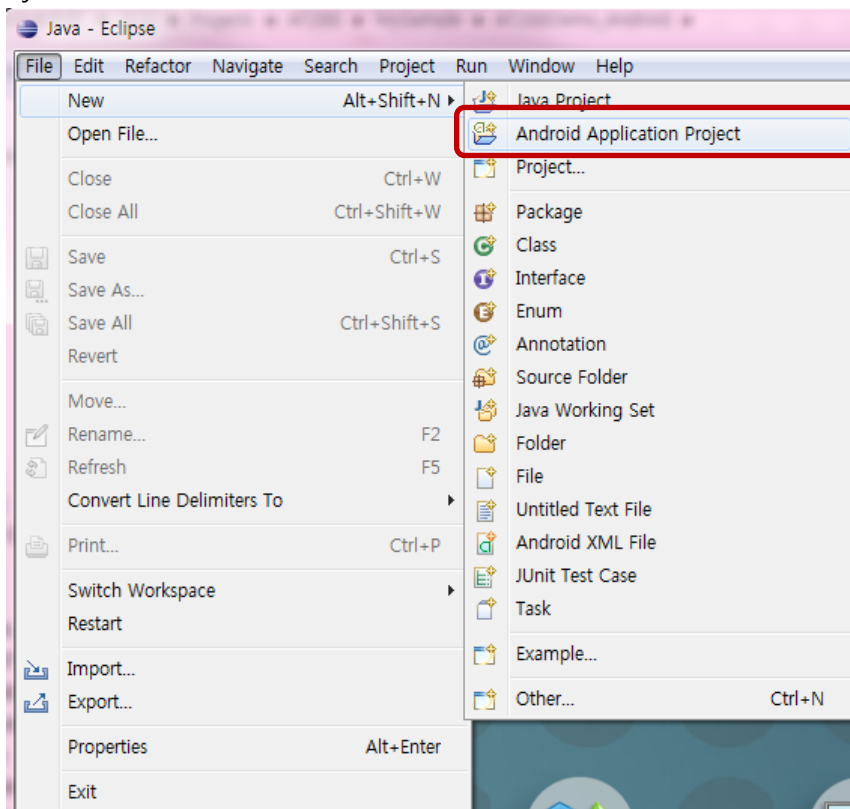
Eclipse Juno is used for the demonstration.

2.1. Create Project for Android

Run Eclipse to begin building application program steps for development.

