CS710S Java Android API and Programmer's Guide

Version 1.0

2022 07 07

Copyright Convergence Systems Limited 2022, All Rights Reserved

Chapter 1: Release Notes

Dates	Release	Description
2022 07 07	1.0	Initial release

Copyright Convergence Systems Limited 2022, All Rights Reserved

Chapter 2: Content

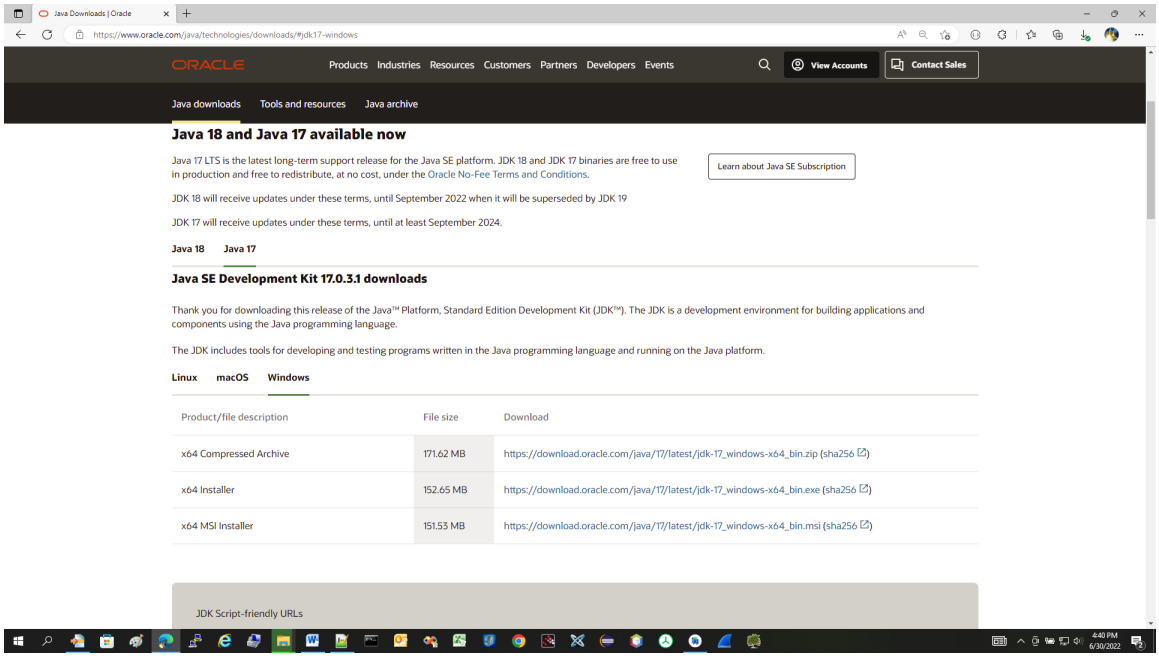
Copyright Convergence Systems Limited 2022, All Rights Reserved

Chapter 3: Programming Environment

3.1 System Requirements

1) The demo project is developed under following computer configuration
    a. Android Studio Chipmunk | 2021.2.1
    b. Embedded JDK version 11.0.12 or above (Oracle Java 17 or above)
    c. Microsoft Windows 10 Pro
    d. 64-bit operating system, x64-based processor
    e. Intel(R) Core™ i5-4200U CPU @ 1.60GHz, 229MHz
    f. Physical Memory(RAM) 8.00 GB
    g. Hard-disk 500 GB with more than 20 GB free space
2) Download and install Java JDK from http://www.oracle.com/technetwork/java/javase/downloads/index.html
    a. Select "JDK Download" -> Java 17 -> Windows
    b. Select 'jdk-17_windows-x64_bin.exe or .msi' for Windows x64

Copyright Convergence Systems Limited 2022, All Rights Reserved



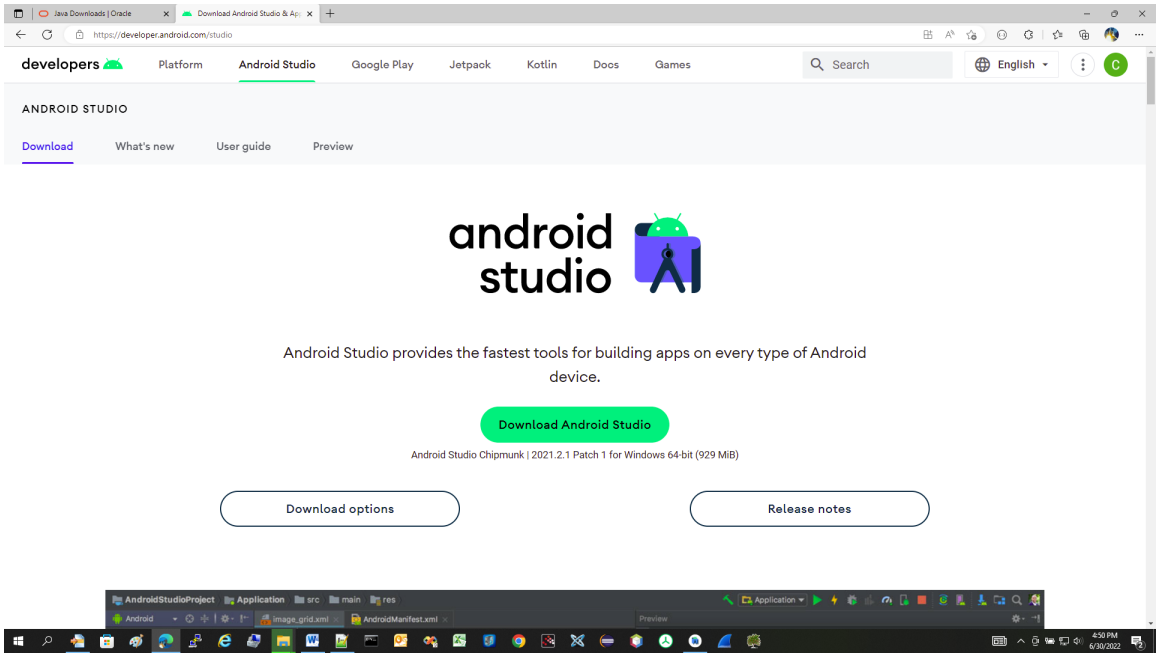
c. Run the downloaded installer and continue

d. Check the presence of Java in command prompt by typing “java –version”

3) Download and install latest Android Studio from https://developer.android.com/studio/

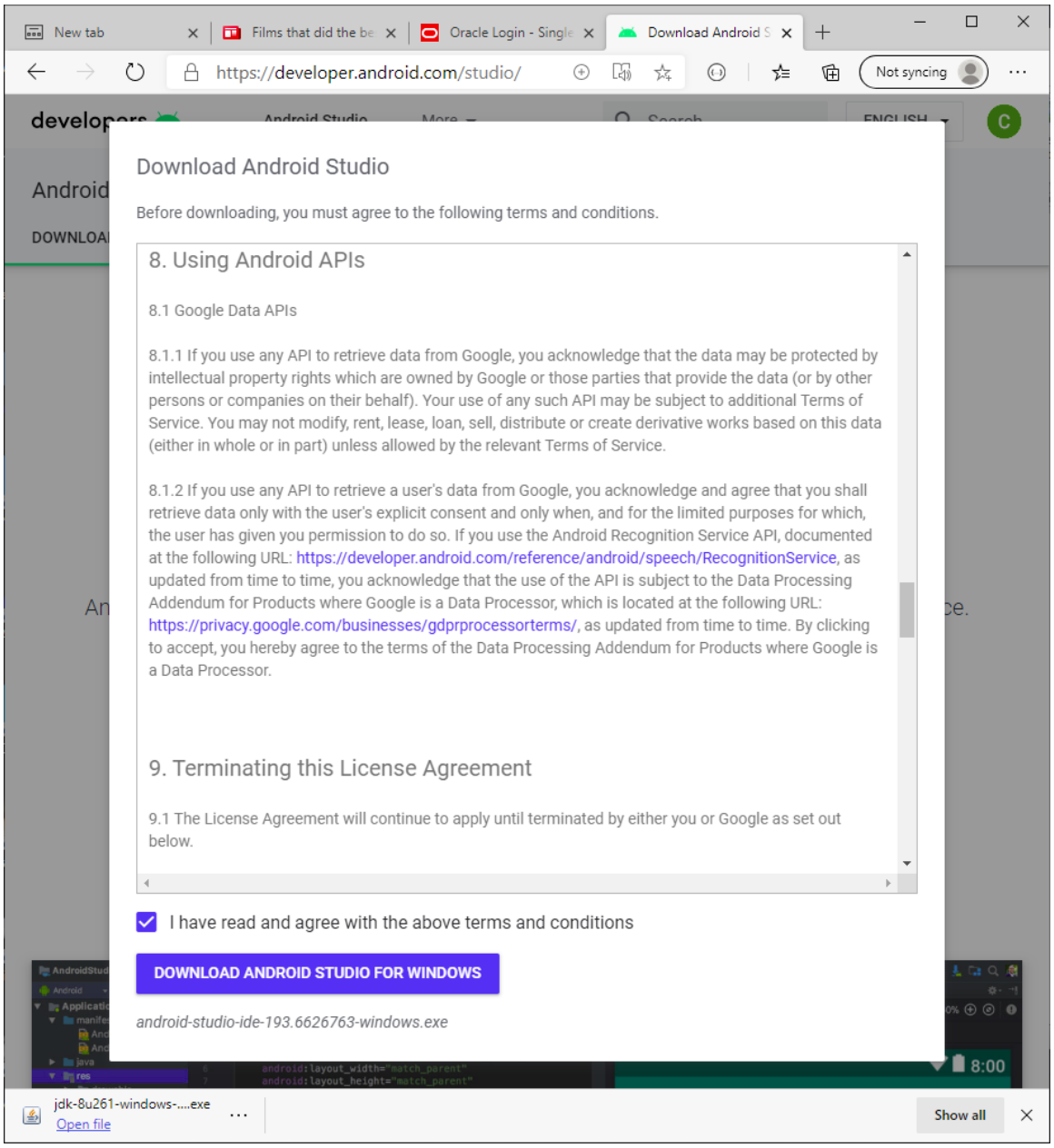
a. Press “Download Android Studio”

Copyright Convergence Systems Limited 2022, All Rights Reserved



b. Tick the box “I have read and agree the above items and conditions” and press “Download Android Studio for Windows”

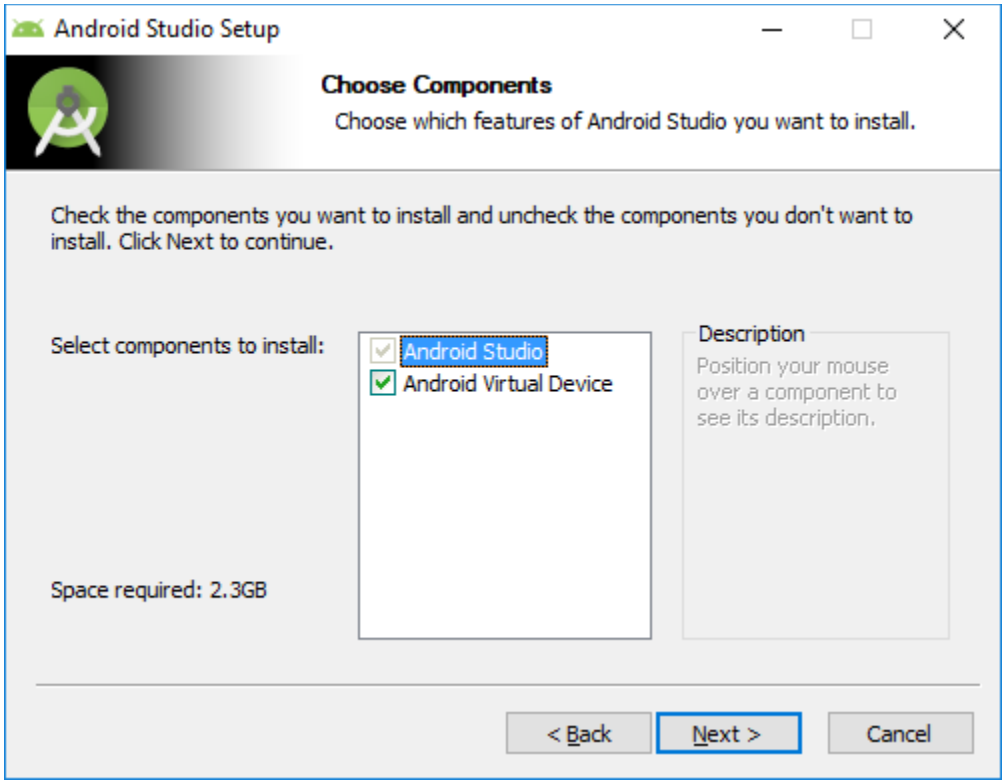
Copyright Convergence Systems Limited 2022, All Rights Reserved



c. “Run” the downloaded installer and continue up to the “Welcome to Android Studio Setup” page.

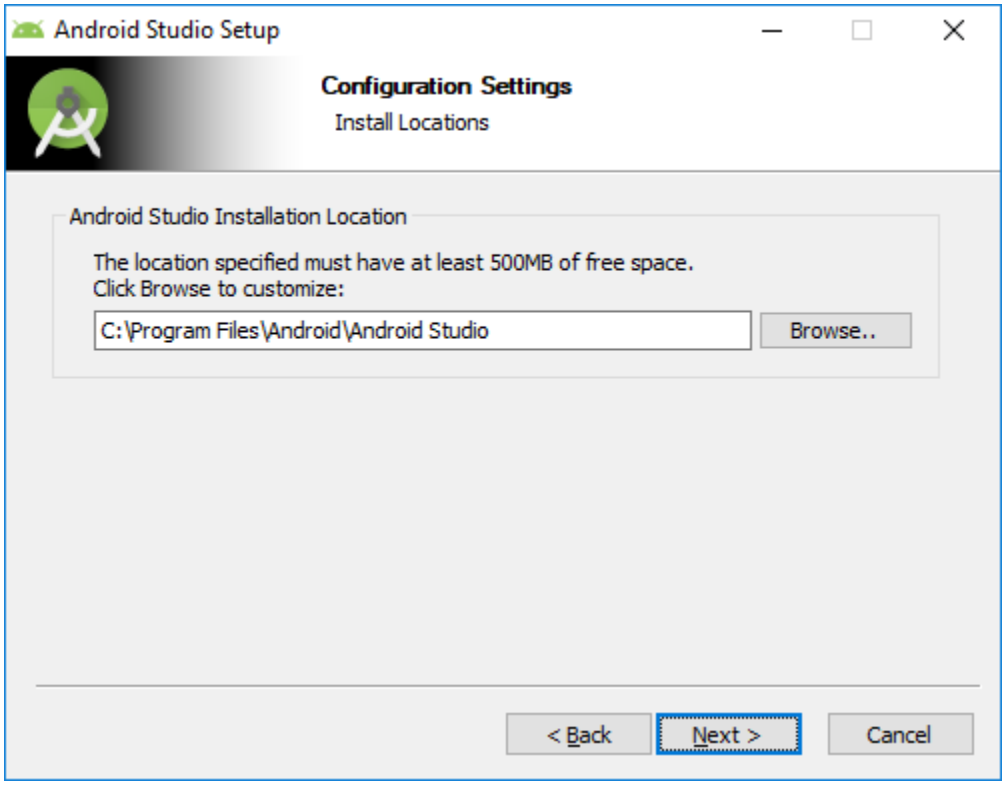
Copyright Convergence Systems Limited 2022, All Rights Reserved

d. Tick “Android Virtual Device” and Select “Next”

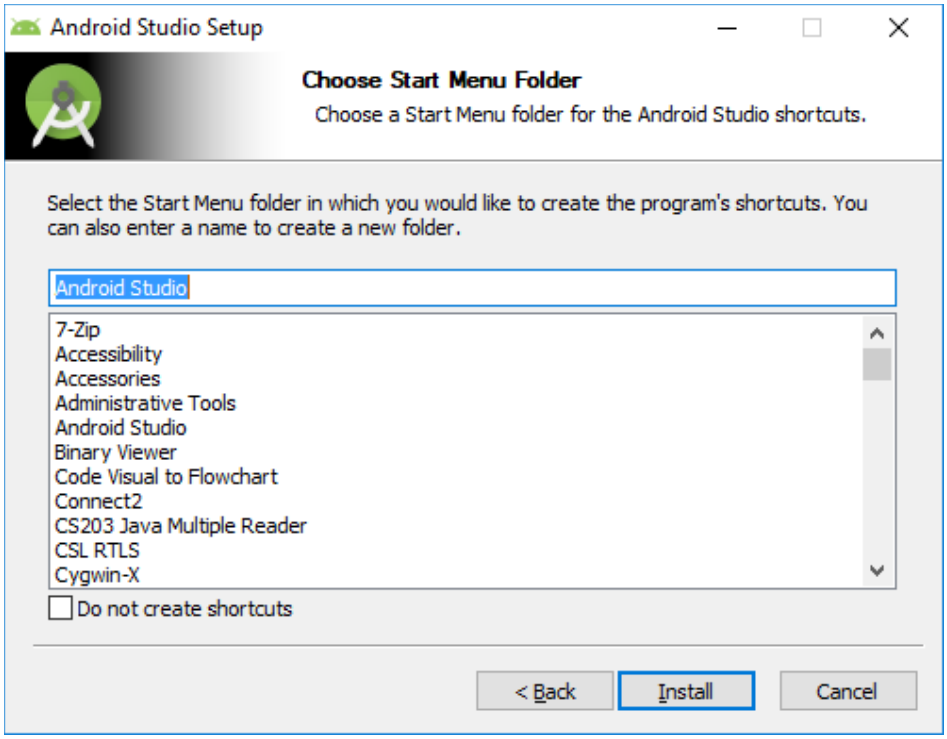


e. Select “Next”

Copyright Convergence Systems Limited 2022, All Rights Reserved

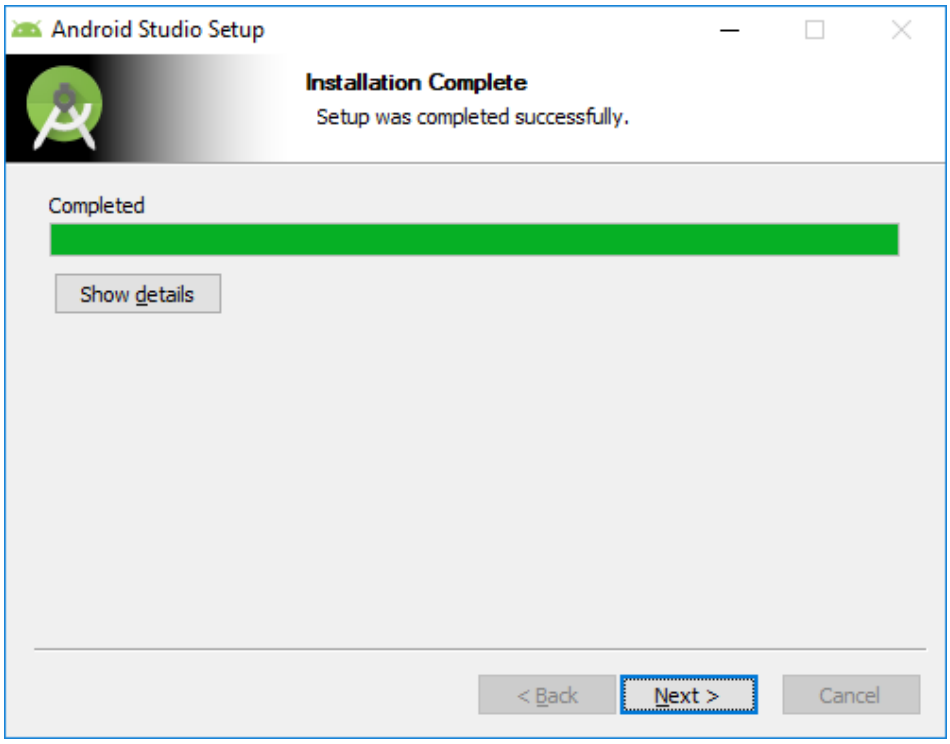


f. Select “Install” and Wait

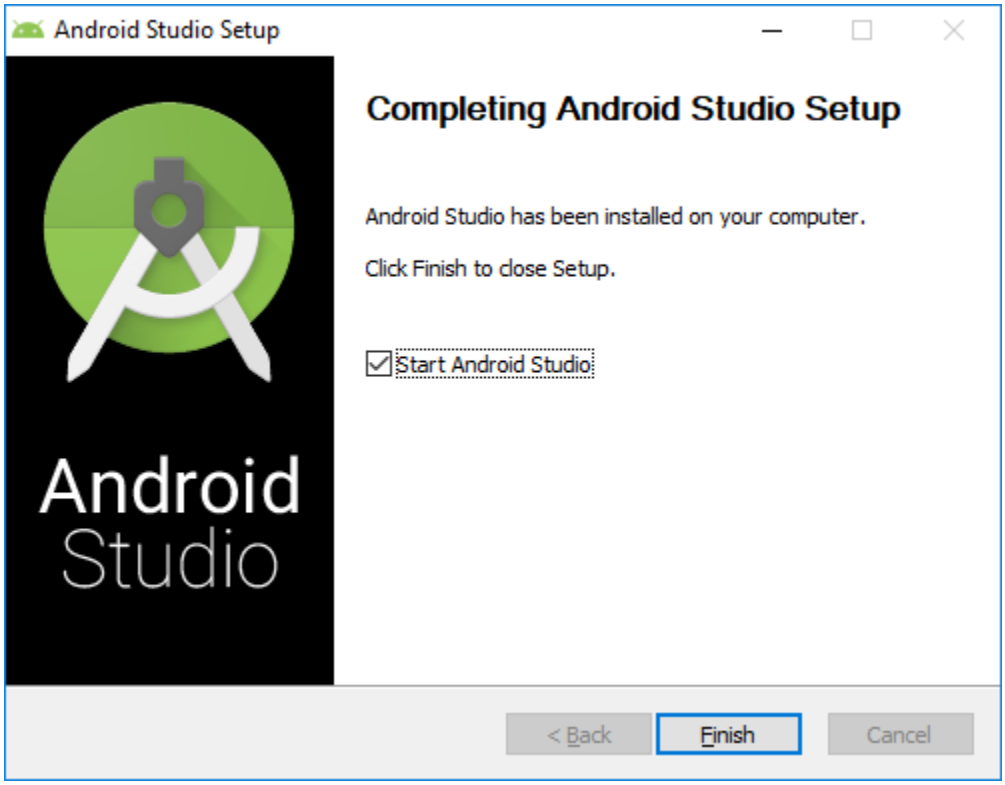


g. Select “Next”