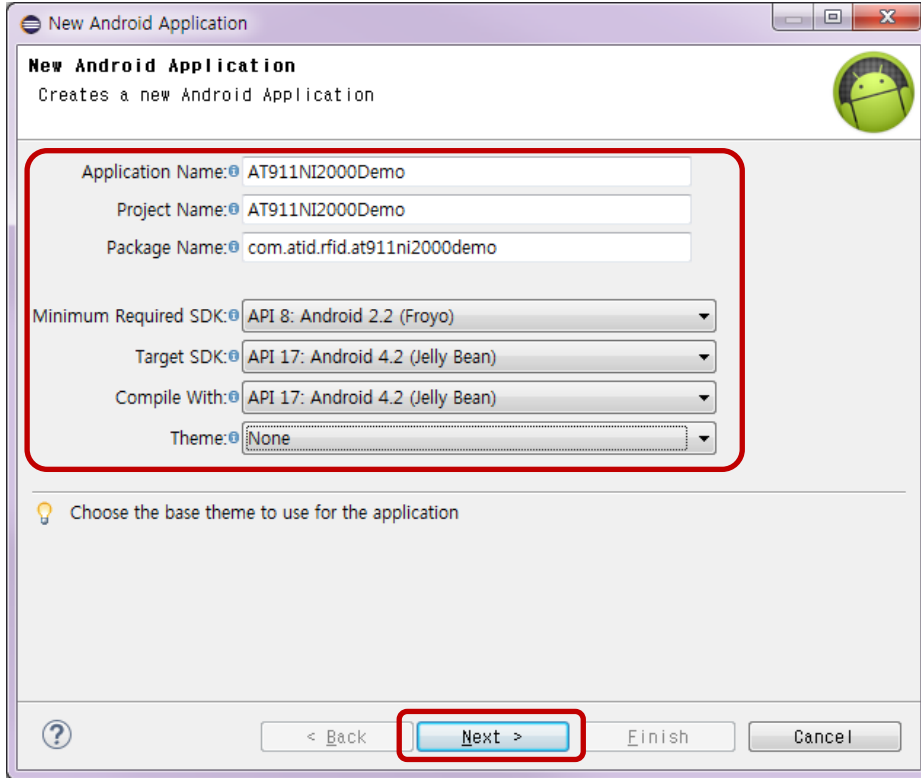


When developer is running Eclipse, dialogue box is shown,
Located the project folder and Click "OK"

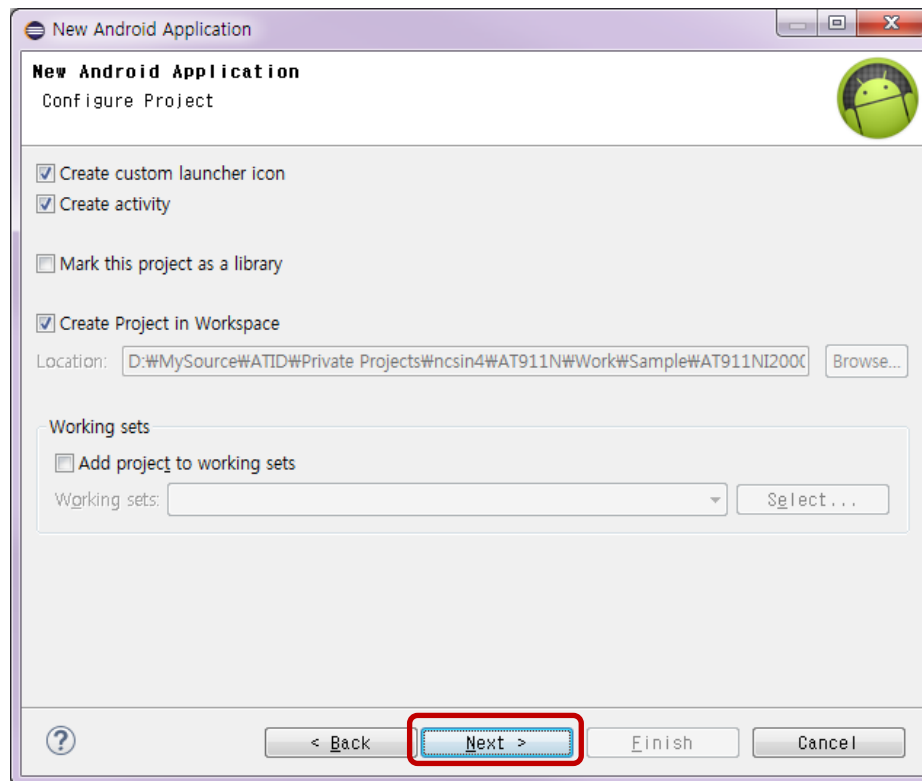


 All That Identification		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

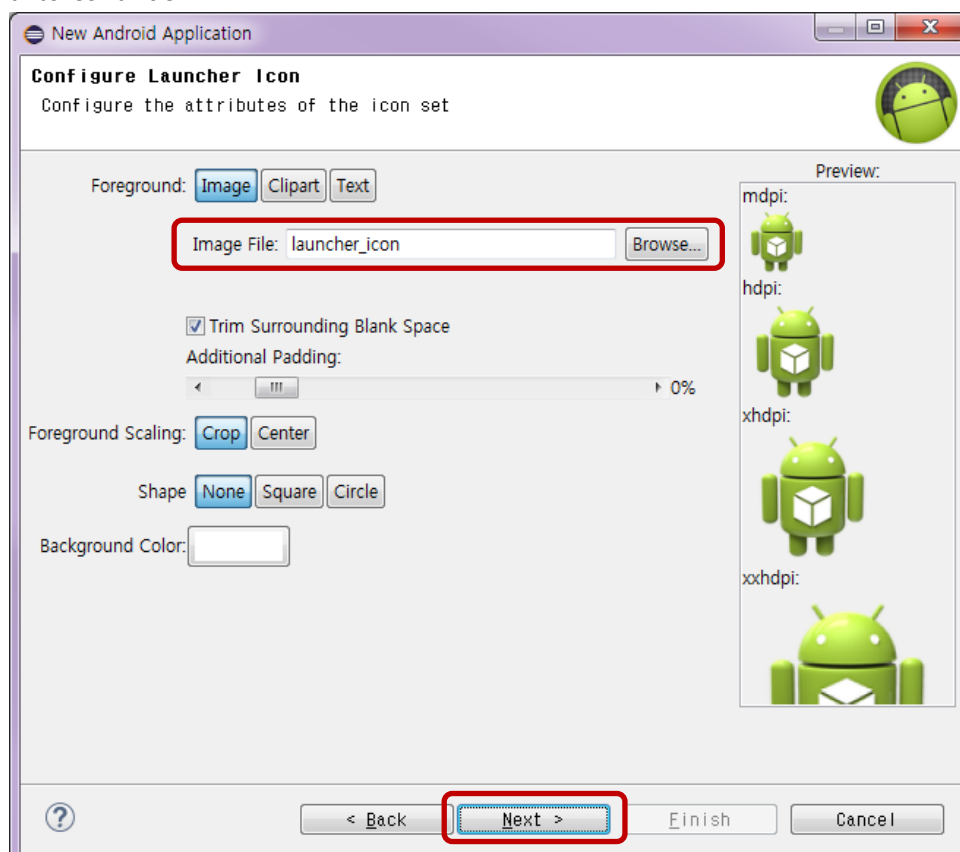
Select file → New → Android application project from the file menu.



When new android application window is displayed, fill in "Application Name", "Project Name" And "Package Name". Select "Minimum required SDK" "Target SDK"" Compile with" and Theme Then Click "Next" Button to proceed.