Copyright Convergence Systems Limited 2022, All Rights Reserved

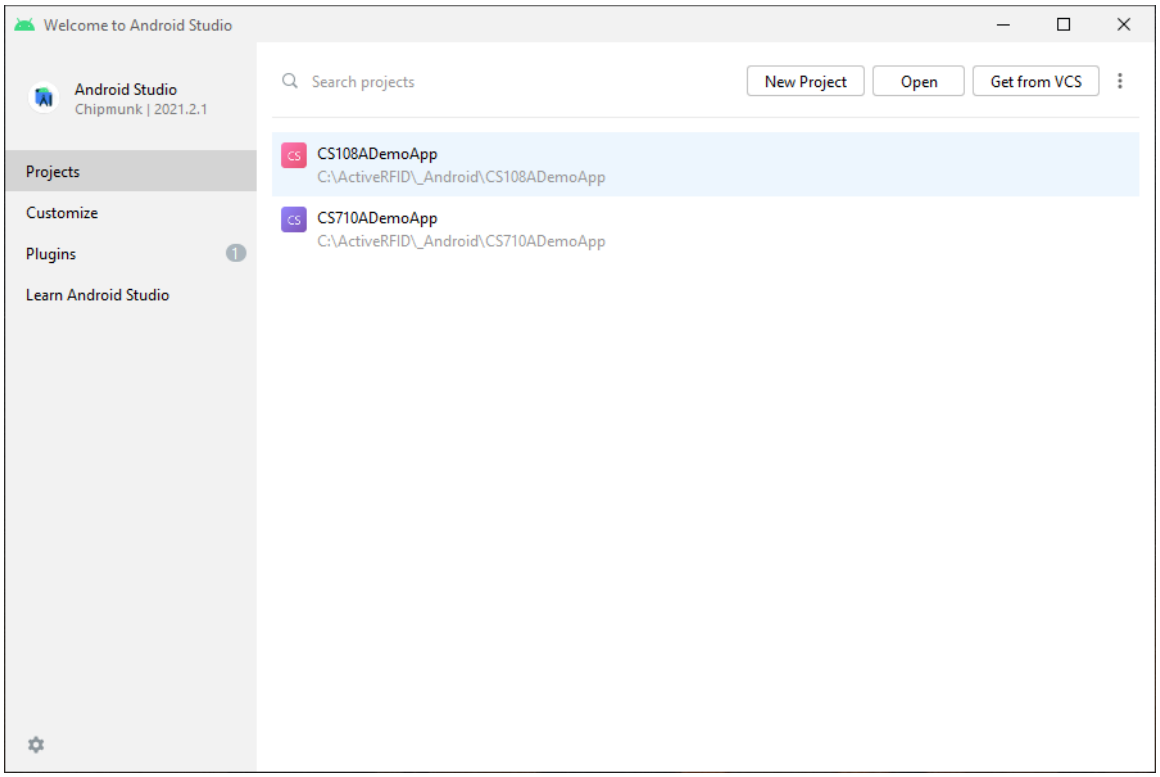


h. Tick “Start Android Studio” and Press “Finish”



i. Final display

Copyright Convergence Systems Limited 2022, All Rights Reserved



Copyright Convergence Systems Limited 2022, All Rights Reserved

3.2 CS710S Project Download

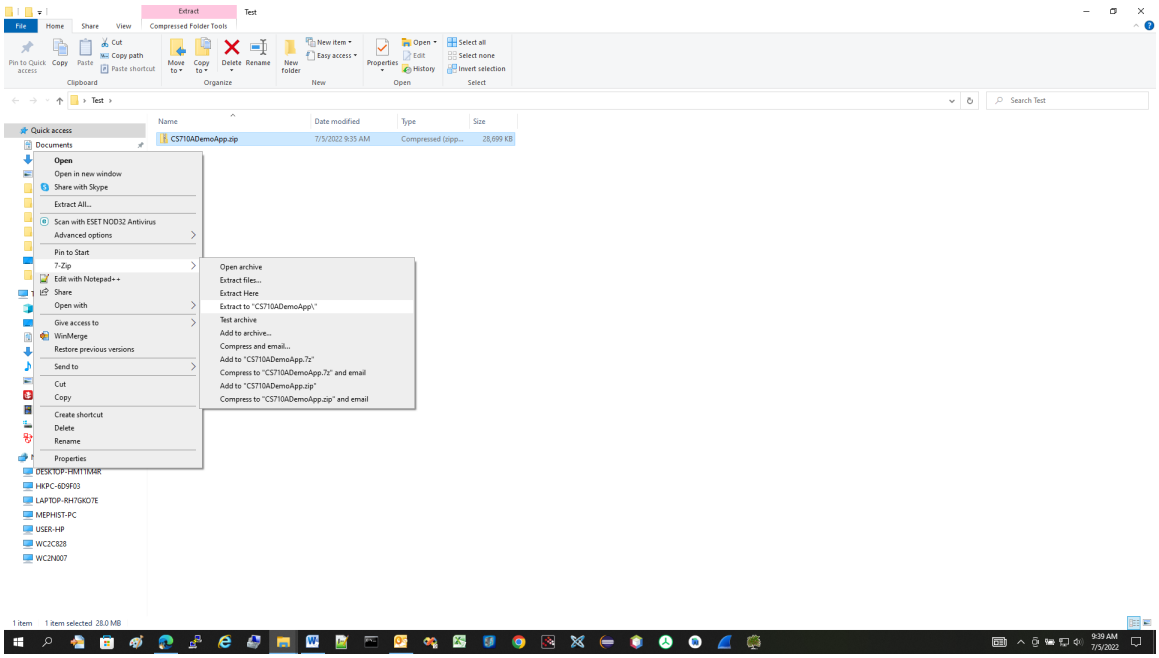
1) Please download Demo code from https://www.convergence.com.hk/downloads/cs710S/

   i. Select "CS710S Android Java Bluetooth Demo App and SDK" to go to GitHub

   ii. Select "Code" -> "Download ZIP" to download the latest code.

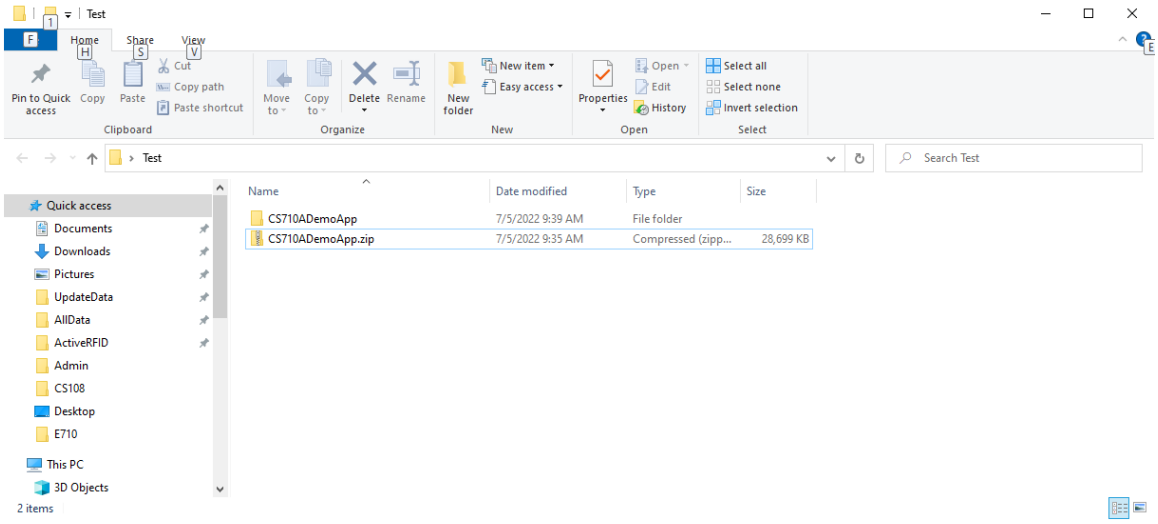
   iii. Find your saved file and move the zip file in your project sub-directory

   iv. Unzip the zip file



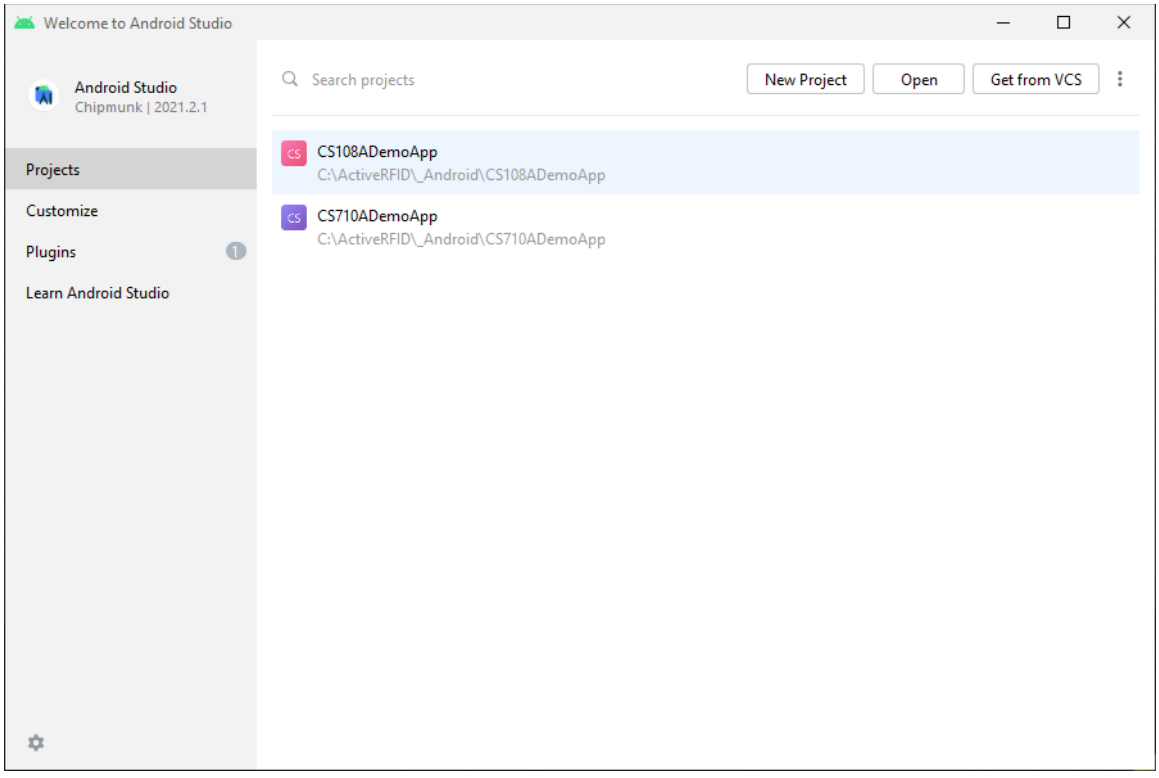
   v. Final display

Copyright Convergence Systems Limited 2022, All Rights Reserved



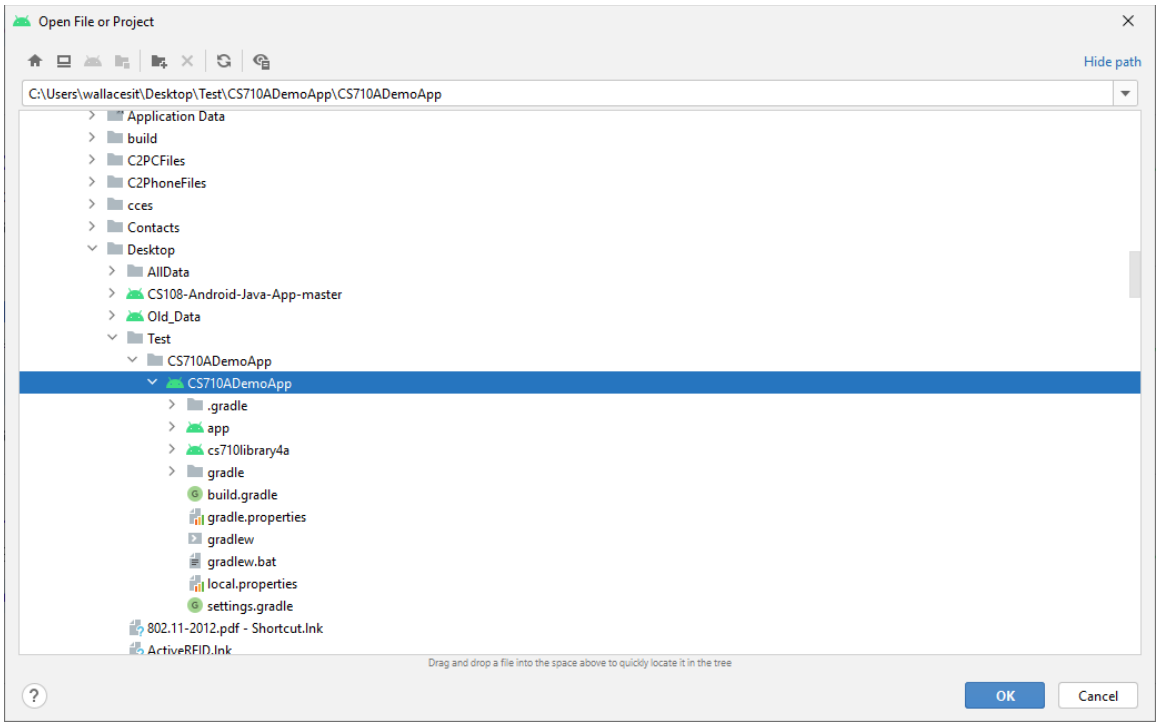
2) Android Studio SDK platform setup for the CS710ADemo project

i. Open android Studio

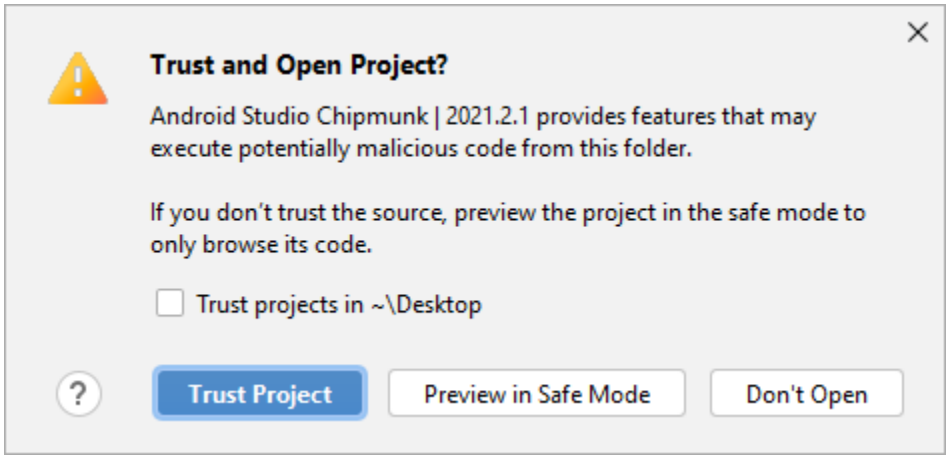


ii. Select “Projects” -> “Open”, the unzipped “CS710ADemoApp” project. And Select “OK”

Copyright Convergence Systems Limited 2022, All Rights Reserved

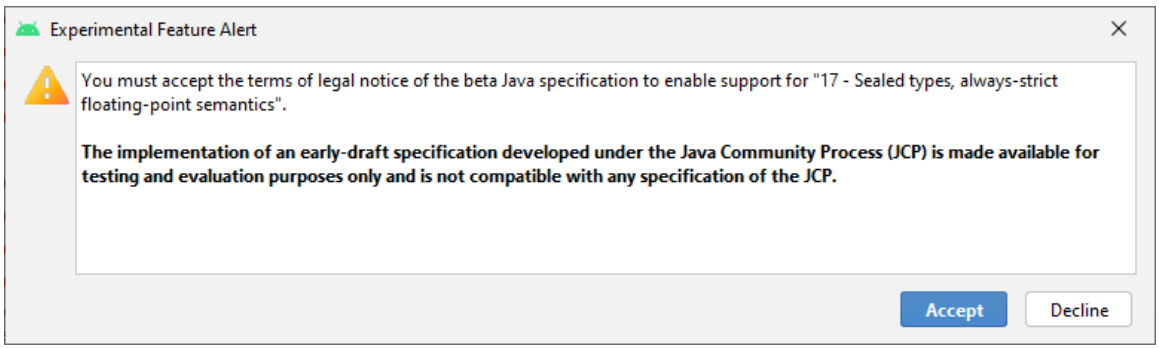


iii. Following pop up may appear. Select “Trust Project”

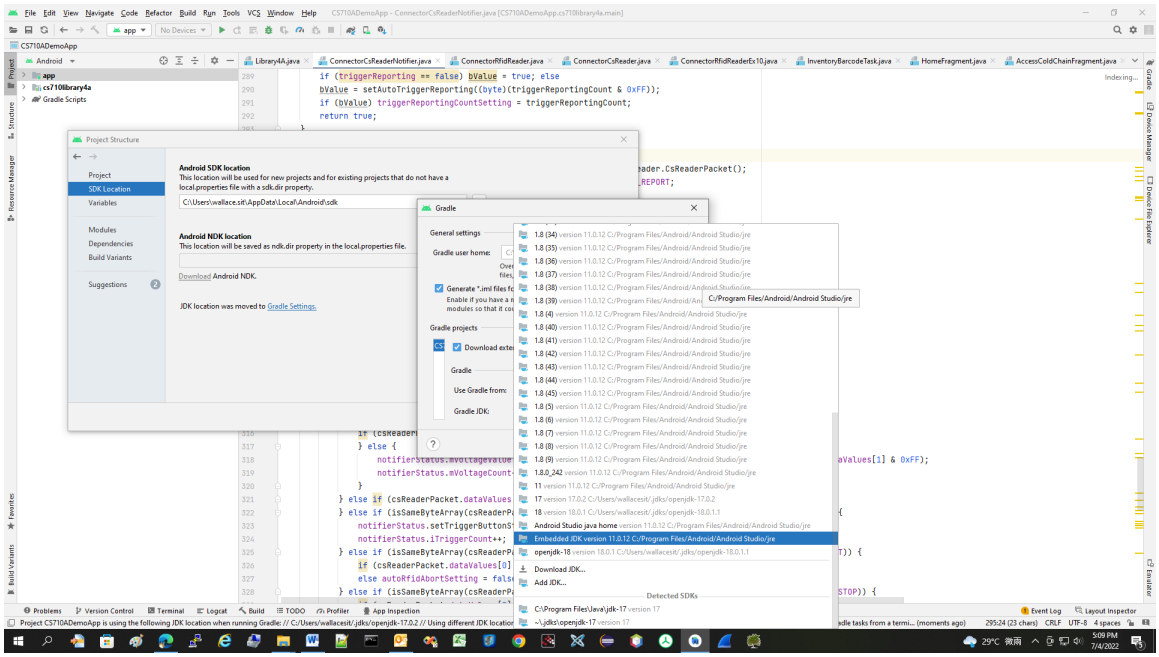


iv. Following pop up may appear. Select “Accept”

Copyright Convergence Systems Limited 2022, All Rights Reserved

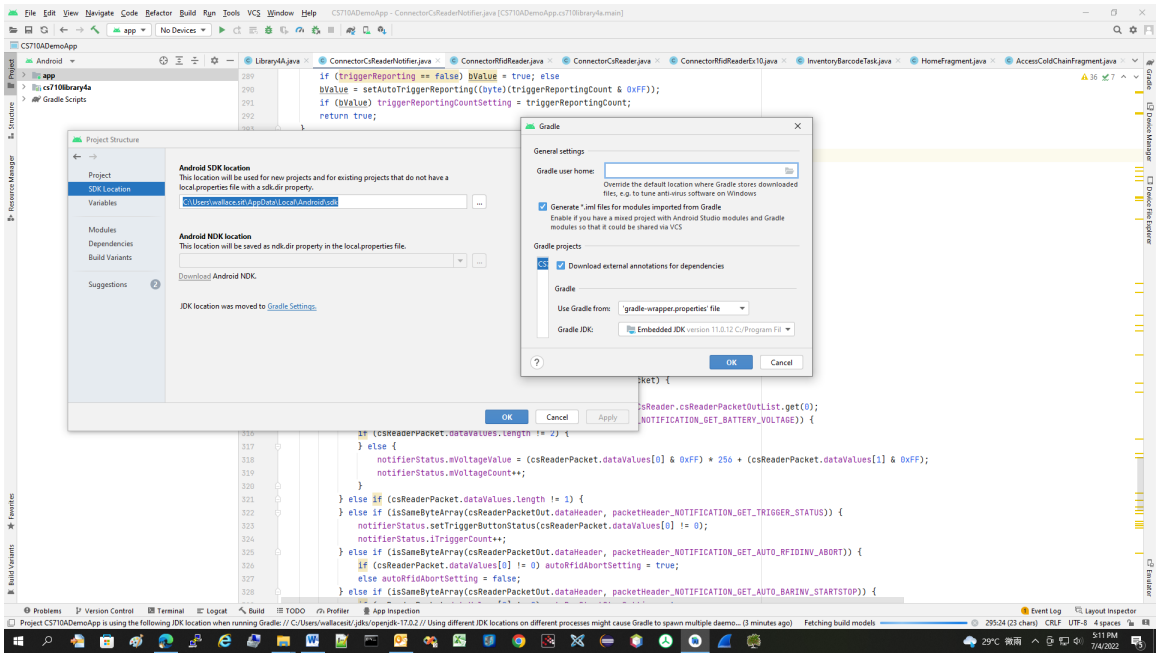


v. You may have error message about “Unsupported Java”. Select “File” -> “Project Structure” -> “SDK Location” -> “Gradle Settings” -> “Gradle JDK”. Select “Android Studio java home” or “Embedded JDK”. Or try “download JDK” to download latest JDK and select the updated JDK.

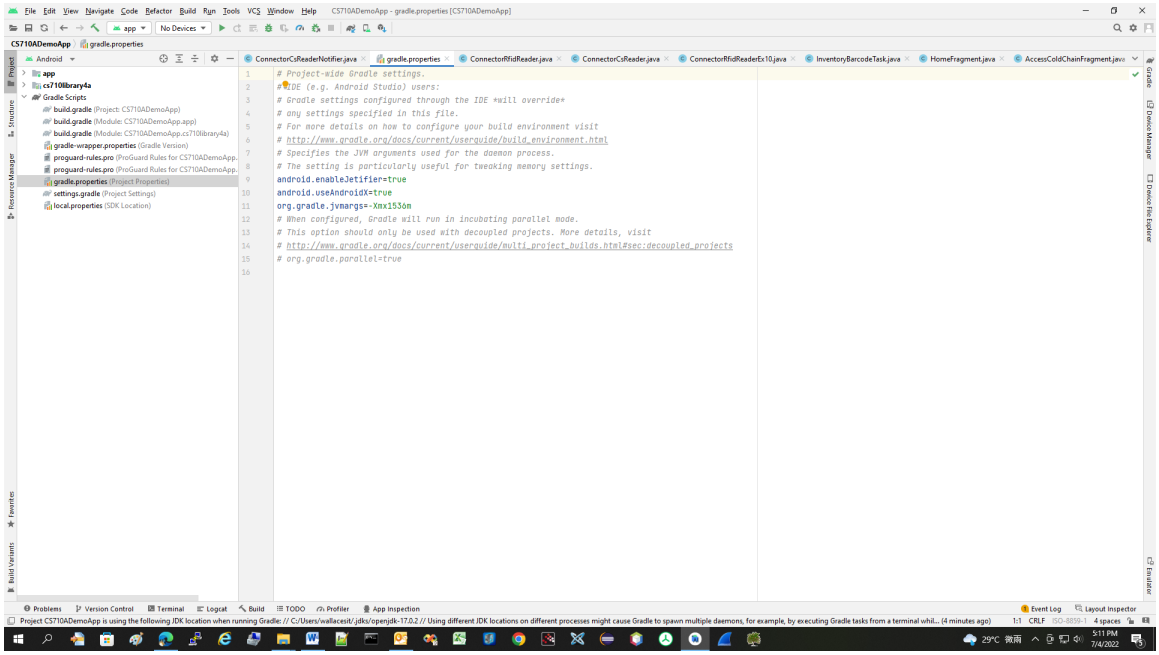


vi. Select “OK” and “OK” to close the settings

Copyright Convergence Systems Limited 2022, All Rights Reserved



vii. Final display



Copyright Convergence Systems Limited 2022, All Rights Reserved

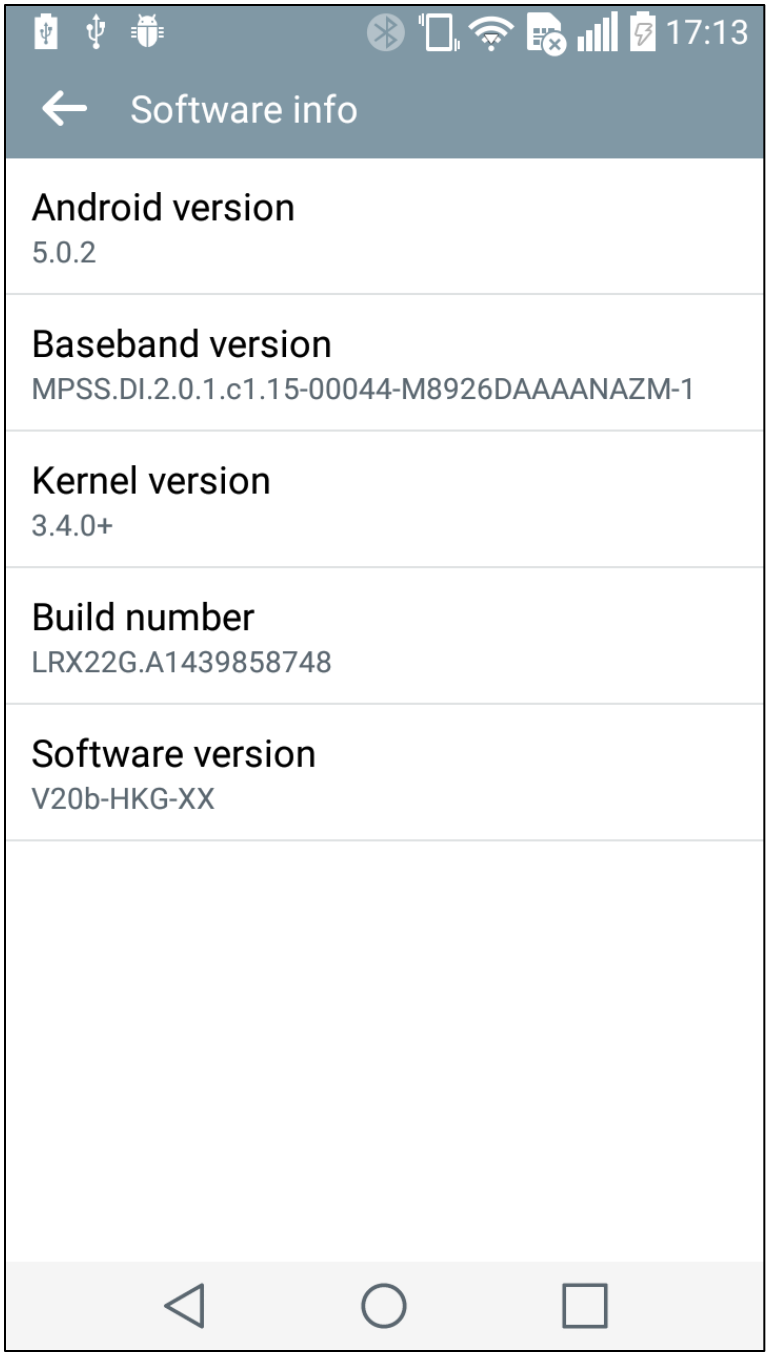
3.3 Debugging Setup

1) Android Debugging setup for Android phone

    i. Enable “Developer options” in android phone

        1. Go to “Setting” app -> “General” tab -> “About Phone” -> “Software Info”

Copyright Convergence Systems Limited 2022, All Rights Reserved

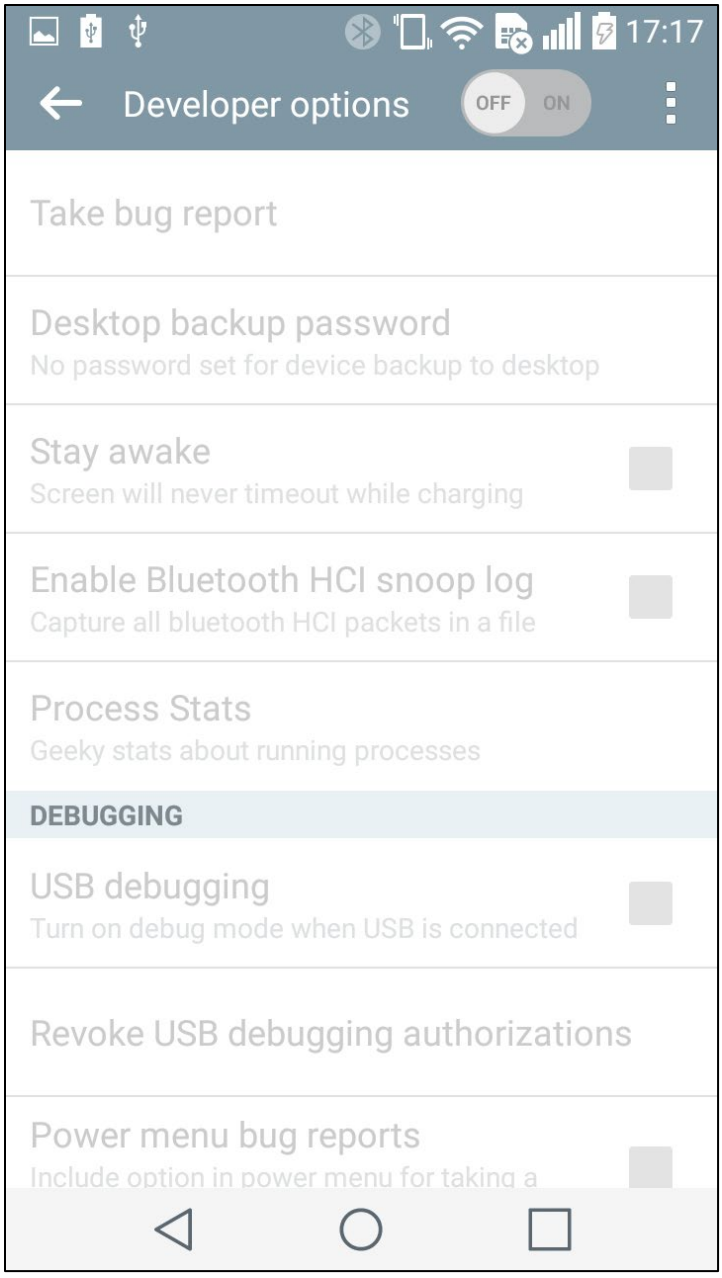


2. Press “Build number” more than 6 times until it shows that the phone is in “developer” mode

ii. Enable USB debugging

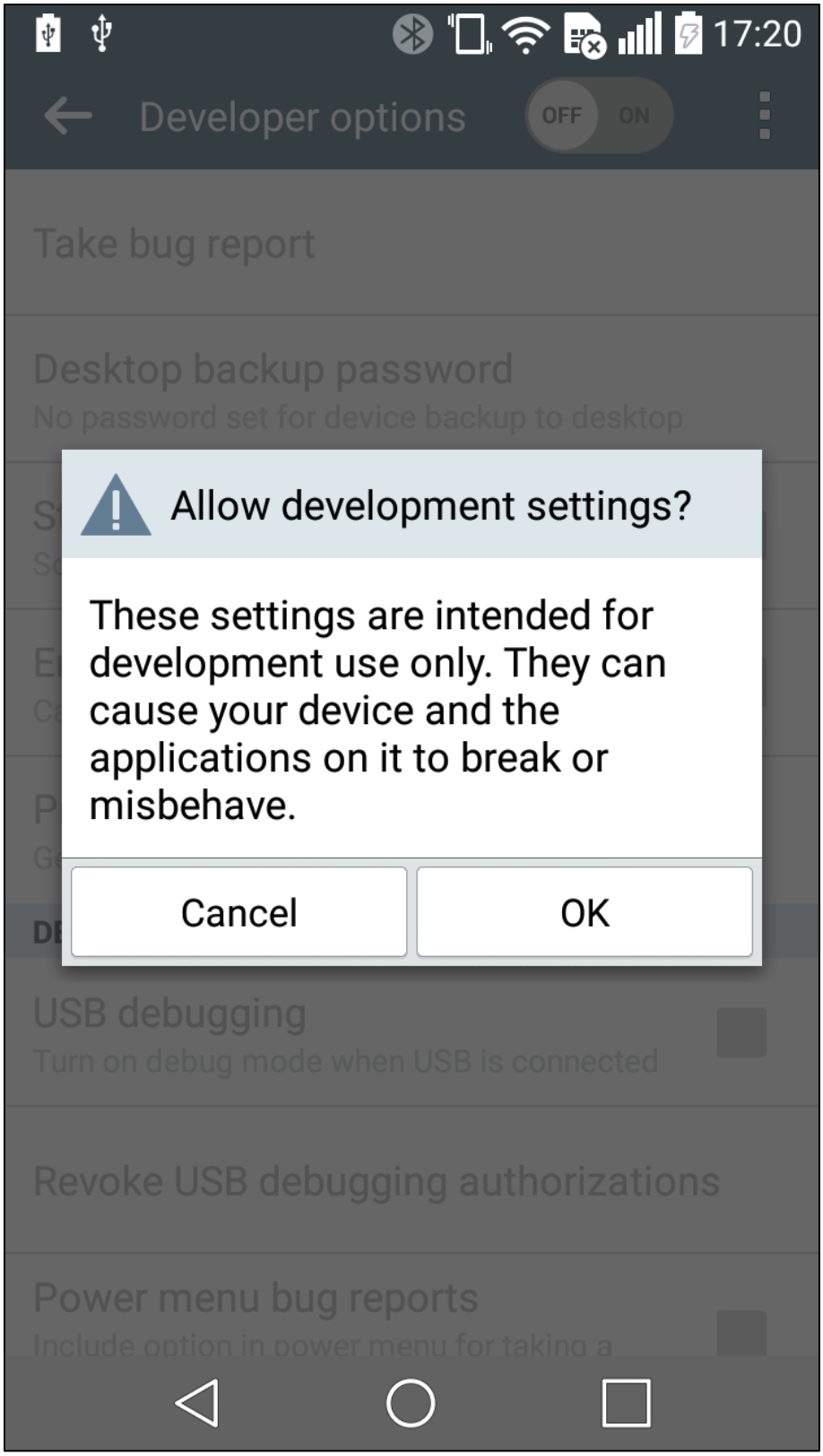
1. Go to “Setting” app -> “General” tab -> “Developer options”

Copyright Convergence Systems Limited 2022, All Rights Reserved



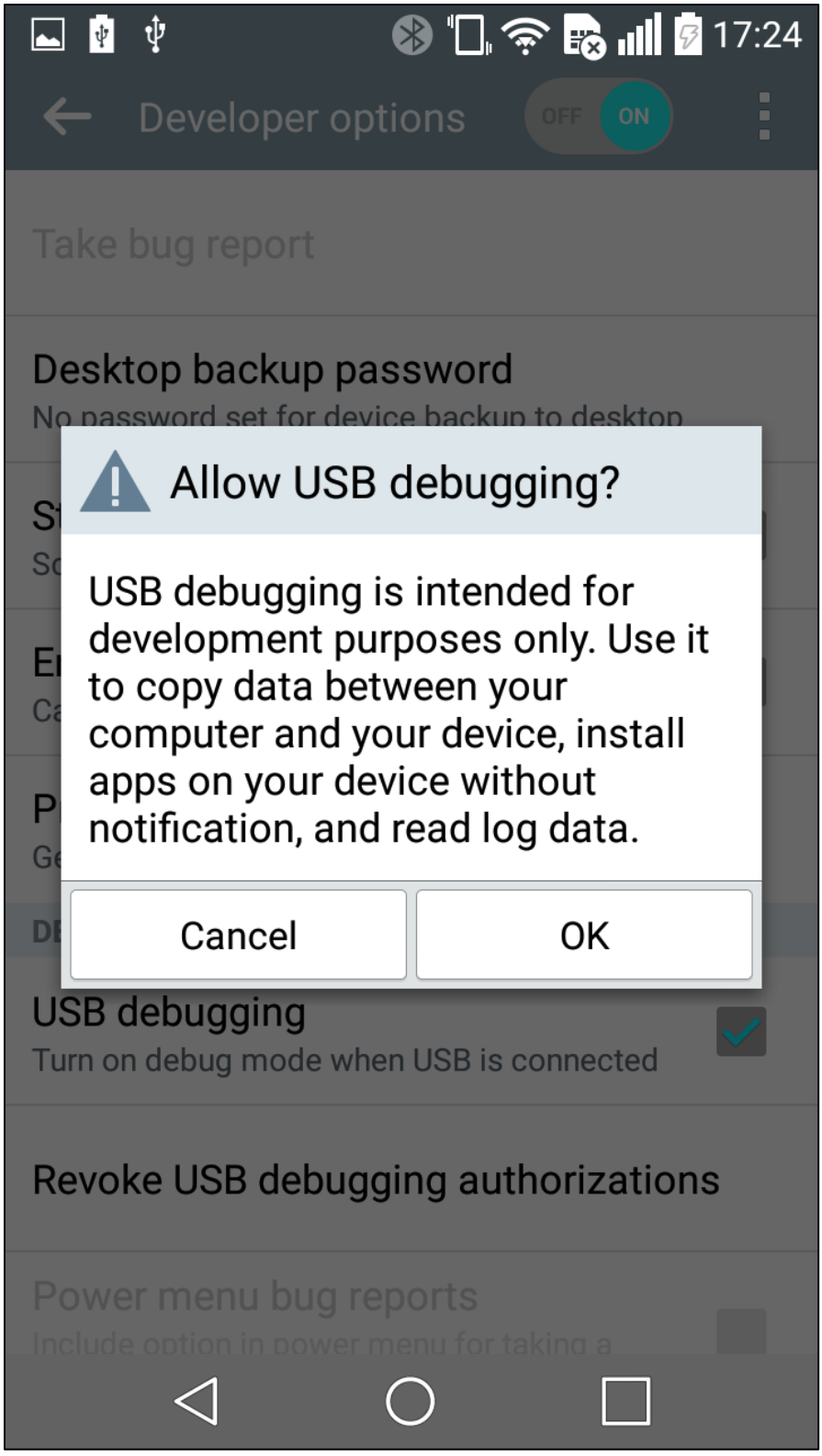
2. Turn on “Developer options” and select “OK” to allow development settings

Copyright Convergence Systems Limited 2022, All Rights Reserved



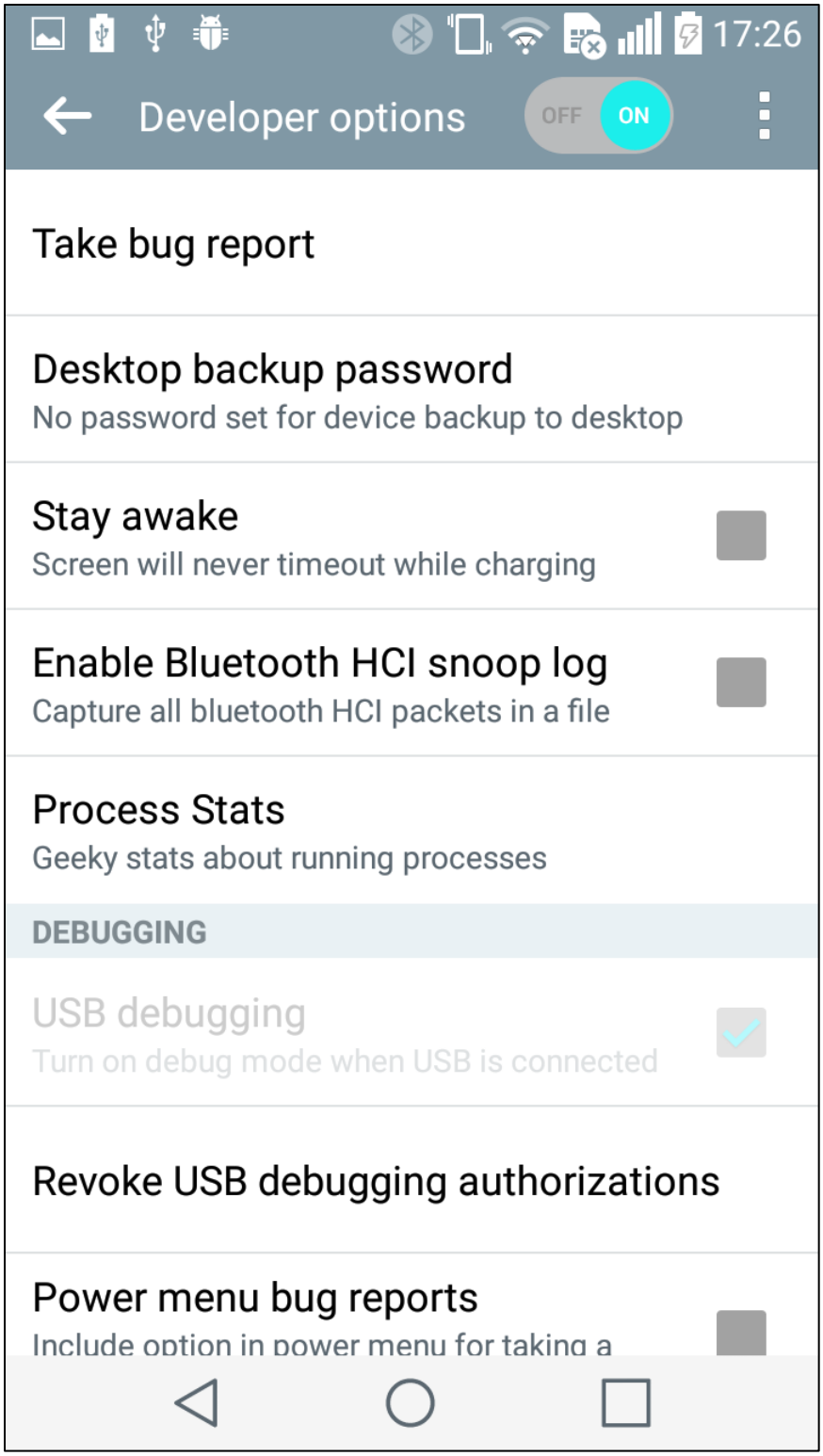
3. Select “USB debugging” and “OK” to allow USB debugging

Copyright Convergence Systems Limited 2022, All Rights Reserved



4. Final display which allows USB debugging

Copyright Convergence Systems Limited 2022, All Rights Reserved

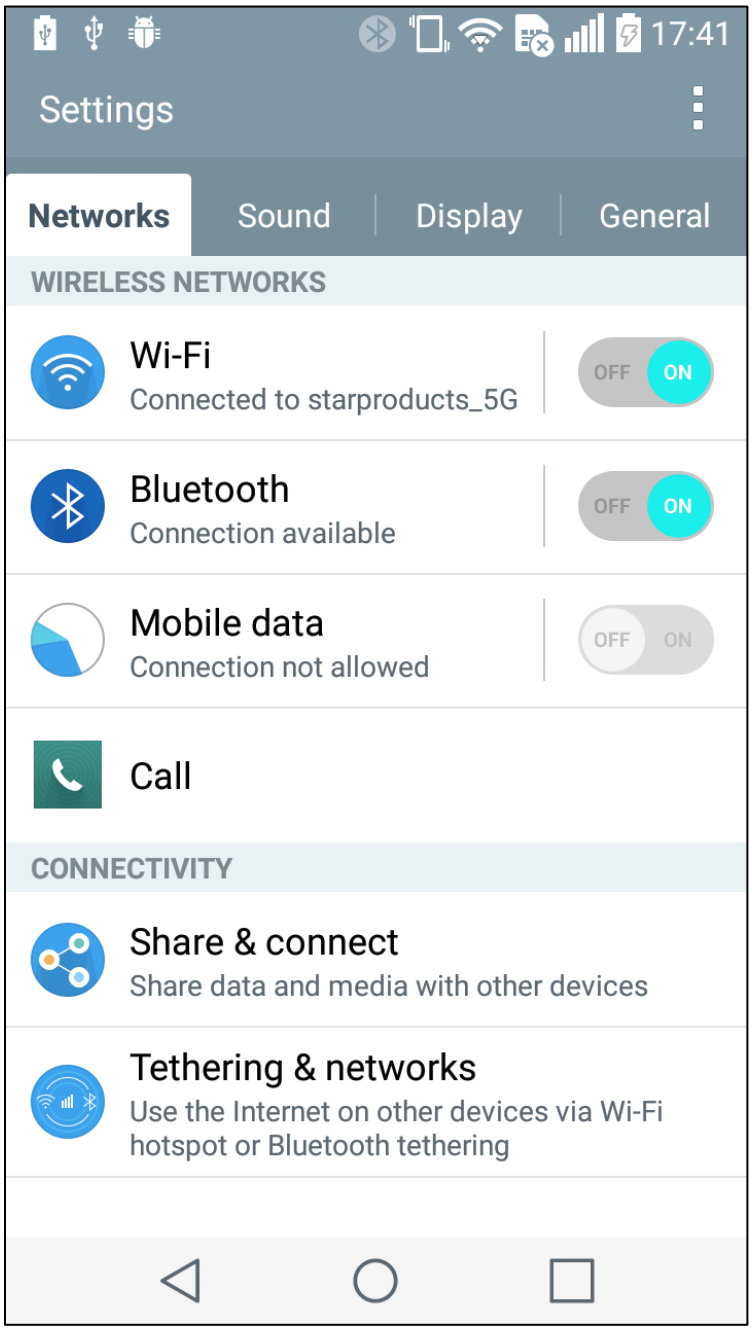


iii. Connect the USB cable between android phone and computer. There should have an  icon on the top besides  icon.