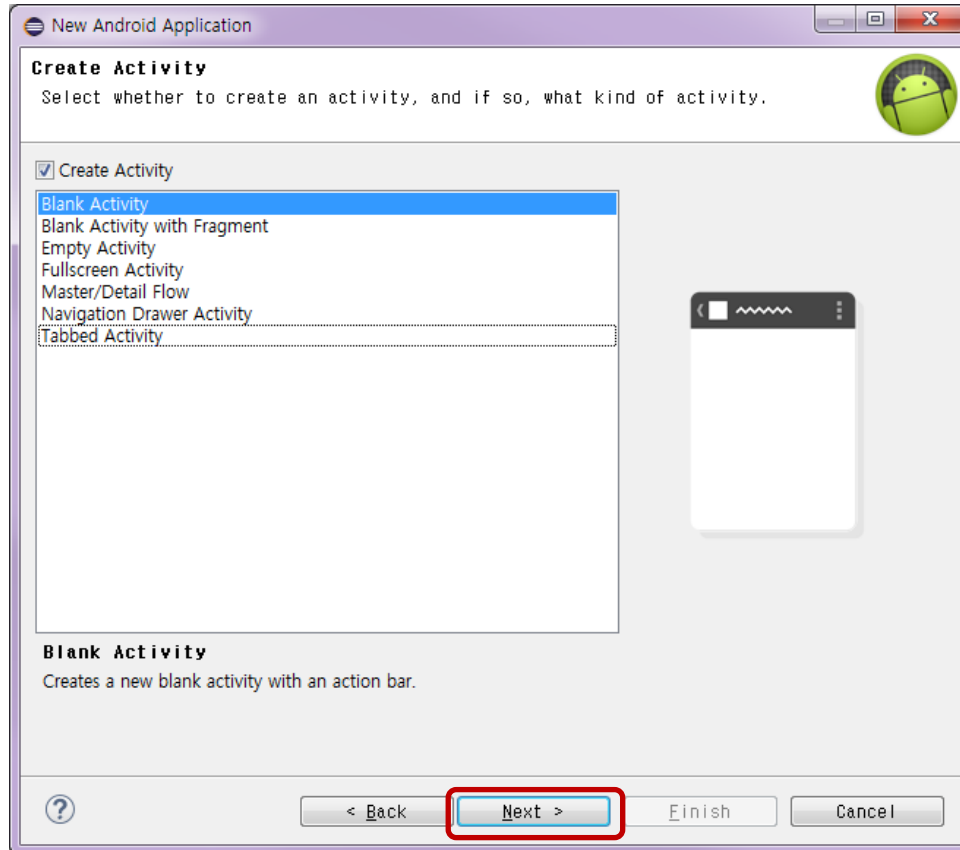


Click 'Next' to continue



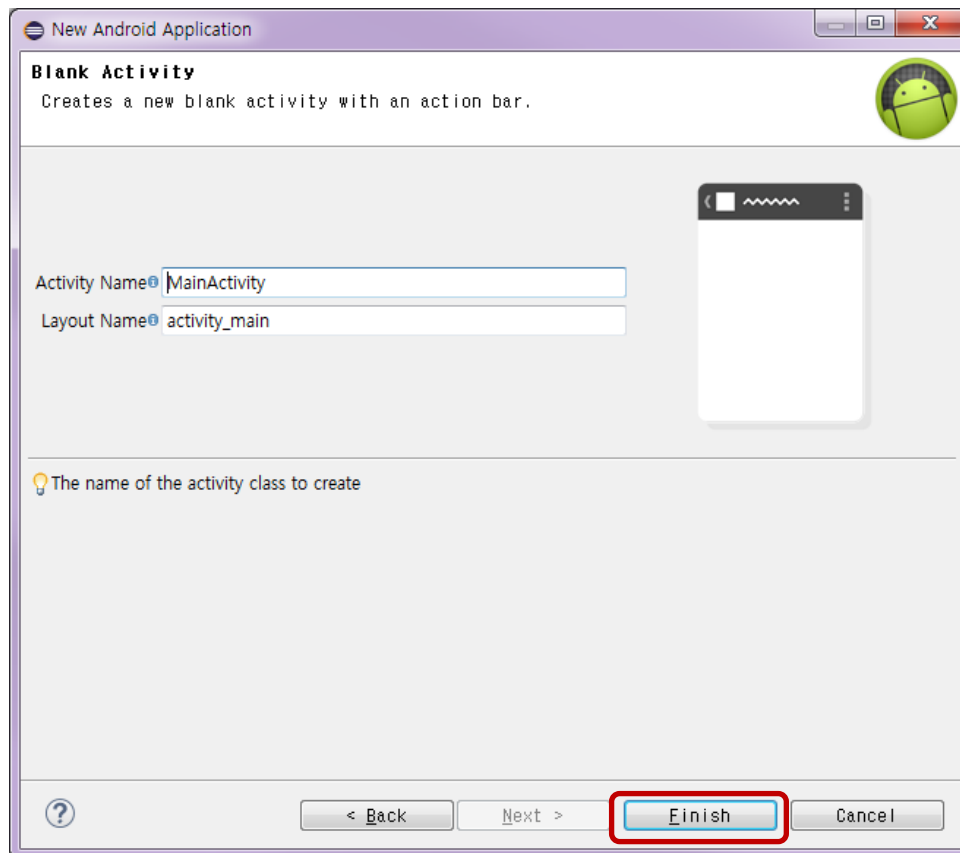
 All That Identification		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

Select the Icon that would be used in Application and Click "Next" to Proceed.