Copyright Convergence Systems Limited 2022, All Rights Reserved

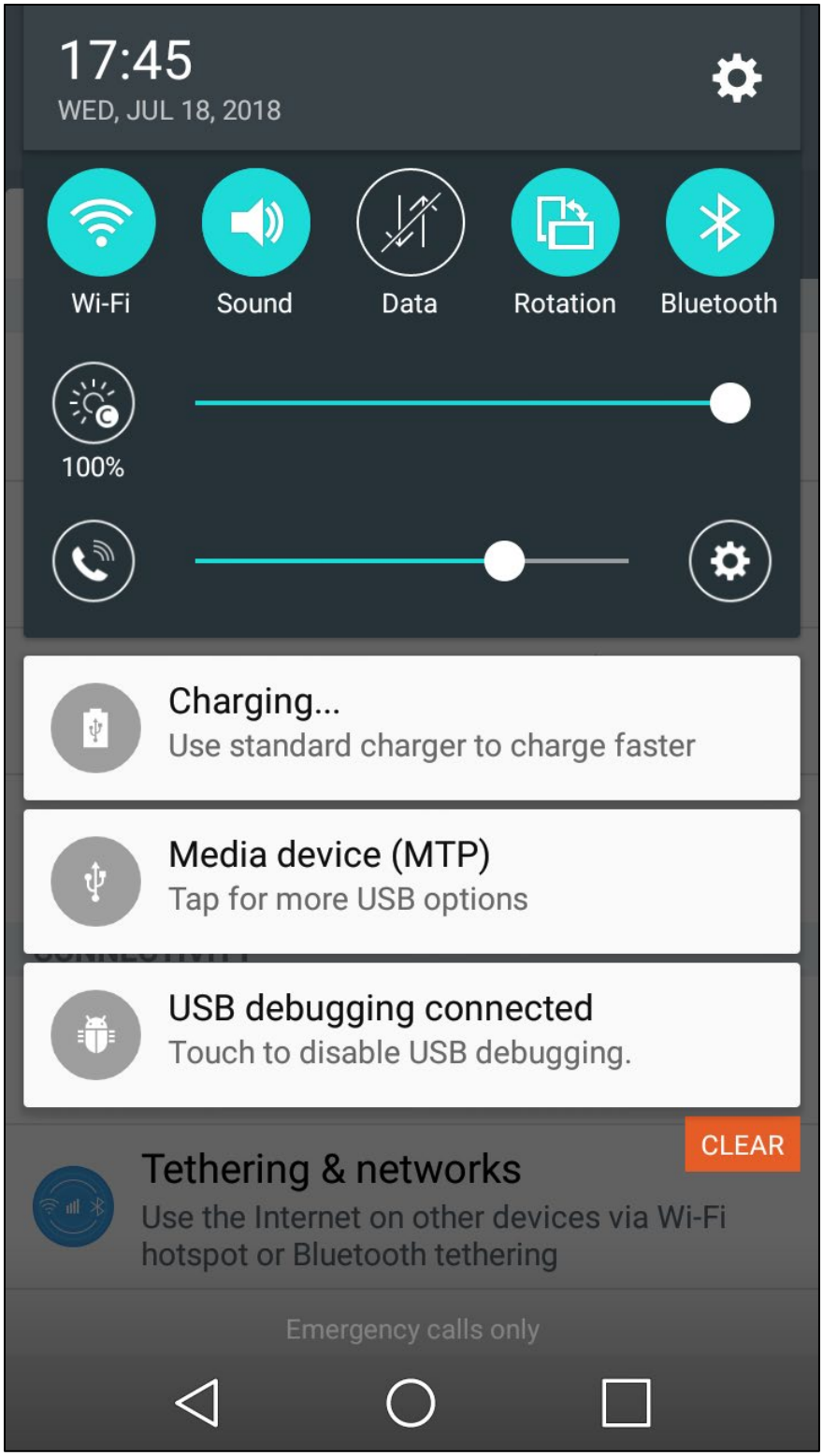
iv. Enable MTP mode for USB PC Connection setting

1. Connect the USB cable between android phone and computer



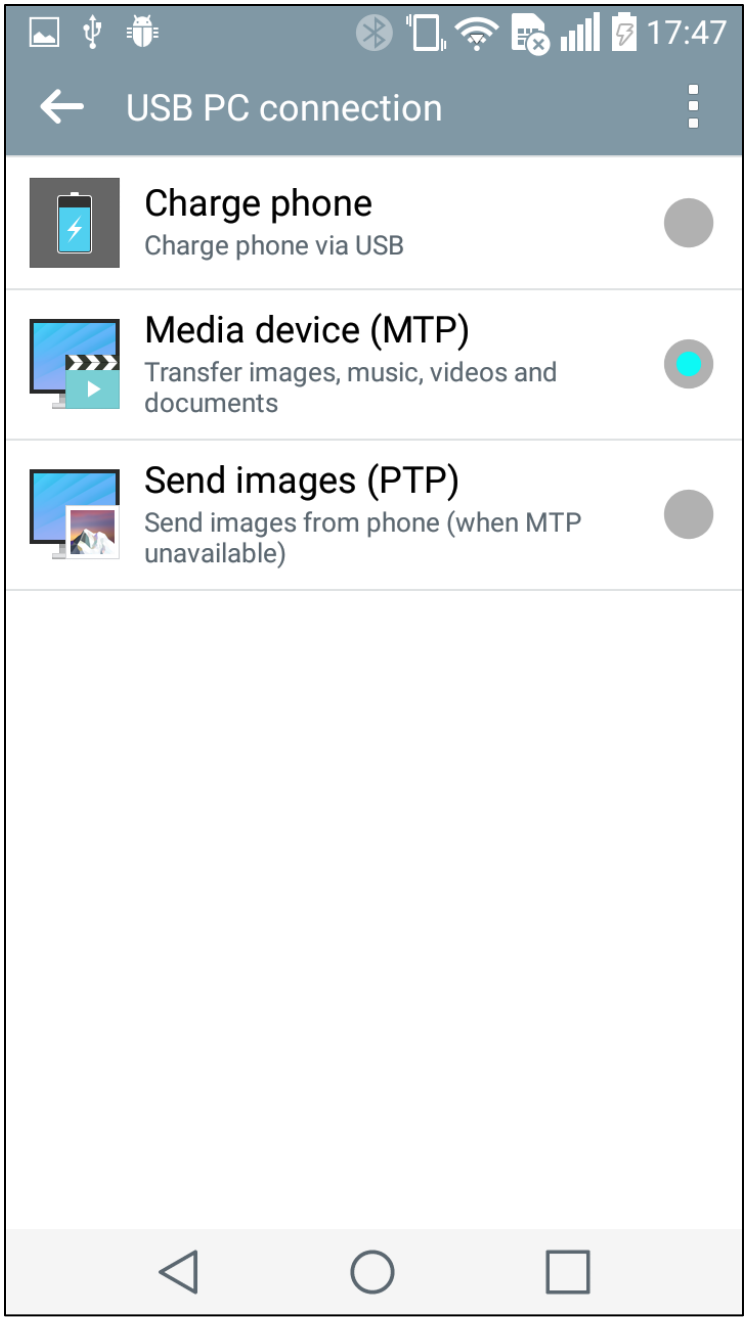
2. Swipe from top line downwards

Copyright Convergence Systems Limited 2022, All Rights Reserved



3. Select “Tap for more USB options”

Copyright Convergence Systems Limited 2022, All Rights Reserved



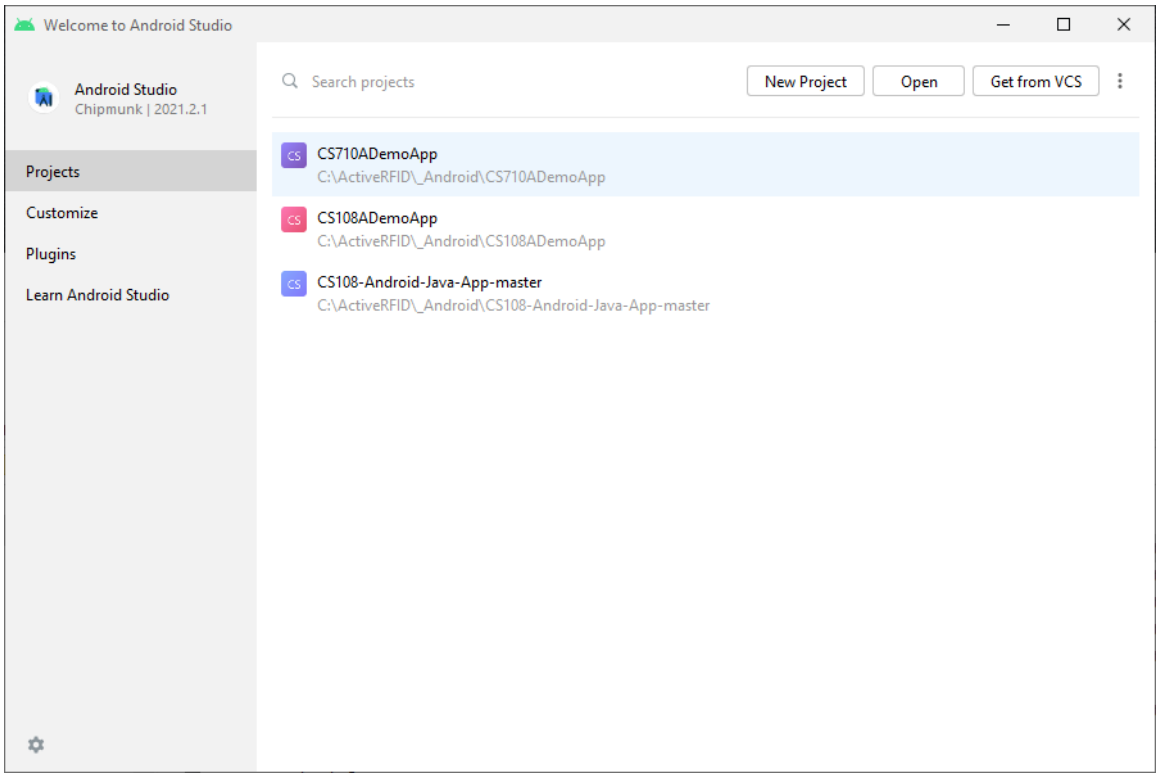
4. Select “Media device (MTP)” for most android phone. Noted that we need to select “Send images(PTP)” in some phone to allow USB debugging.

v. End

2) Normal Debugging: please connect USB cable from PC to smart phone

i. Open “Android Studio”

Copyright Convergence Systems Limited 2022, All Rights Reserved

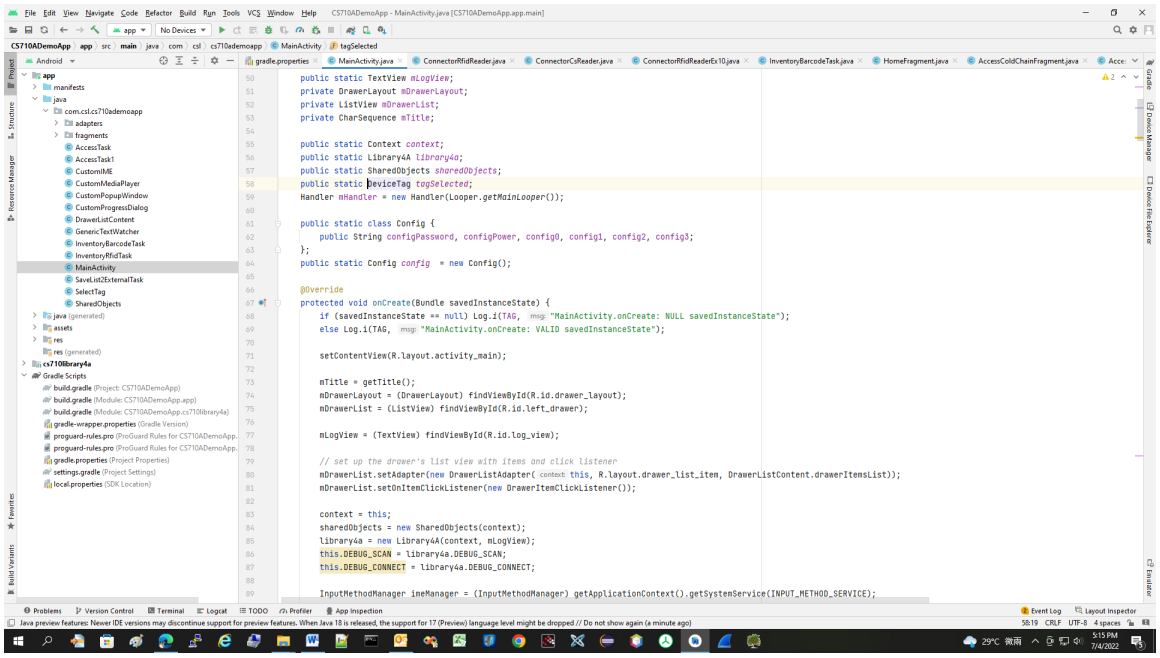


ii. Select the project file on the CS710ADemoApp project on the right. Noted that this step may not be necessary if we do not close the project before previous exit of the android studio.

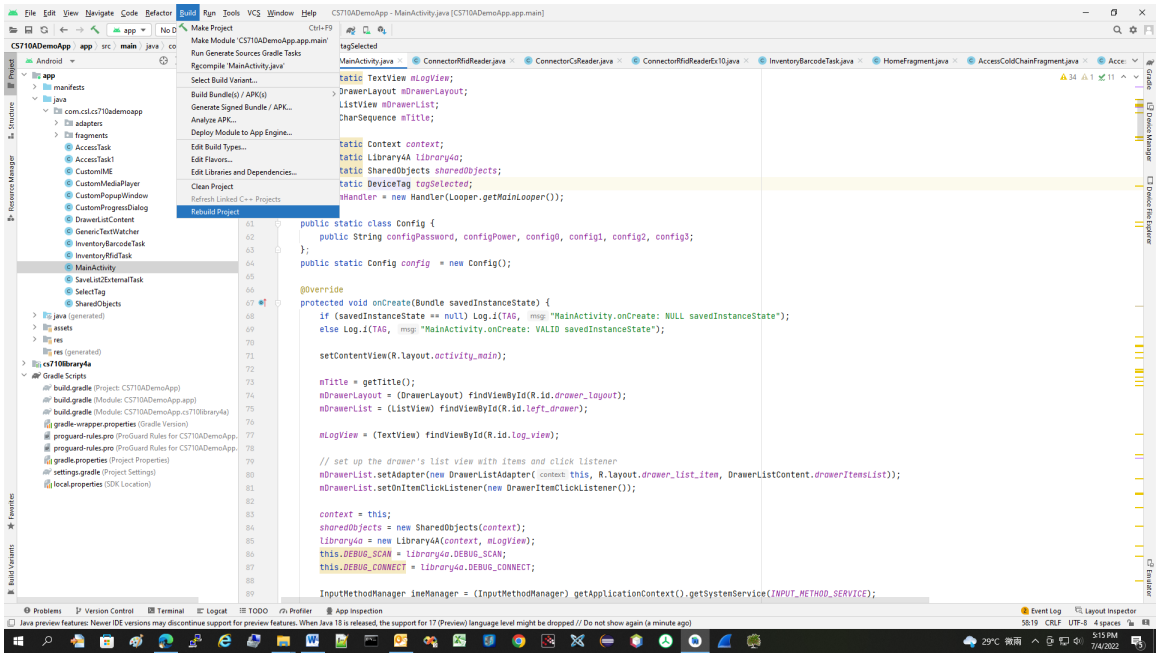
3.4 Modify CS710S Project to be Your Project

1) Modify the project (the files under “app” module) for your own project

Copyright Convergence Systems Limited 2022, All Rights Reserved



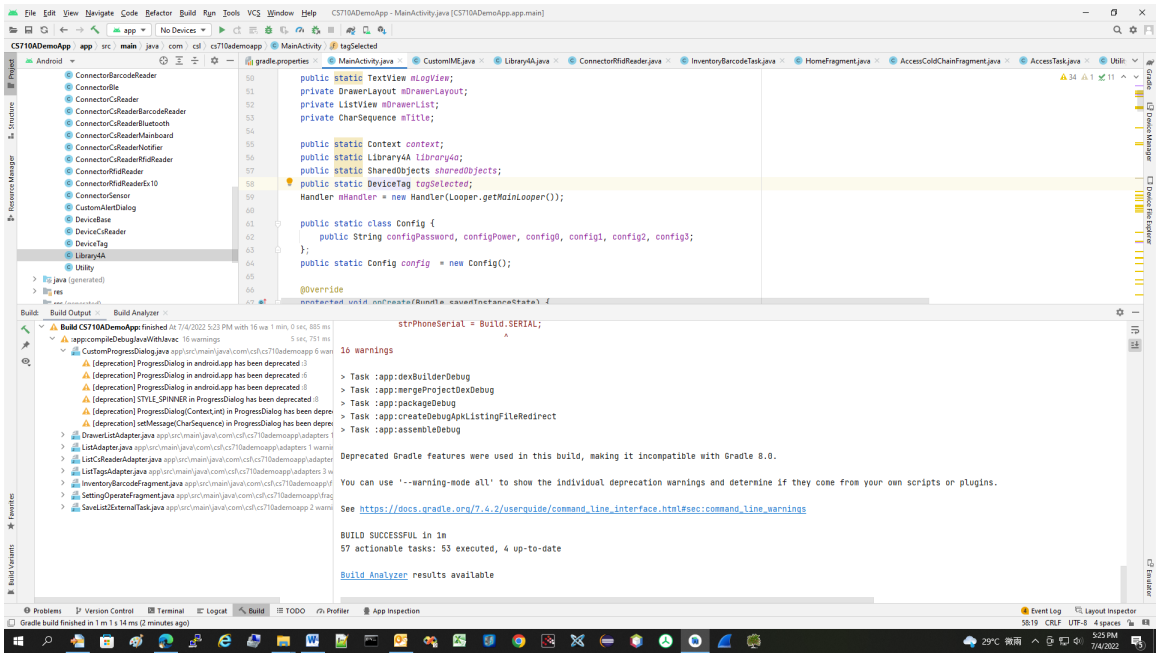
2) Select “Rebuild Project” in the “Build” menu bar of Android Studio. This cleans and compiled the whole project.



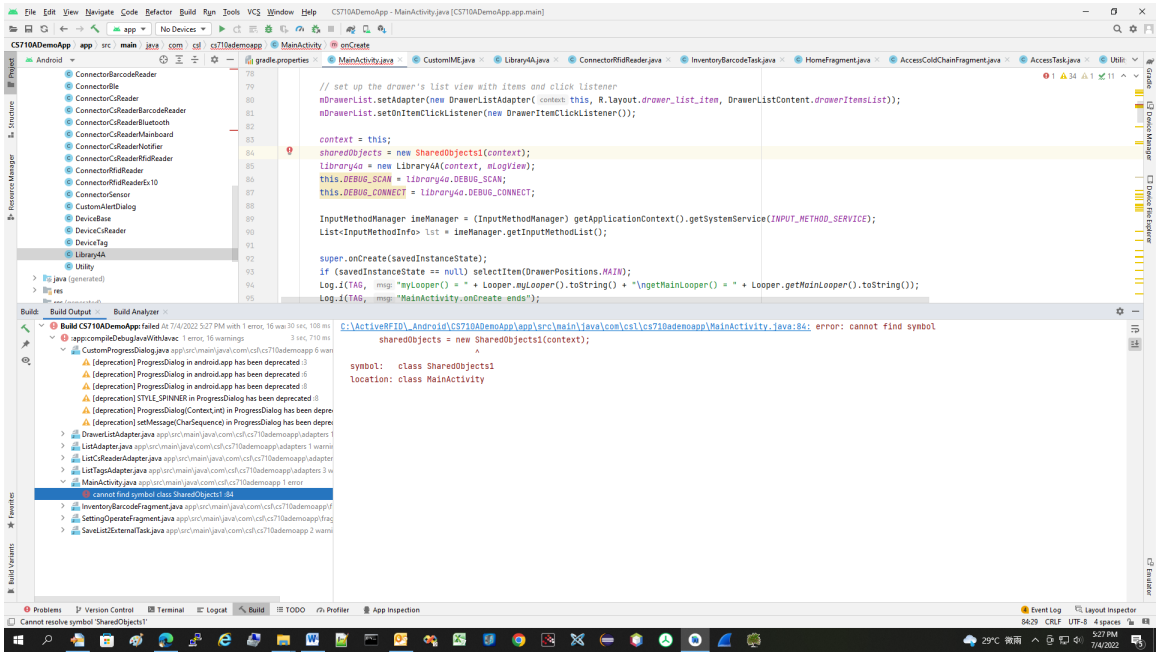
3) Select “Build” tab in the bottom to see if there is error

If no error, it shows as followings.

Copyright Convergence Systems Limited 2022, All Rights Reserved



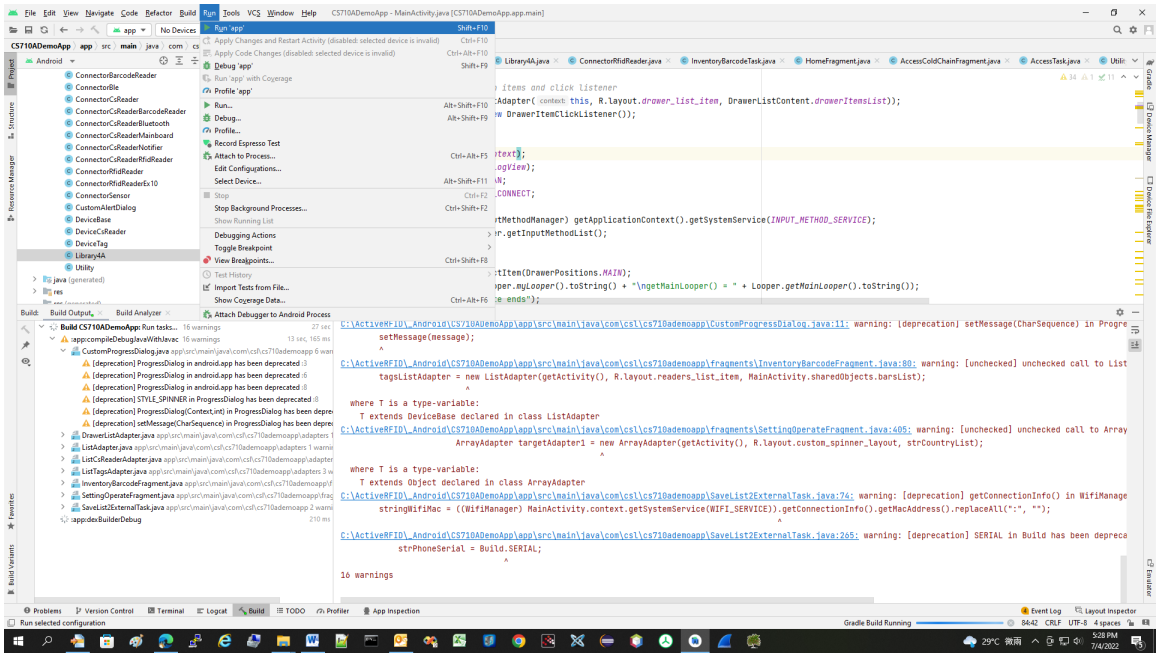
If error, it shows as followings. Select the items with  in the “Build” message window. Android studio should open the appropriate file with error in the source code windows the top right part. It shows the reason of the error in the lower right part of the ‘build’ window. Modify the source code.



4) Plug in the android phone that has set USB debugging.

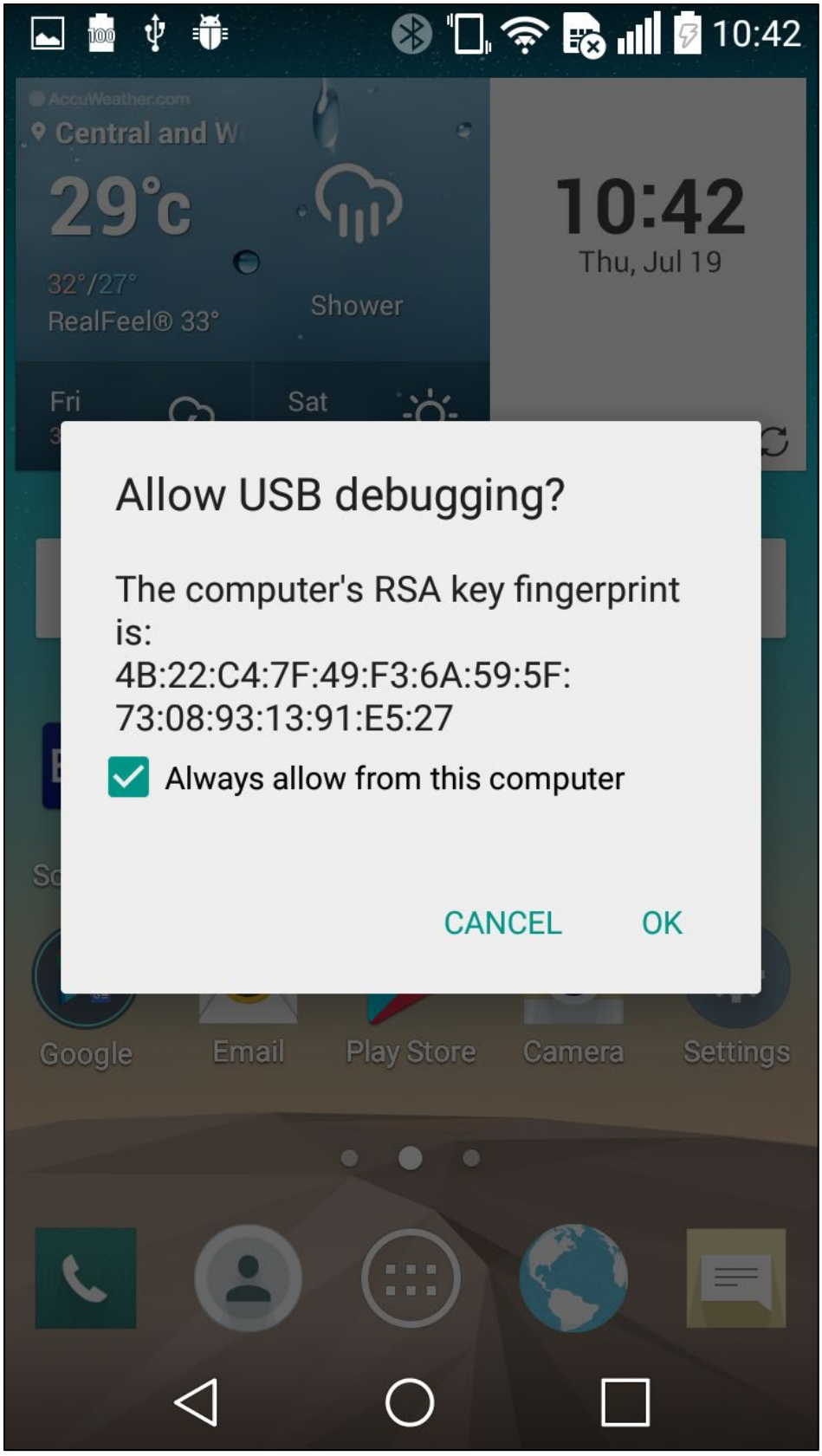
5) Select “Run app” in the “Run” menu to download and to install the test app to android phone.

Copyright Convergence Systems Limited 2022, All Rights Reserved



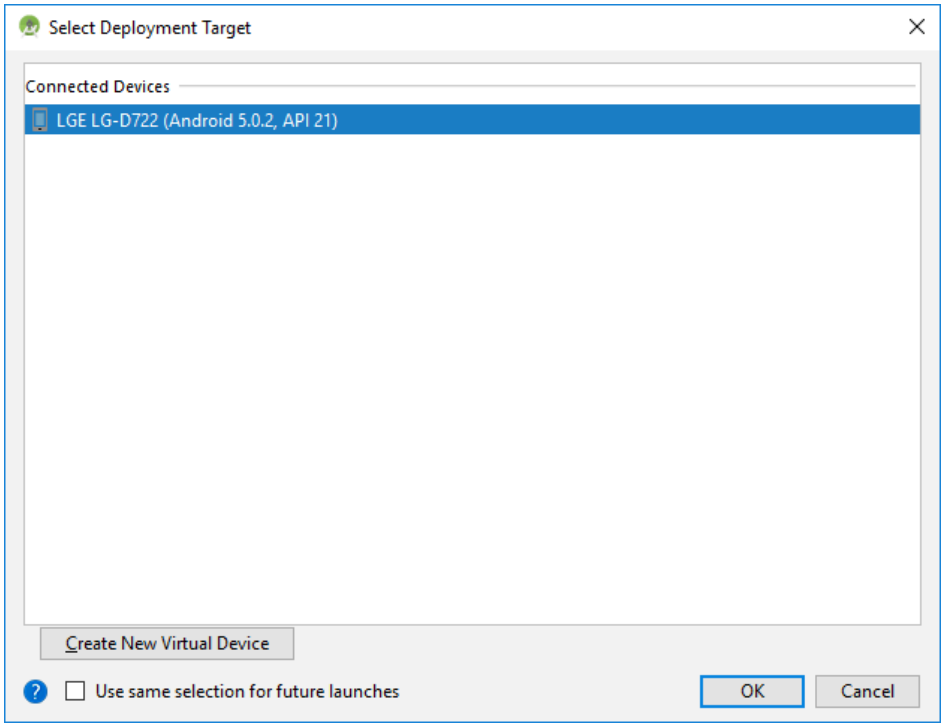
5) For the first time of android debugging, the android phone has following message to request to initialize a data transmission key between android phone and the android studio. Tick the box “Always allow from this computer” and Select OK.

Copyright Convergence Systems Limited 2022, All Rights Reserved

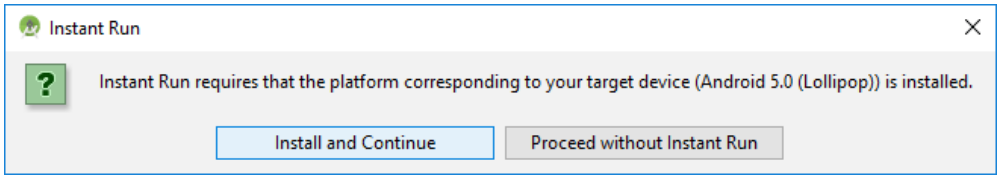


Copyright Convergence Systems Limited 2022, All Rights Reserved

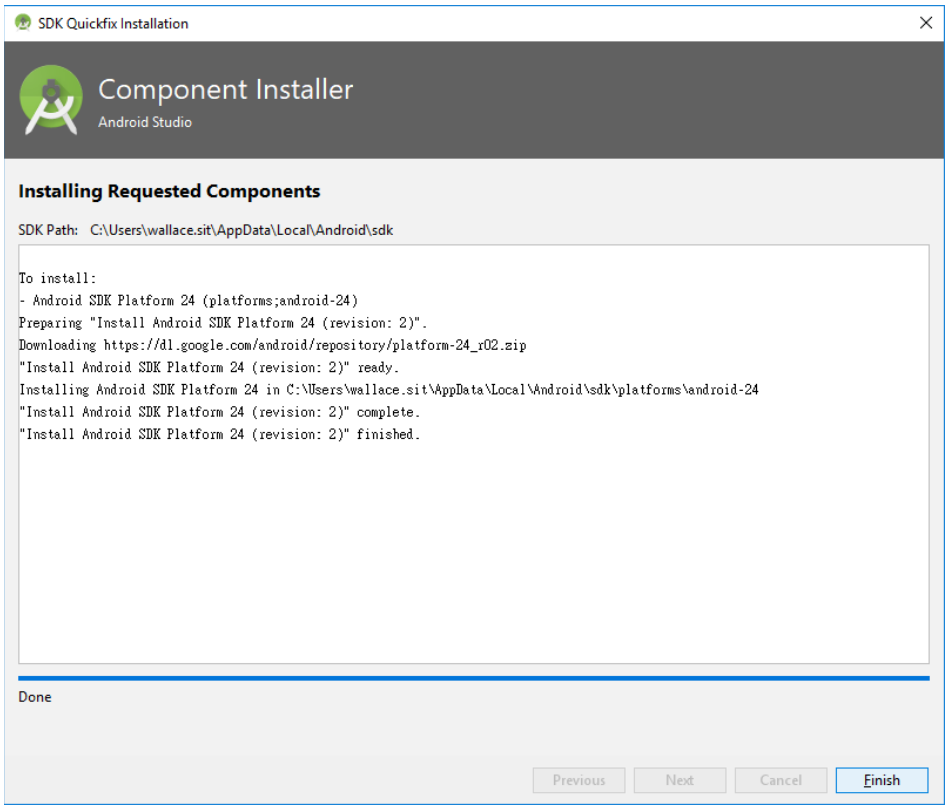
6) In Android Studio, select “OK” in the “Select Deployment Target”



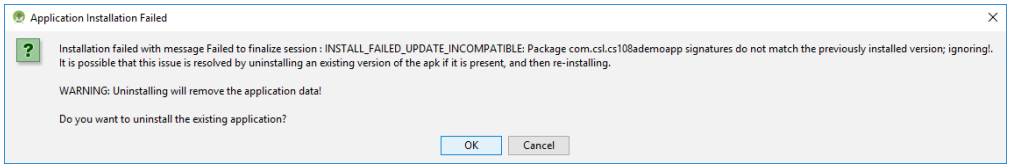
7) For the first time of android debugging, after the Android studio recognizes the target phone device platform, it requests to install appropriate “Run” platform. Select “Install and Continue”. Wait. Then press “Finish”



Copyright Convergence Systems Limited 2022, All Rights Reserved



8) For the some android phone or for the first time of debugging, it may have following message to request to uninstall the previous version of app having the same name. Select “OK” to continue



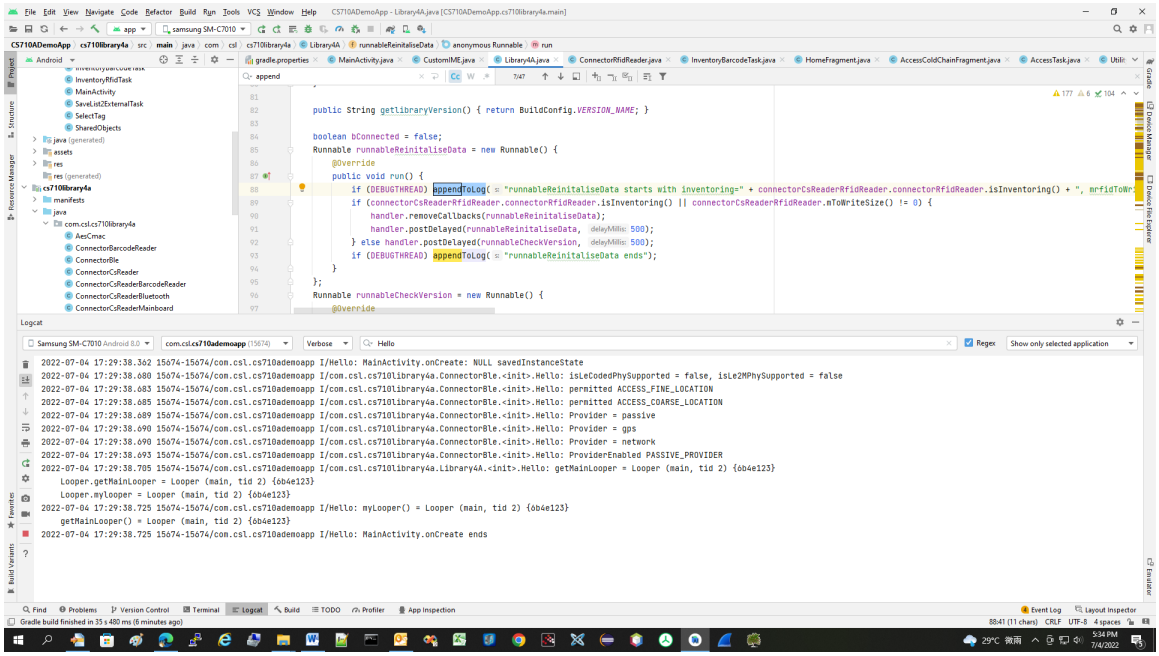
9) The previous apps having same name should be uninstalled. The new apps should be installed and executed with the following display in the android phone.

Copyright Convergence Systems Limited 2022, All Rights Reserved

Copyright Convergence Systems Limited 2022, All Rights Reserved

10) For the logical software error, we need some debug message to debug the problem. We may place the debug message by using the android command Log.i("Hello", "test debug message") or the library4a command appendToLog("test debug message").

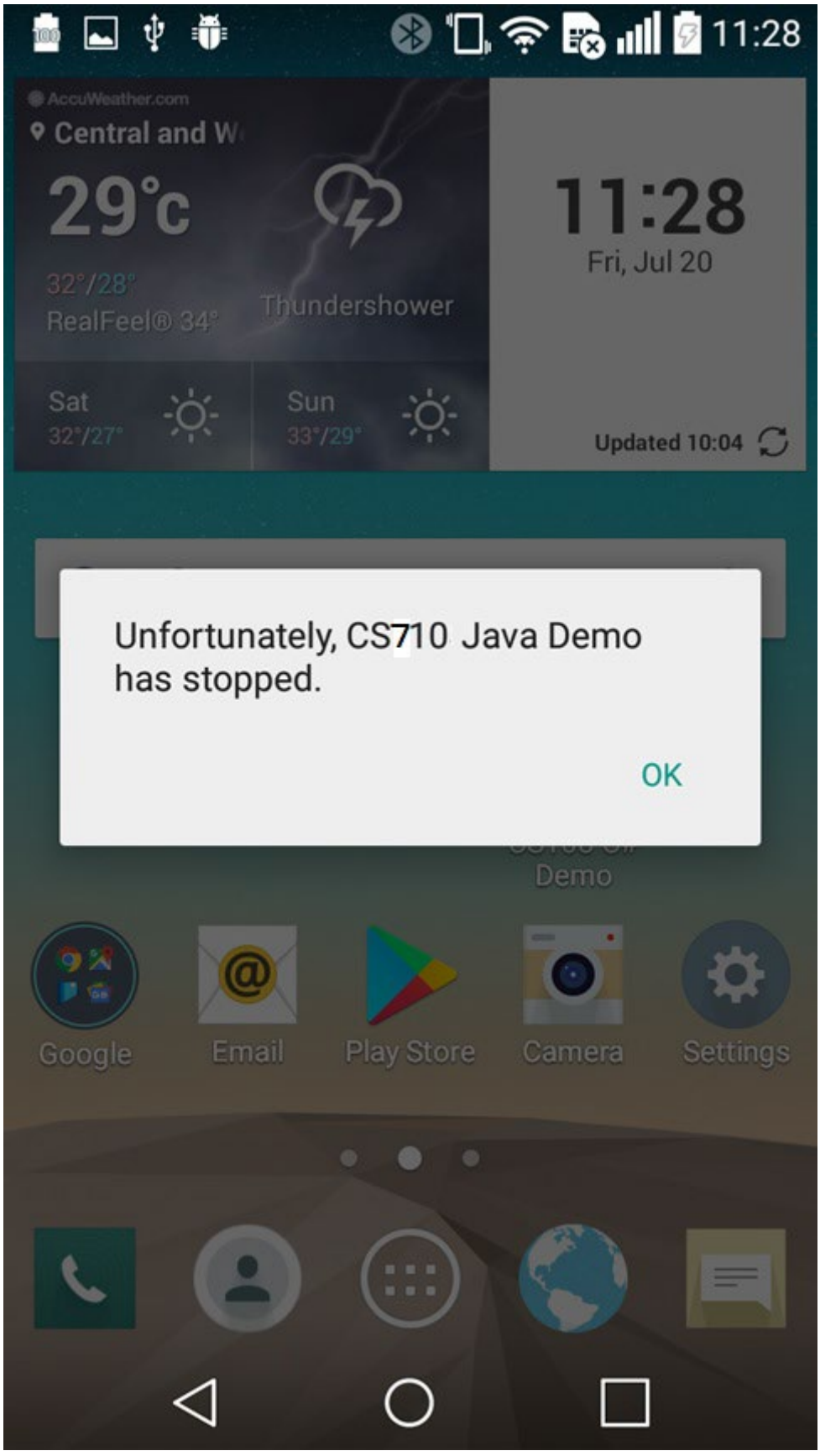
    Select "LogCat" tab in the bottom of Android Studio to display the debug message. Select "Verbose" with appropriate filter word, such as "Hello" which is the default TAG of the Log command of the library4a.



11) For software clash and reset problem, we may use Android studio to debug. Connect the phone with usb. Select "LogCat" tab in the bottom of Android Studio to display the debug message. Select "Error" to see possible error messages especially when the app clashes. No filter is needed. Select Run in android studio menu to download and execute the app. Try repeat the clash.

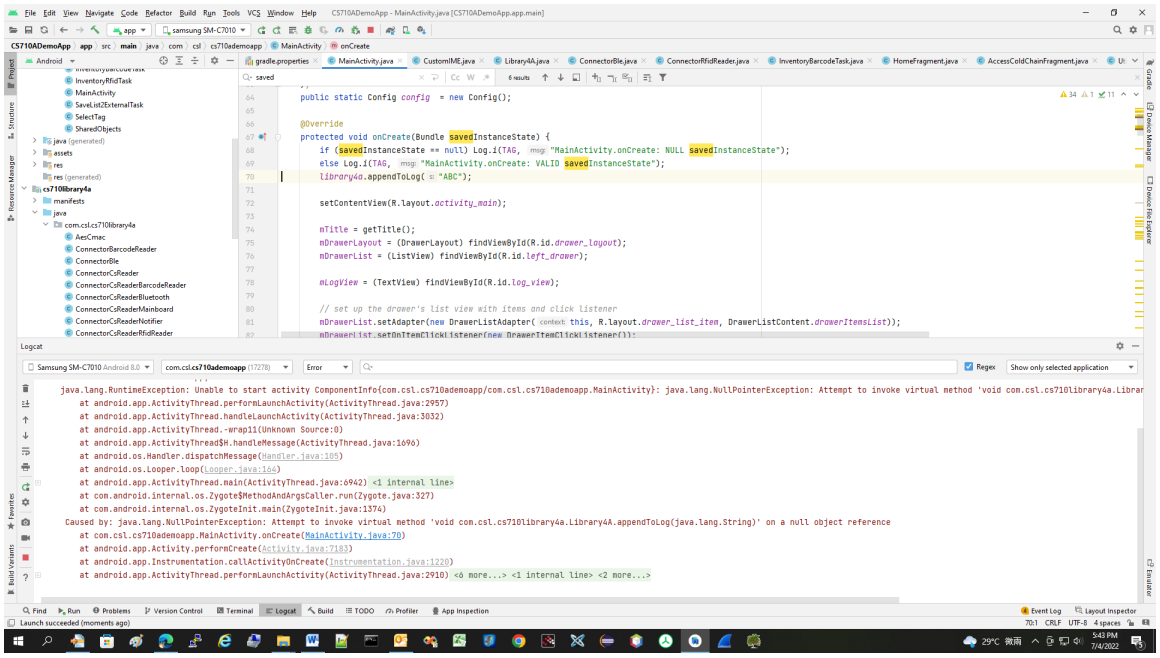
    If the application crashes, the phone app has display like the following.