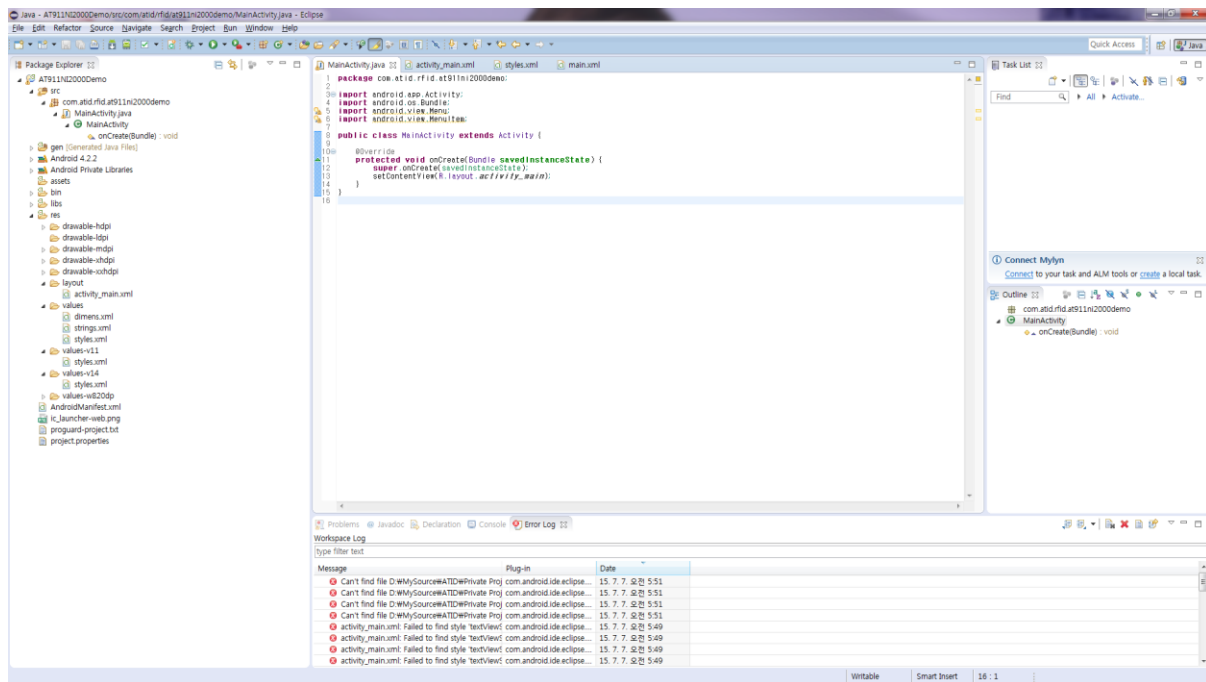


Screen is for selection Theme in the application.

Select Theme then Click "Next"



Fill in Main Activity name in the Activity name and Layout Name,
After fill in the name then click "Finish" then finish creating the project surely.