Copyright Convergence Systems Limited 2022, All Rights Reserved

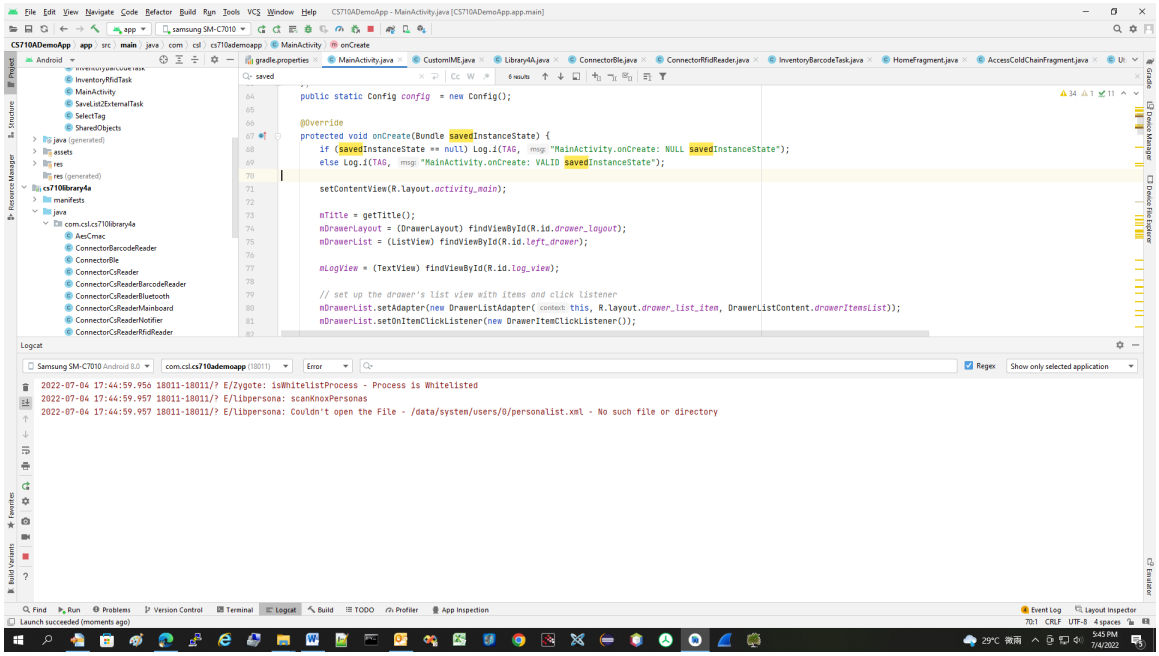


Copyright Convergence Systems Limited 2022, All Rights Reserved

If error, android studio displays like following. Tick the Blue line in the LogCat window to go to the problem line of the problem file in the upper right part of the android studio.

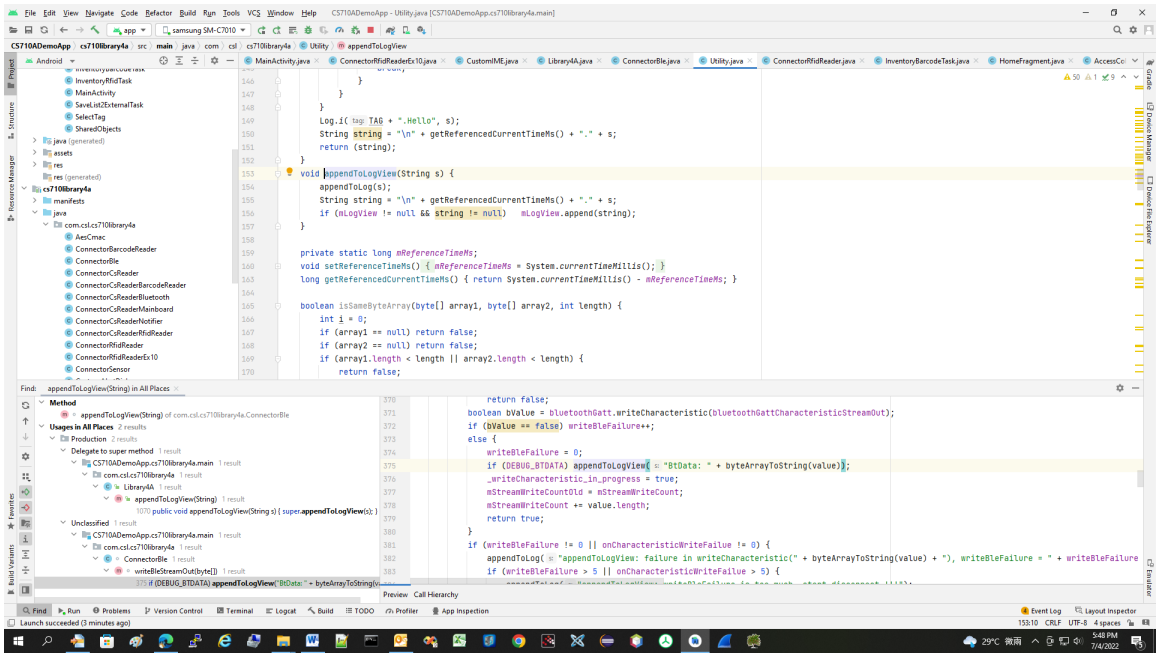


If no error, android studio displays like following.



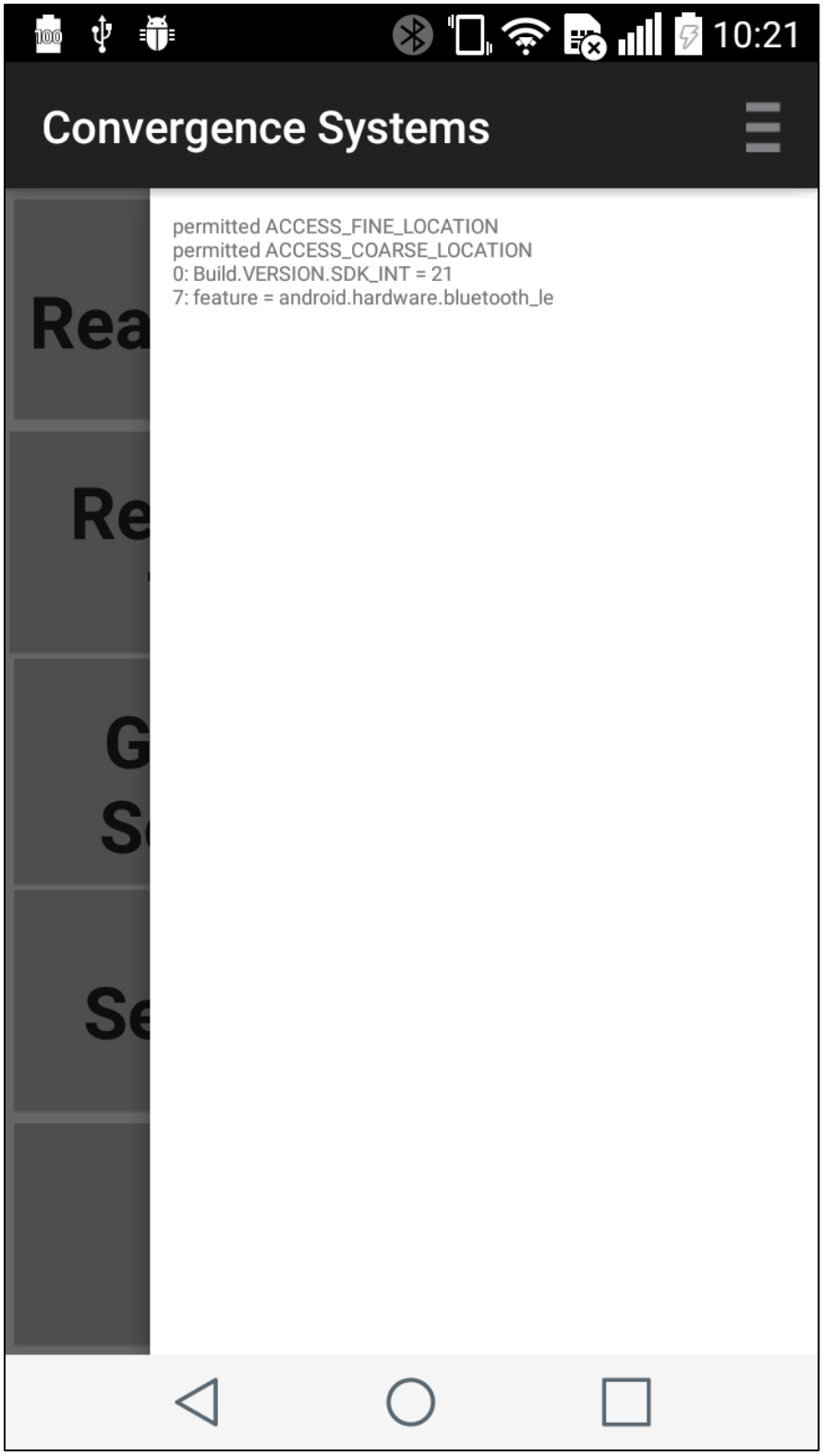
12) Debug message without Android Studio. Sometimes problem cannot be repeated when the phone connects to android studio. The DemoApp adds ListView class mLogView in the right DrawerLayout. We can add the debug message by mLogView.append("Test debug message"); The mLogView is also used by library4a class.

Copyright Convergence Systems Limited 2022, All Rights Reserved



The debug message is seen by swiping the right edge of the apps.

Copyright Convergence Systems Limited 2022, All Rights Reserved



13) Modify and debug the test project. Repeat above.

14) End

Copyright Convergence Systems Limited 2022, All Rights Reserved

3) Demo App Architecture

The demo app has three modules: app, cs710library4a and Gradle

A. ‘app’: the modules contains the UI related classes and methods. It is grouped as manifests, java, assets and resources

i. ‘manifests’: it has AndroidManifest.xml which defines the applications and activities, user-permission used in the ‘app’ module.

ii. ‘java’: it contains the java UI related source files. It is grouped as adapters, fragments and shared.

1. ‘adapters’: it contains the different adapters for ViewPager, or ListView Android UI components used in some UI pages.

2. ‘fragments’: it contains the definition of different UI pages

3. shared: it is located directory under ‘java’ directory. it includes ‘AccessTask’, ‘CustomIME’, CustomMediaPlayer,… They are common UI related tasks called by several UI pages.

4. MainActivity.java: it is the starting point of the Demo app.

iii. ‘assets’: it contains the audio files used by the UI pages to output sound

iv. ‘res’: resource group has the UI data used by the UI pages. It is grouped as drawable, layout, menu, values and xml.

1. Drawable: it contains the pictures for the UI pages

2. Layout: it contains the layout definition of different UI components in different UI pages.

3. Menu: it contains the layout definition of different menu components used in different UI pages

4. Values: it contains colors, dimens, strings and styles used in different UI pages.

5. Xml: it is not used now.

B. ‘cs710library4a’: the library modules contains non-UI related classes and methods. It is also group under manifests, java and resources. ‘manifests’ has AndroidManifest.xml which defines the classes, user-permission used in the module. ‘java’ contains the related source codes. There are 18 files. ‘res’ has the data used by the library modules. There are only useful file – device_filter.xml, popup.xml and strings.xml.

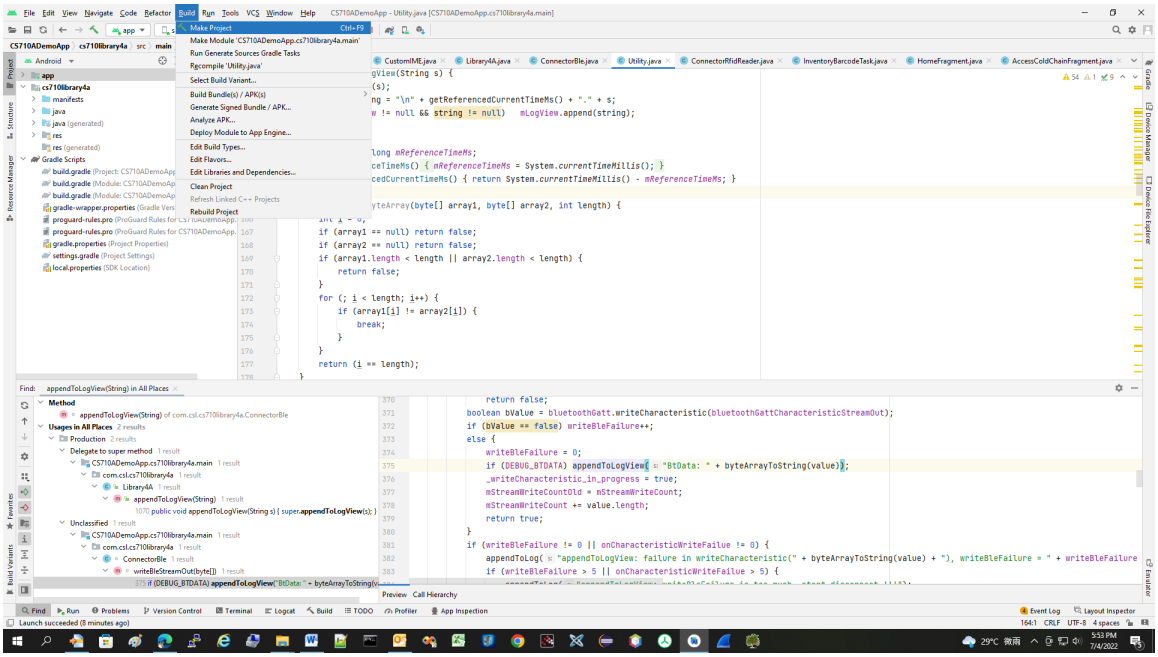
C. ‘Gradle Scripts’: it contains some global properties with .gradle files. Those properties includes required sdk versions, version name,…..

4) Generate apk for direct download and installation

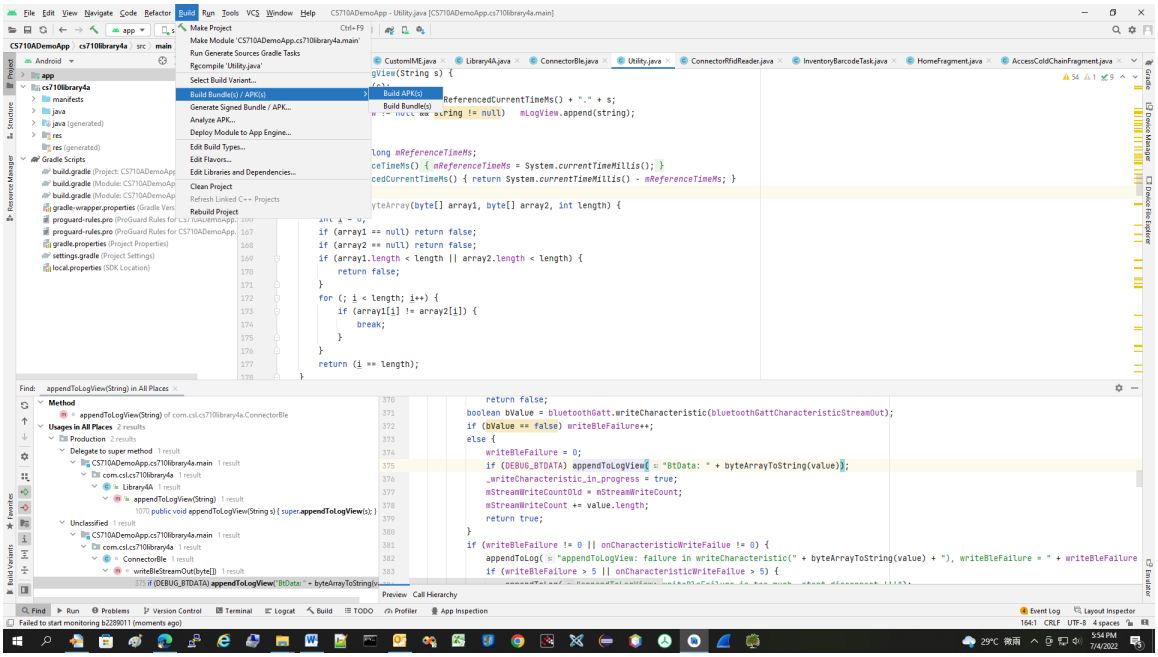
Copyright Convergence Systems Limited 2022, All Rights Reserved

A. Source file preparation – Change the versionName value within build.gradle for the Module app so that new app version can be displayed for new generated apps.

B. Select “build” -> ‘Make Project’ in Android Studio



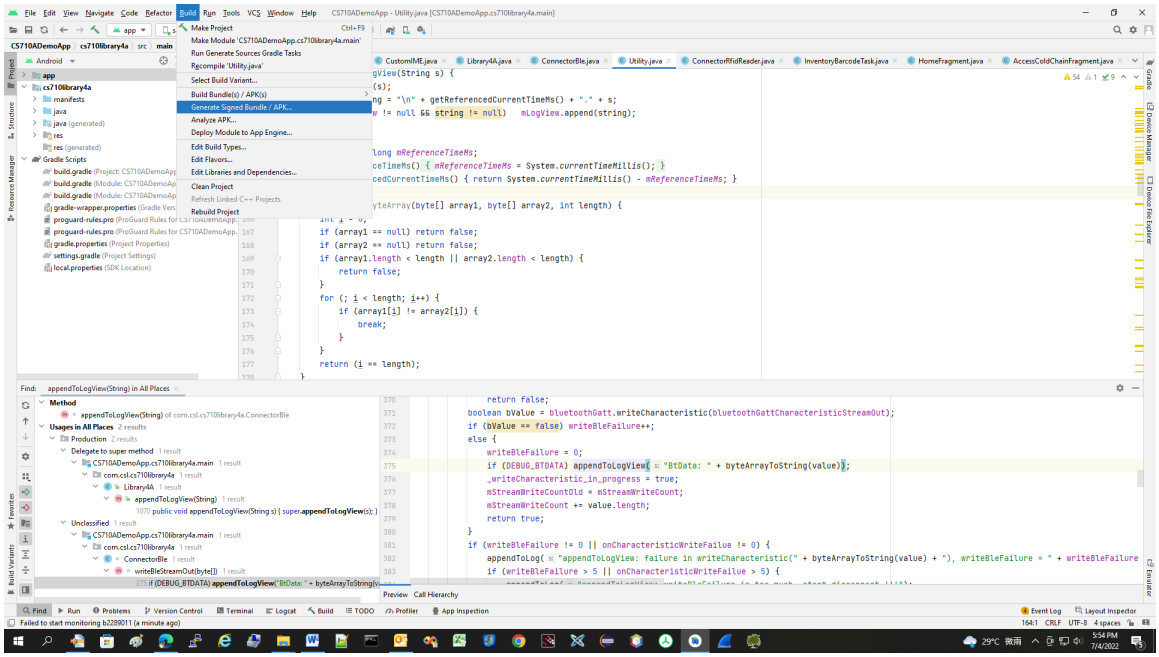
Or select “build” -> “build bundle/apk” -> “build apk”



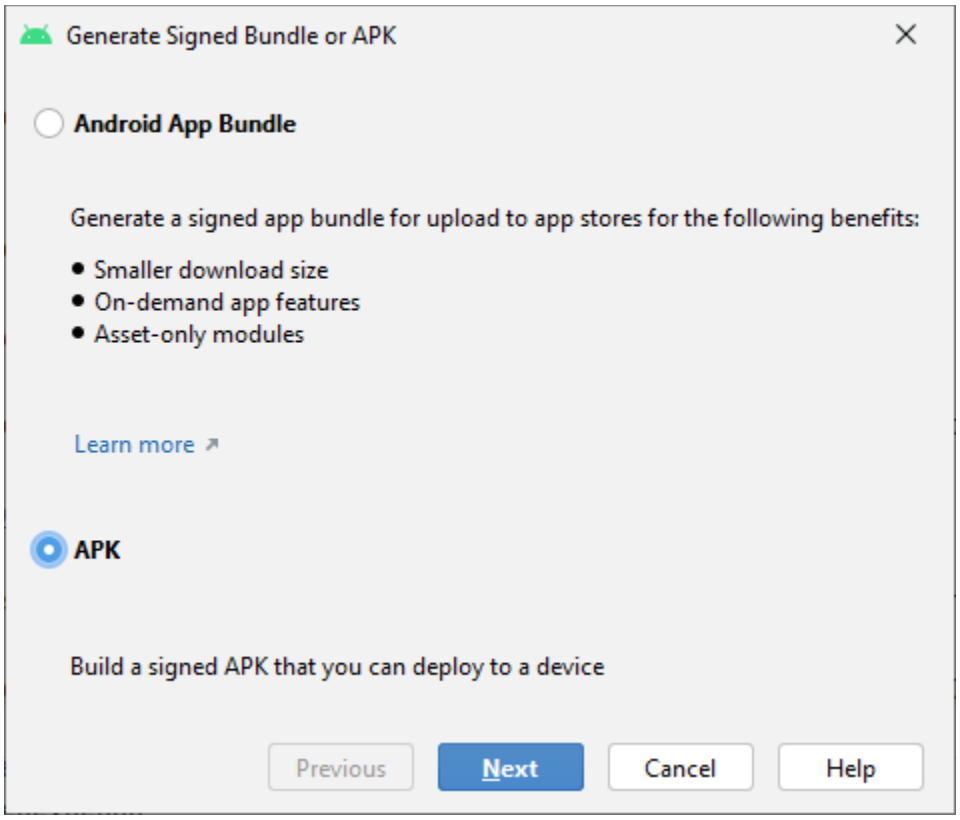
If successful, the file app-debug.apk is in app\build\outputs\apk\debug

Copyright Convergence Systems Limited 2022, All Rights Reserved

Or Select “build” -> “generate signed bundle/apk”

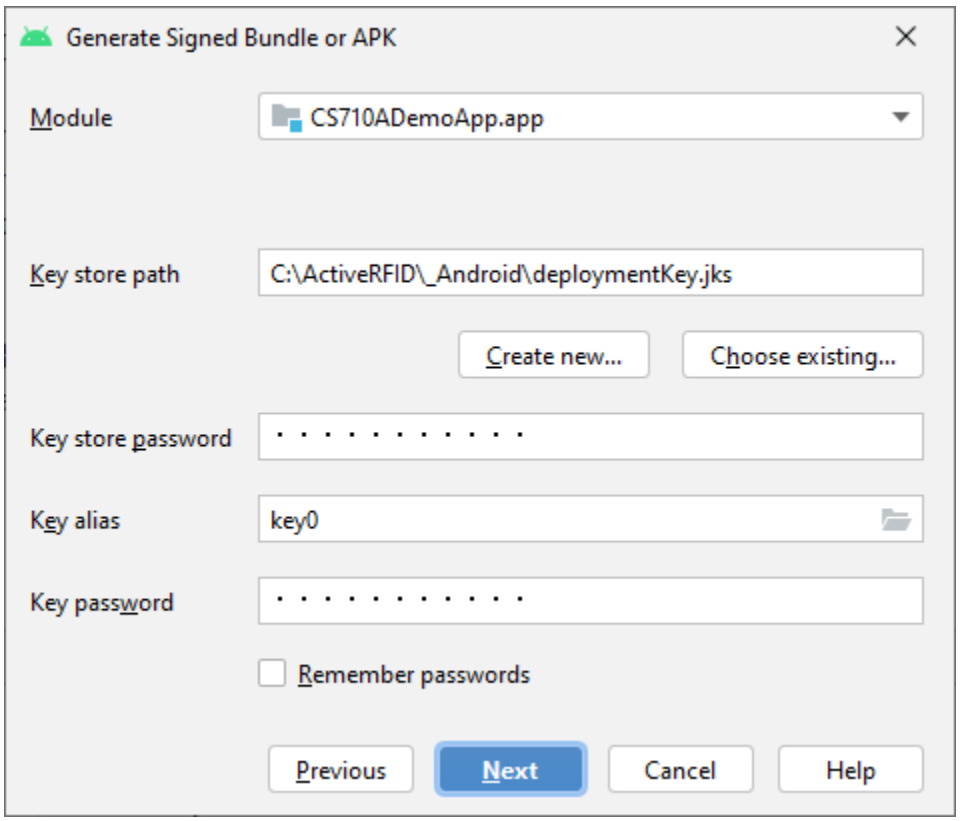


Then Select “apk” and then “next”

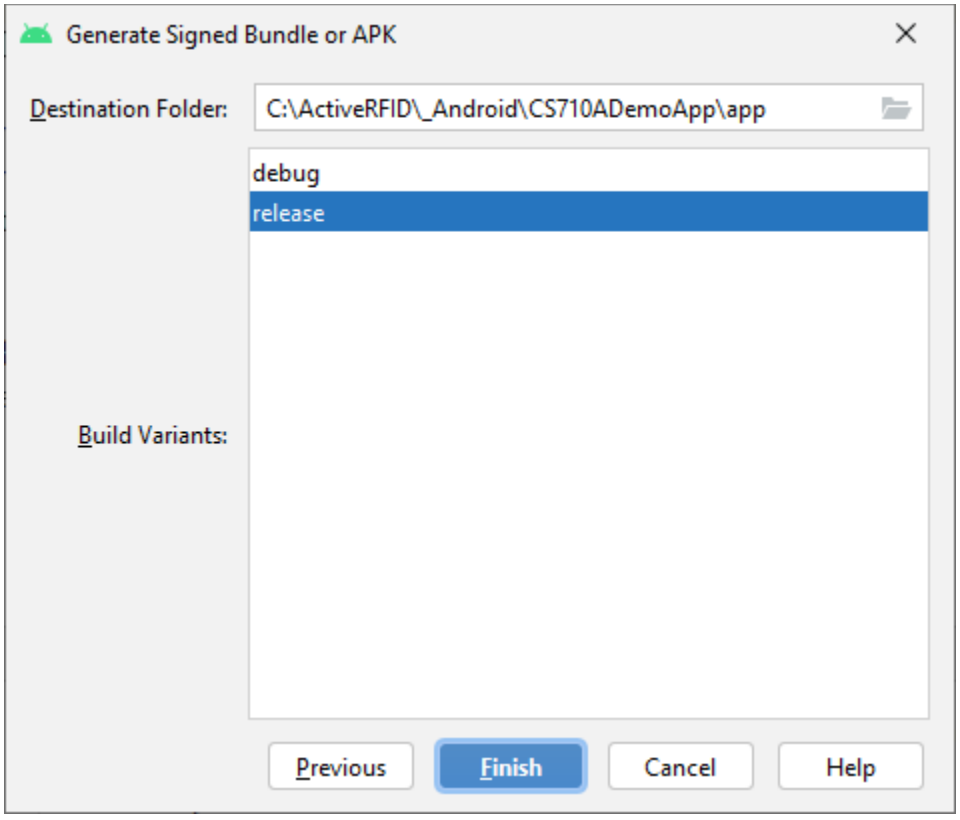


Then input the key location, passwords and alias, then “next”

Copyright Convergence Systems Limited 2022, All Rights Reserved



Then select “debug” or “release” and “Finish”



Copyright Convergence Systems Limited 2022, All Rights Reserved

If successful and debug file is selected, the signed file app-debug.apk is located in app\debug. If successful and release file is selected, the signed file app-release.apk is located in app\release.

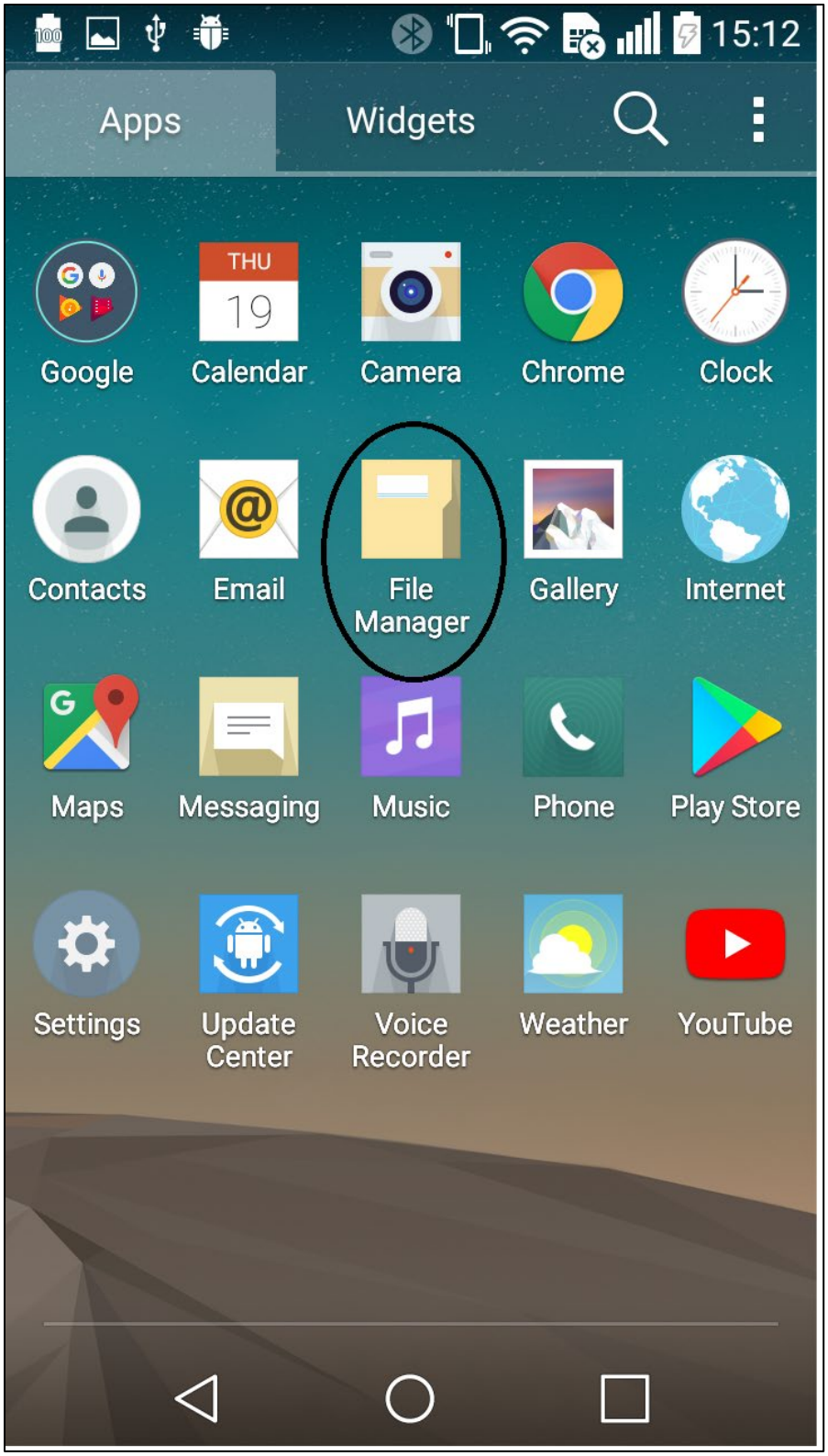
C. File inventory: Rename the app-debug.apk to any name such as app-debug-180718.apk with the .apk extension. You may zip the file. You may put the file in the website, such as the CS710-Android-Demo-App-1.19.0_Installer.zip file in https://www.convergence.com.hk/downloads/cs710S/ for android phone to download. You may send the file to others for installation.

D. File reception: android phone can get the apk file from website using Android phone browser, or from computer using USB connection with computer. The file is usually placed under /Download sub-directory.

E. File location in android phone:

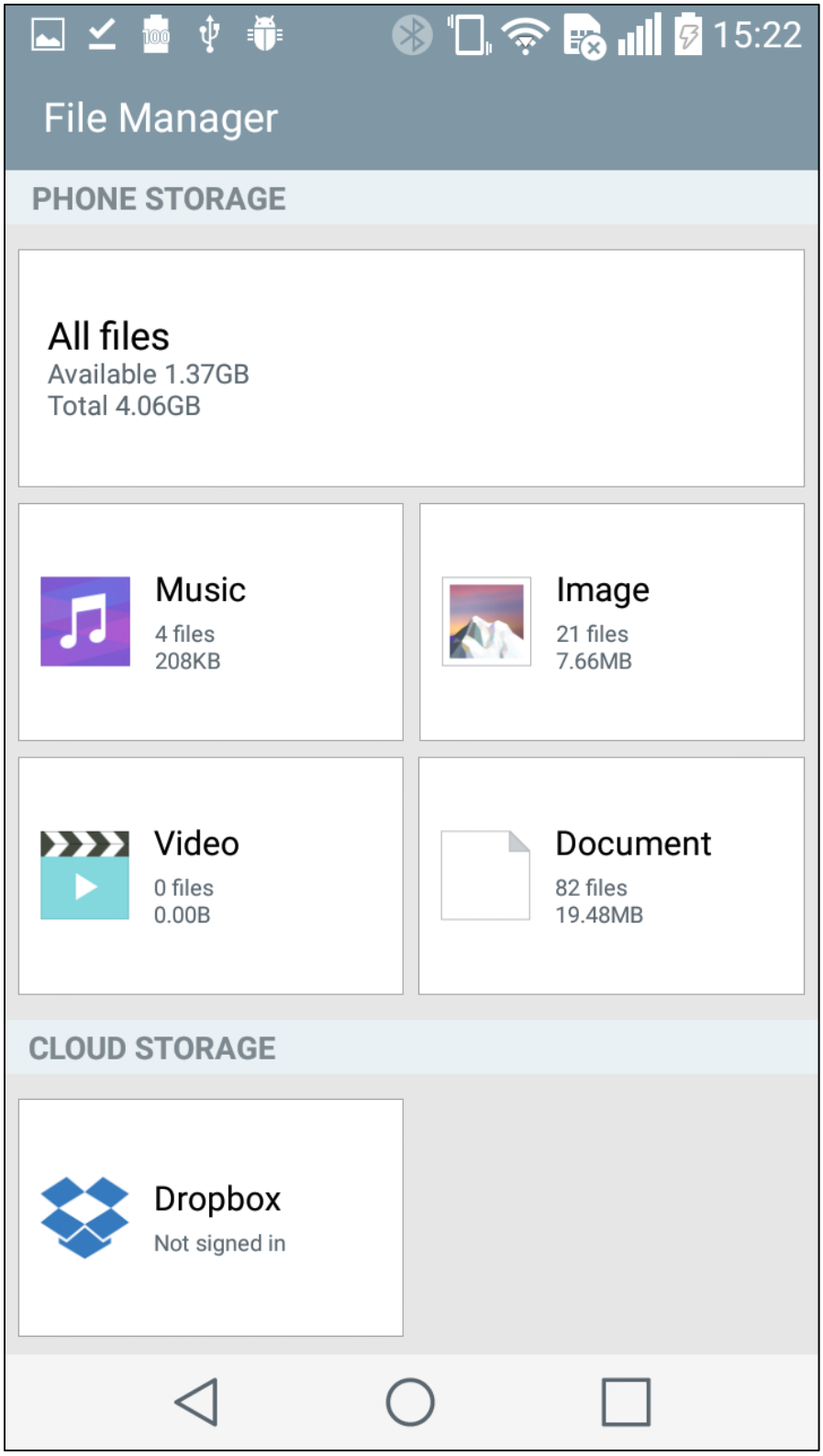
   i. use some 'File Manager' app to locate the received apk file

Copyright Convergence Systems Limited 2022, All Rights Reserved



ii. Select ‘Document’

Copyright Convergence Systems Limited 2022, All Rights Reserved



iii. Browse through different sub-directories until the appropriate apk file is found in the list. Noted that the usual directory is in /storage/emulated/0/Download

Copyright Convergence Systems Limited 2022, All Rights Reserved

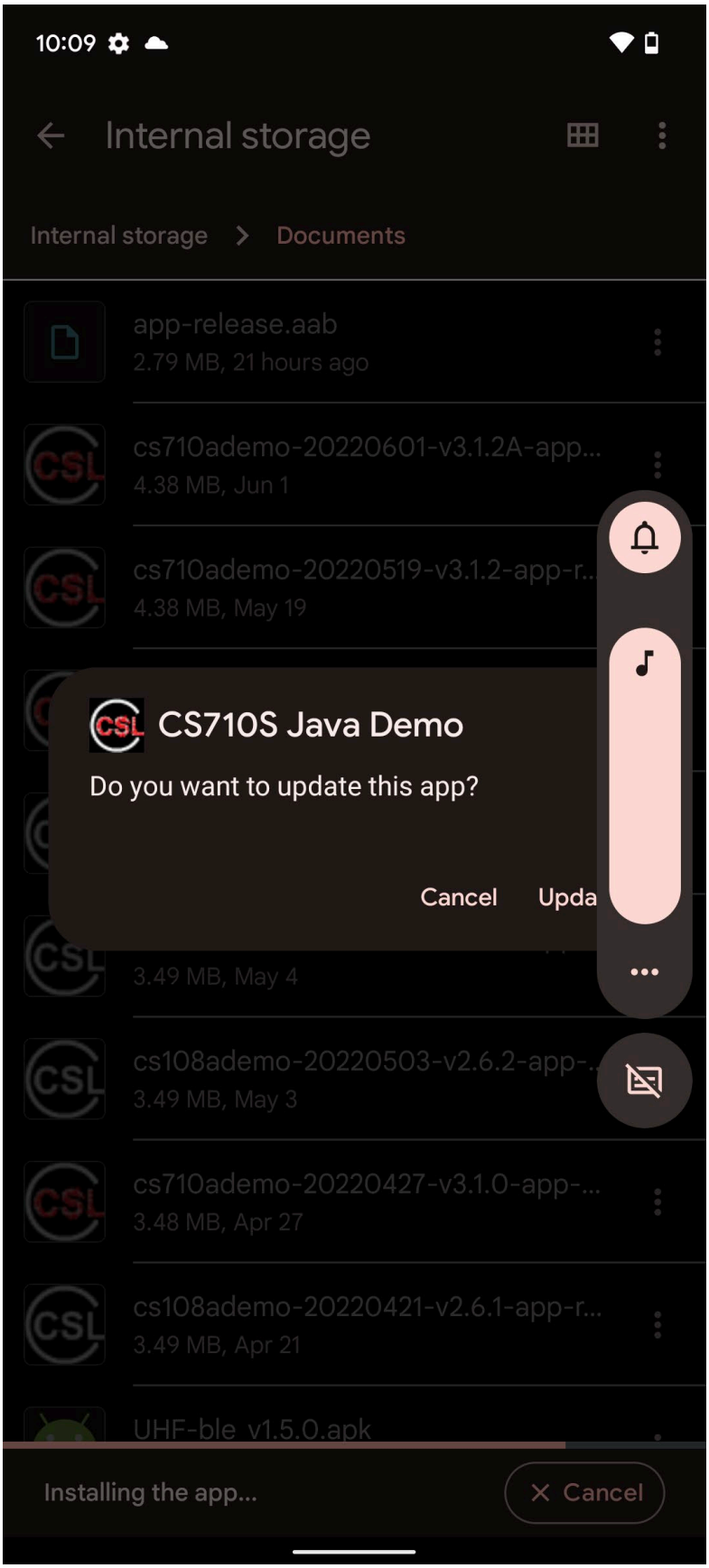
iv. end

F. apk file installation in android phone

i. Select the appropriate apk file found in ‘FileManager’ app

ii. Select ‘Install’ or ‘Update’

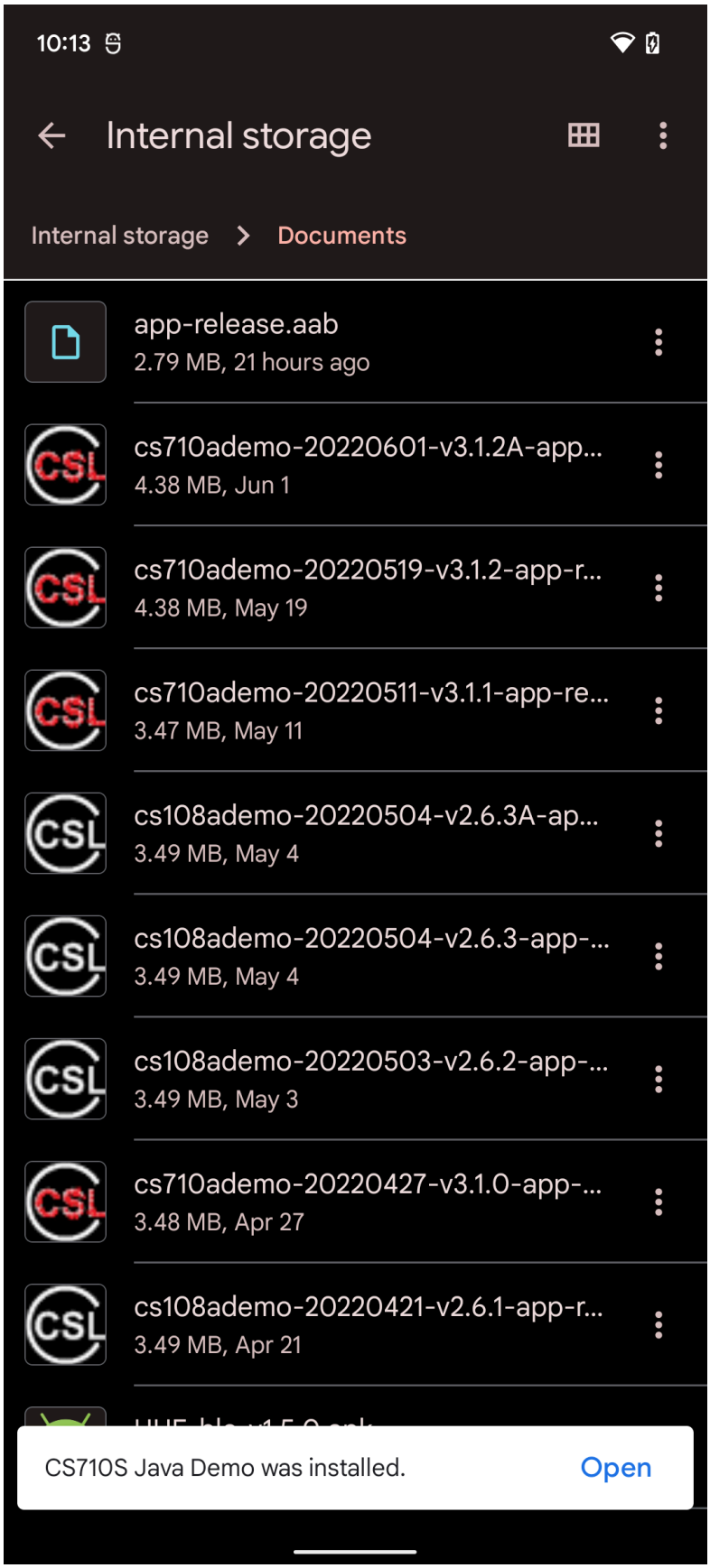
Copyright Convergence Systems Limited 2022, All Rights Reserved



Copyright Convergence Systems Limited 2022, All Rights Reserved

iii. Select Open

Copyright Convergence Systems Limited 2022, All Rights Reserved



Copyright Convergence Systems Limited 2022, All Rights Reserved

iv. Final display

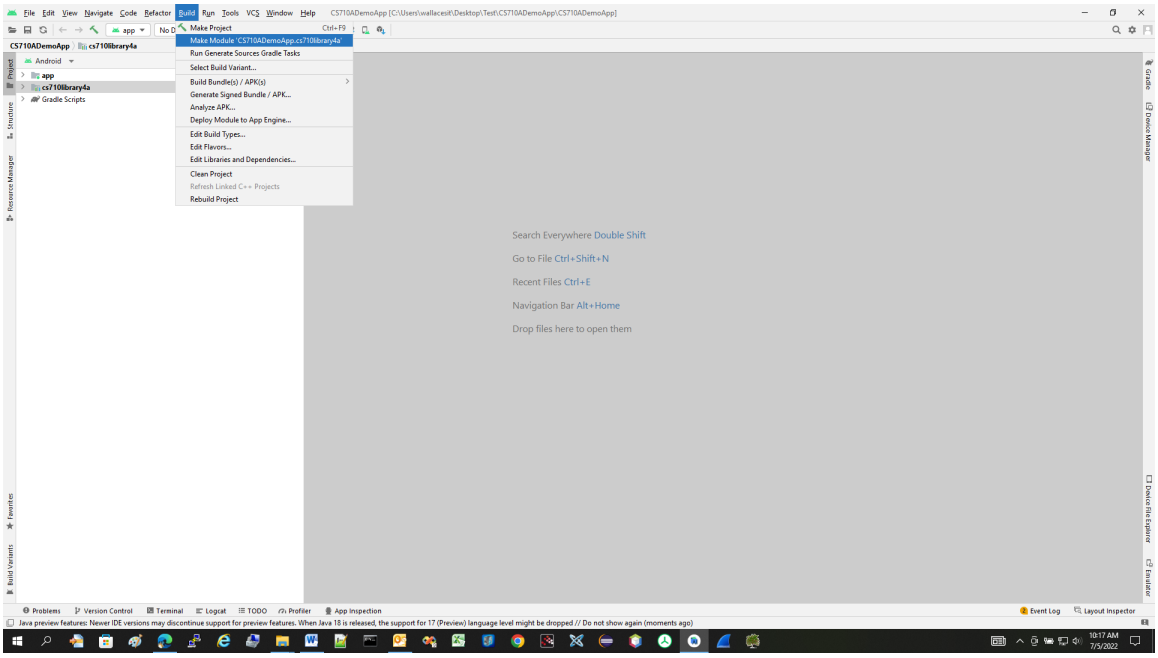
G. End

Copyright Convergence Systems Limited 2022, All Rights Reserved

3.5 Import CS710S Library to Your Own Project

1) create library module

a. Select the “cs710library4a” module in “Project”->”Andriod” in the left hand side. Select “build”-> “make module ‘…cs710library4a’.