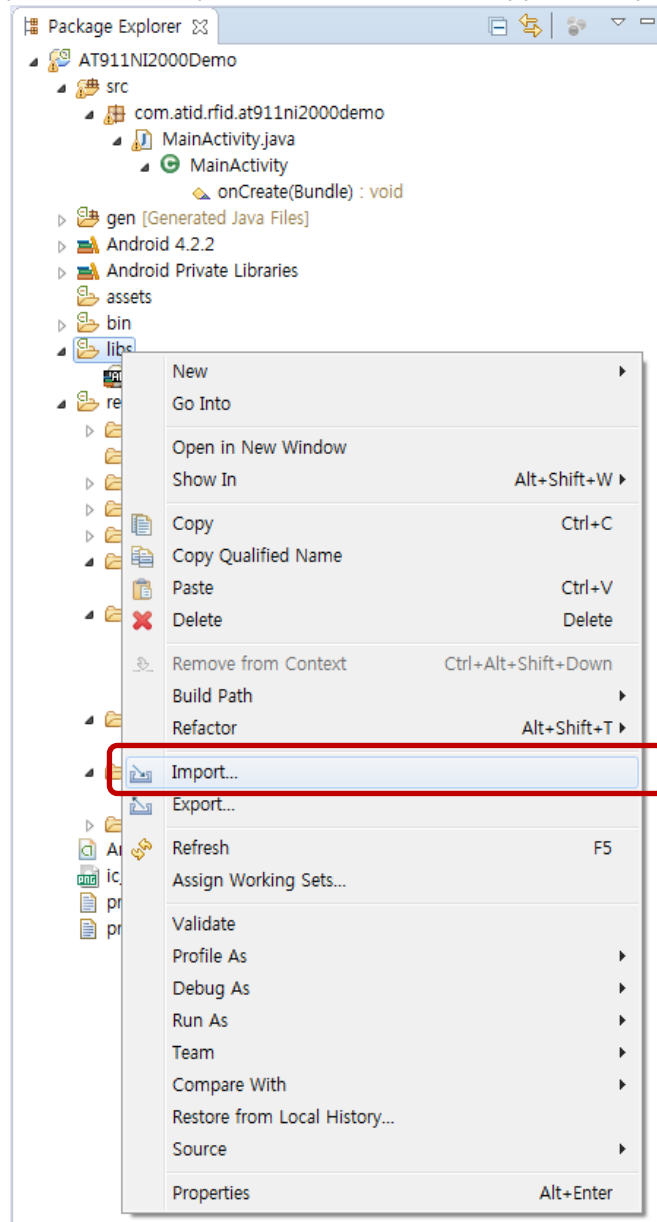


Eclipse screen is shown as above when the project creation is completed.

		RFID Programming Guide					
All That Identification		Android Developer Guide				Company	ATID Co.,Ltd
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

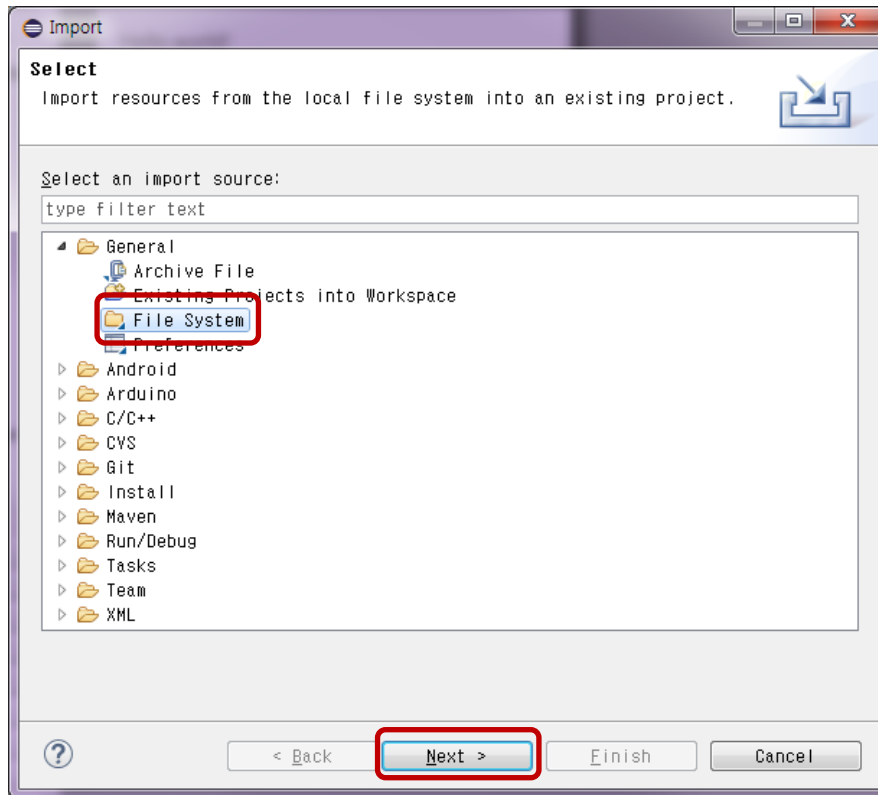
2.2. Development Environment

RFID SDK jar file is required to develop AT911N RFID android application program.

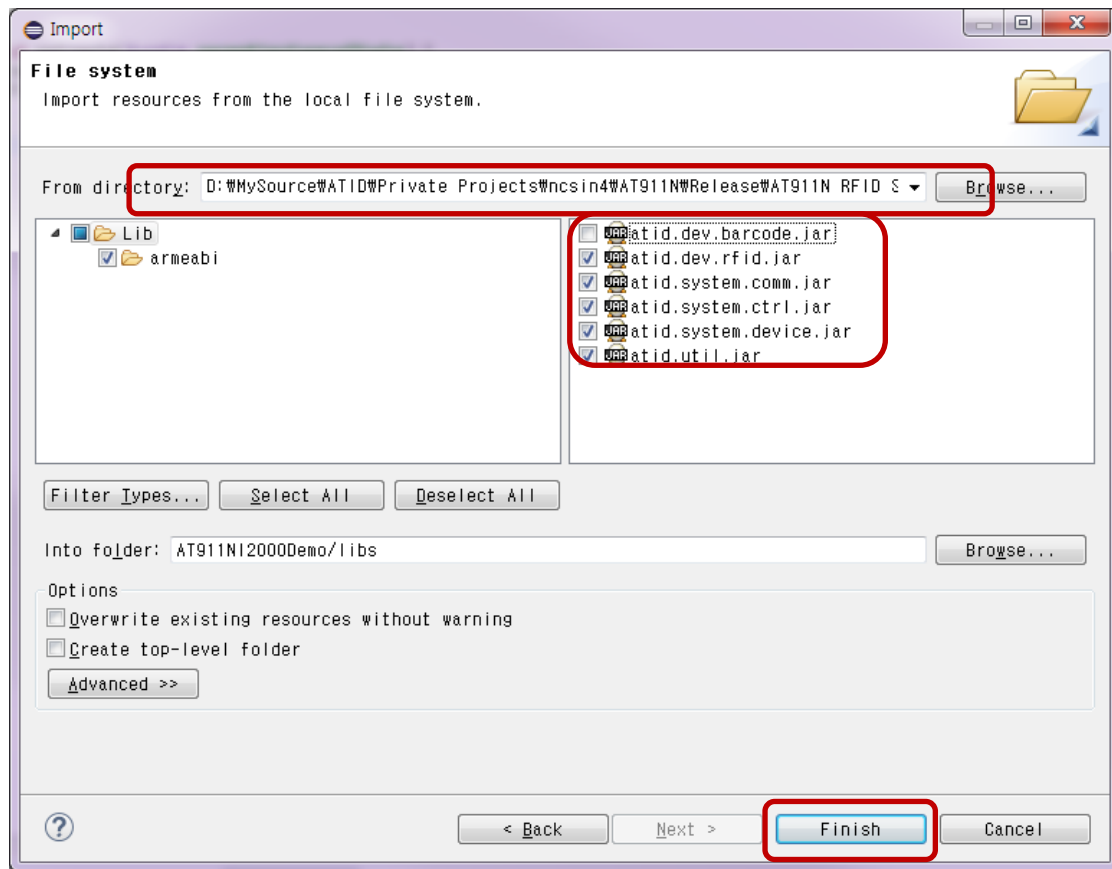


To import RFID SDK jar file, Select libs folder from package Explorer, Click on the right folder and select "import" from the Pop-up menu.

		RFID Programming Guide					
All That Identification		Android Developer Guide				Company	ATID Co.,Ltd
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0



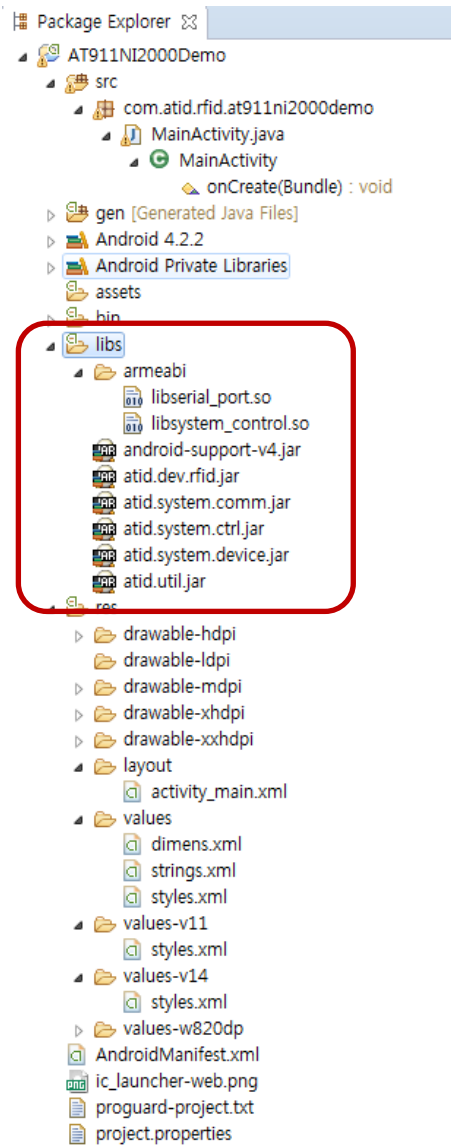
Show up the import dialog box, Select "General" → file system and Click next to proceed.



Select android folder under Library folder and you can see atid.dev.rfid.jar file.

Check all the checkboxes and Click "Finish".

atid.dev.rfid.jar file will be copied to libs folder created by the project.



You can see the Jar file is copied to package explorer.

3. Programing Guide

3.1. Create and Synchronization

3.1.1. Create Reader

To use AT911N RFID via AT911N RFID SDK Library, the instance of ATRfidReader must be created first. Crating the instance can be checked from the onCreate method in MainActivity.java file of the sample source.

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

    // Skip

    if ((mReader = ATRfidManager.getInstance()) == null) {
        // Skip
    }

    // Skip
}
```

3.1.2. EventListener register and release

For receiving answer from ATRfidReader instance, to implement RfidReaderEventListener interface on Activity of response and receiver user.

To register Listener via SetEventListener of Activity which is implemented interface,

And To release of registration via removeEventListener.

If Application has different Activity, this processing is needed several times,

For simple processing to register and release, to process Activity's onResume, onPause method.

```
@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();
}
```

		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

3.1.3. To manager Sleep/Wakeup

To manage RFID module's power and resource to compare using Activity or not.

Call `ATRfidManager.wakeUp()` from Activity's `OnStar` and `ATRfidManager.sleep()` from Activity's `OnStop`

Wake up and Sleep must call with Pair, after calling Sleep, if wake up is not calling, Module could not working normally.

```

@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();
}

```

		RFID Programming Guide					
Android Developer Guide					Company	ATID Co.,Ltd	
Doc.		Writer	SDK Team	Date	2017-02-22	Version	V2.0

3.2. Event Handler

After Register EventListener, interface function is executed which is implemented by user Whenever event is coming.

The category of the event is as below.

Module Operation (onReaderStateChanged), Inventory Operation (onReaderReadTag),

Access Operation (onReaderResult)

onReaderStateChange method is for receiving event by module's connection condition.

Refer to sample program for details as 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);
}
```

onReaderReadTag method is working that event when it is reading the Tag as inventory operation. It is implemented as below, and sending 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 event is for checking about Read/Write/Lock/ Kill and etc's result of Access operation

```
@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);
}
```

3.3. Action Operation

3.3.1. Start Operation

For inventory of TAG, the details on using inventory6cTag, inventory6bTag, readEpc6cTag, readEpc6bTag methods

For the way of using inventory method, is implemented in InventoryActivity.java file.

```
// 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()");
}
```

3.3.2. Stop Operation

For stop inventory or access order, call stop.

Refer to 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()");
}
```

3.4. End

After finish to use RFID and close completely,

Calling `ATRfidManager.onDestroy()` and make sure clear RFID related function.

Below is implemented part of `MainActivity.java`'s `onDestroy()`

```
@Override
protected void onDestroy() {

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

    // Wake Unlock
    SysUtil.wakeUnLock();

    saveConfig();

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

    super.onDestroy();
}
```