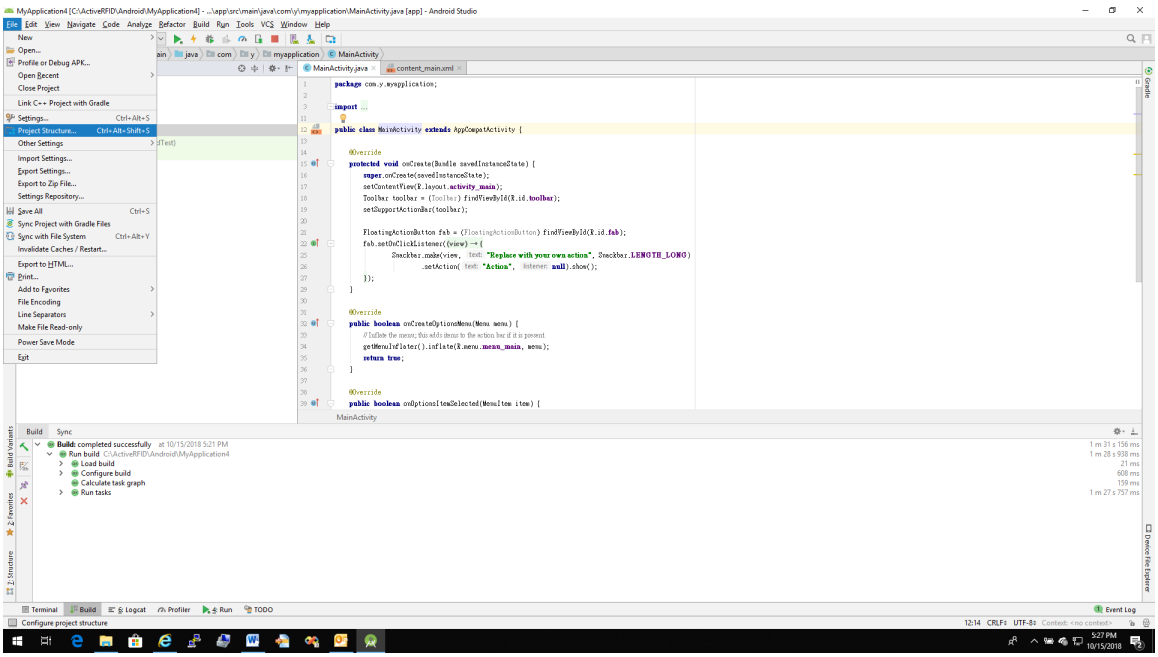


There will have a cs710library4a/build/outputs/aar/cs710library4a-debug.aar

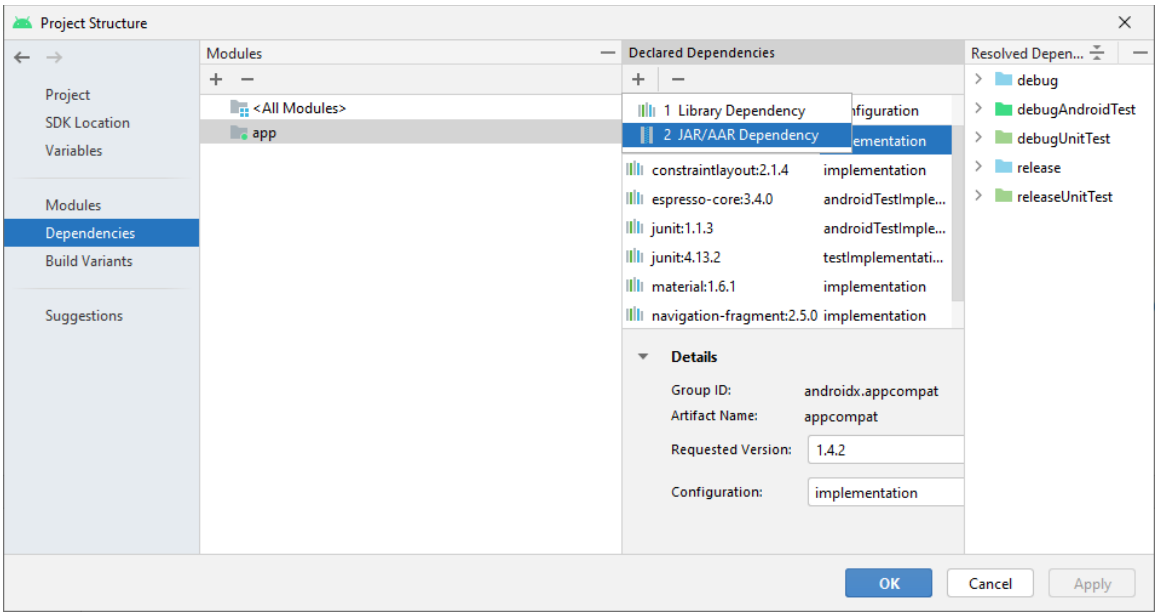
b. Copy the cs710library4a-debug.aar from the CS710ADemoApp\app\libs to your own project, for example, call it MyApplication as in the screen captures below, put it under MyApplication\app\libs

c. Select File -> Project Structure

Copyright Convergence Systems Limited 2022, All Rights Reserved

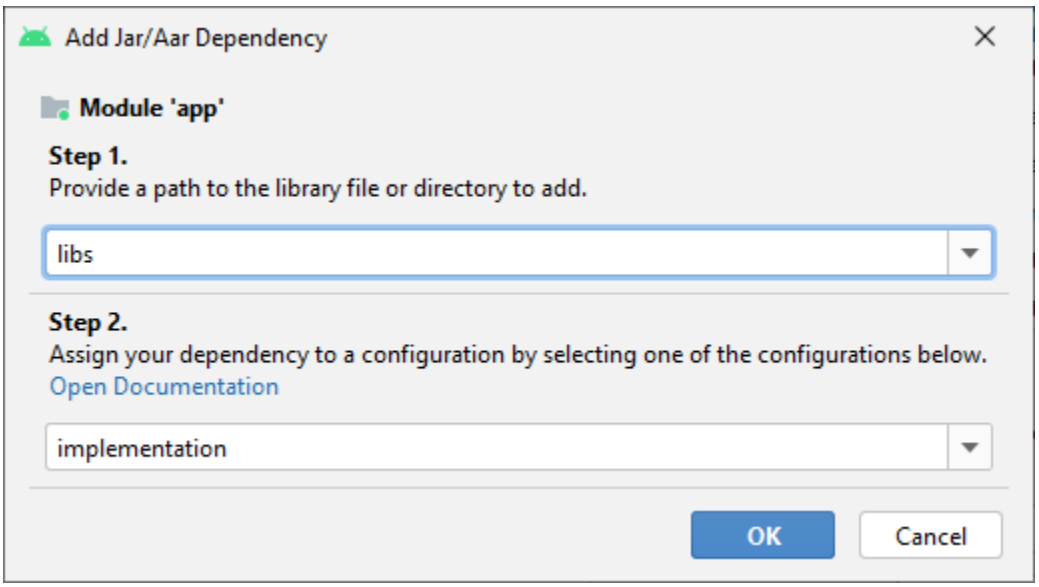


d. Within ‘project structure’, Select ‘Dependencies’, + in Declared Dependencies, ‘JAR/AAR dependency’

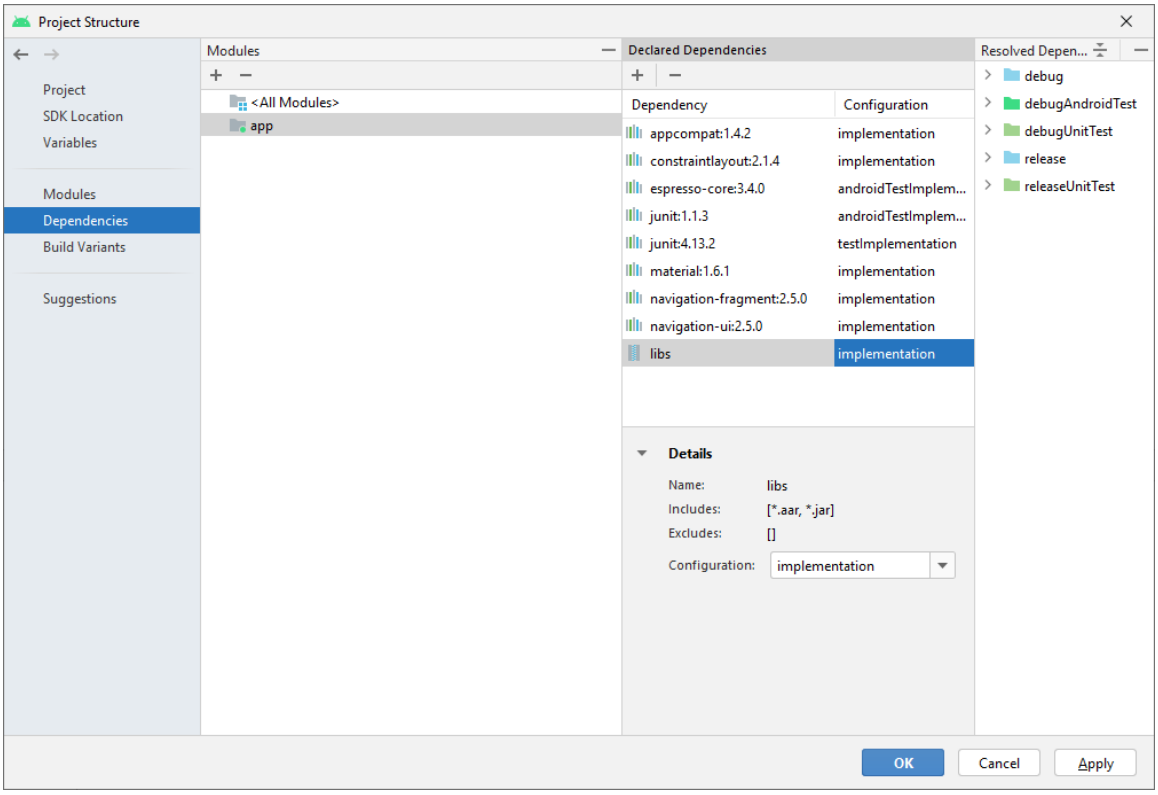


e. Fill in the path where we put the aar file. Select “OK”

Copyright Convergence Systems Limited 2022, All Rights Reserved



f. There is new dependency for app

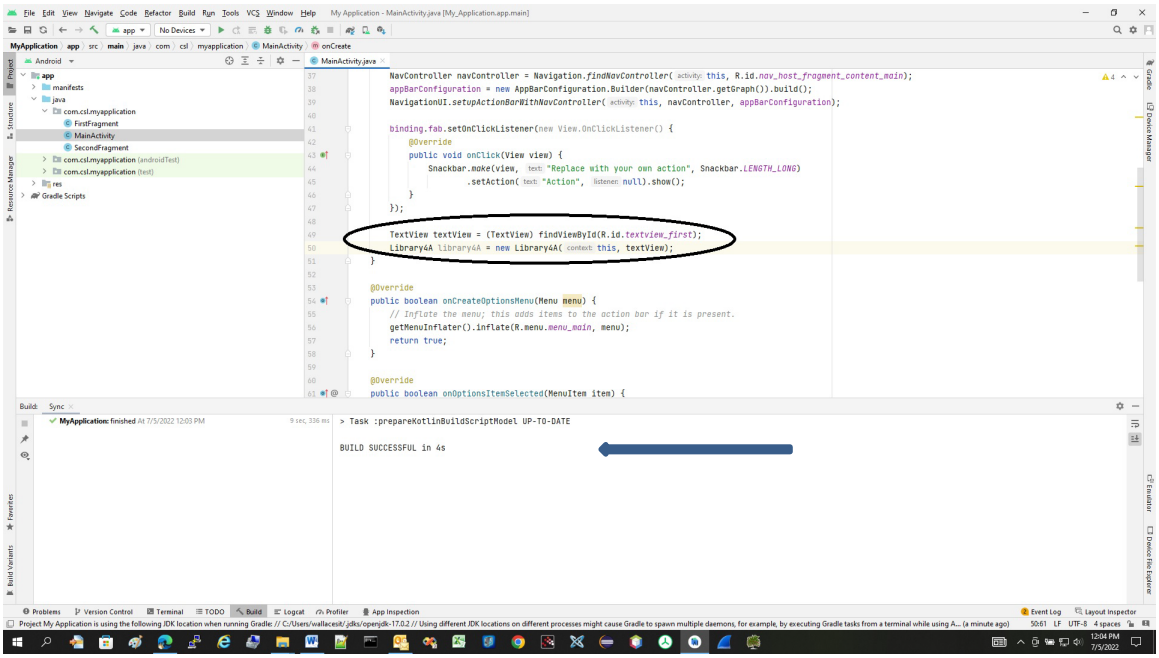


2) Use library module in your project apps

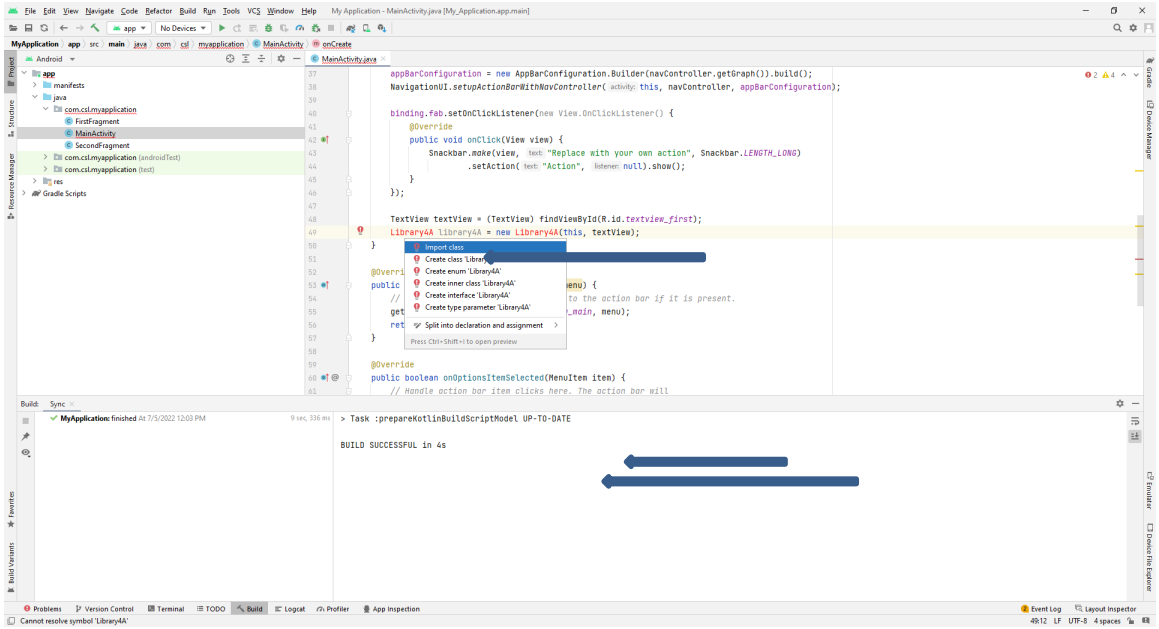
1. Define a TextView class mLogView for library debug

   Define the library class in your project apps (Library4A library4A = new Library4A(this, mLogView))

Copyright Convergence Systems Limited 2022, All Rights Reserved

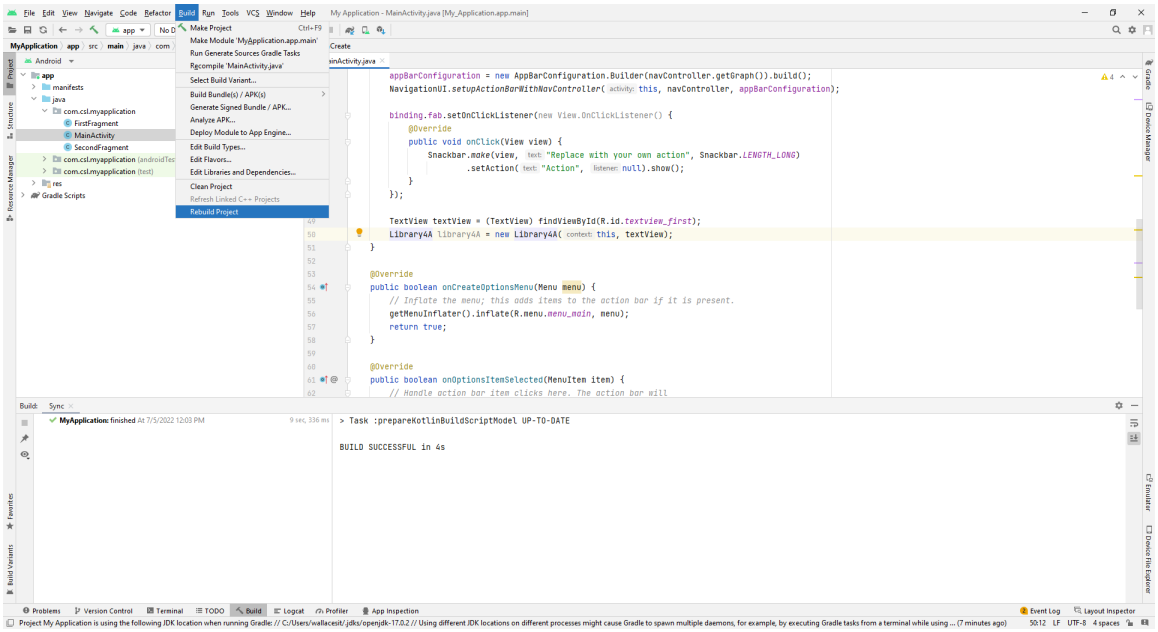


1. The android studio will add “import com.csl.cs710library4a.Library4A” automatically. If now, there is Red word Library4A. Put cursor on the word. Press ALT + Enter. Select ‘Import Class’. You will see Red word changed to black word. You will see new line ‘import com.csl.cs710library4a.Library4A’



2. Rebuild. If no error, it is OK.

Copyright Convergence Systems Limited 2022, All Rights Reserved

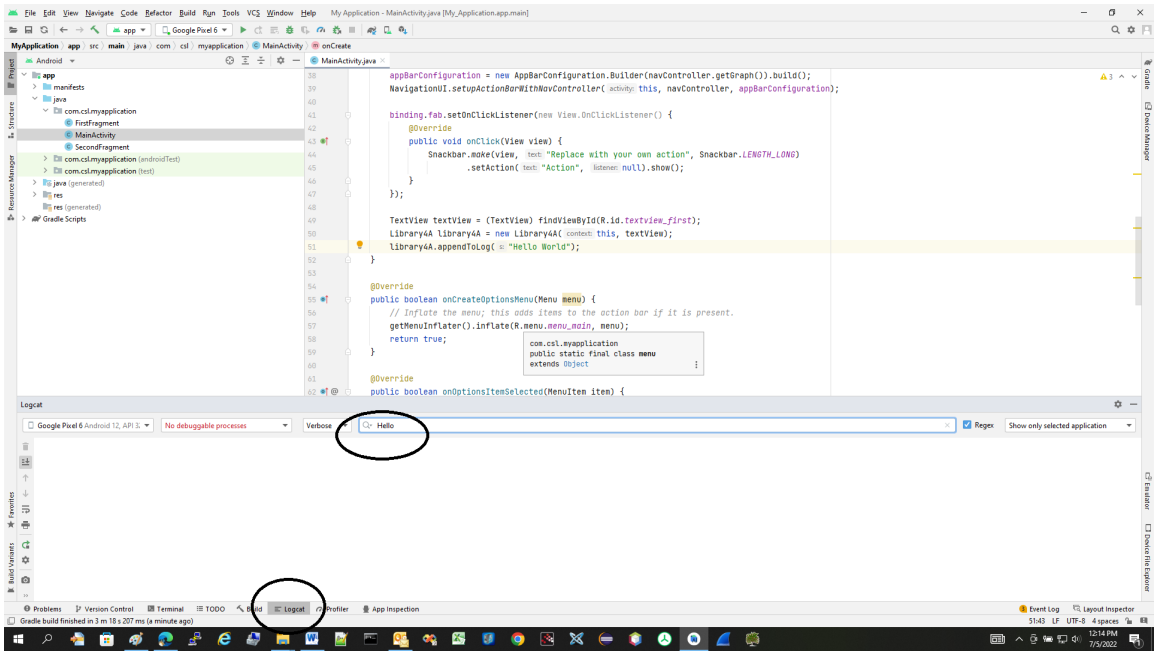


3) Test library module in your project apps

1. Connect your device

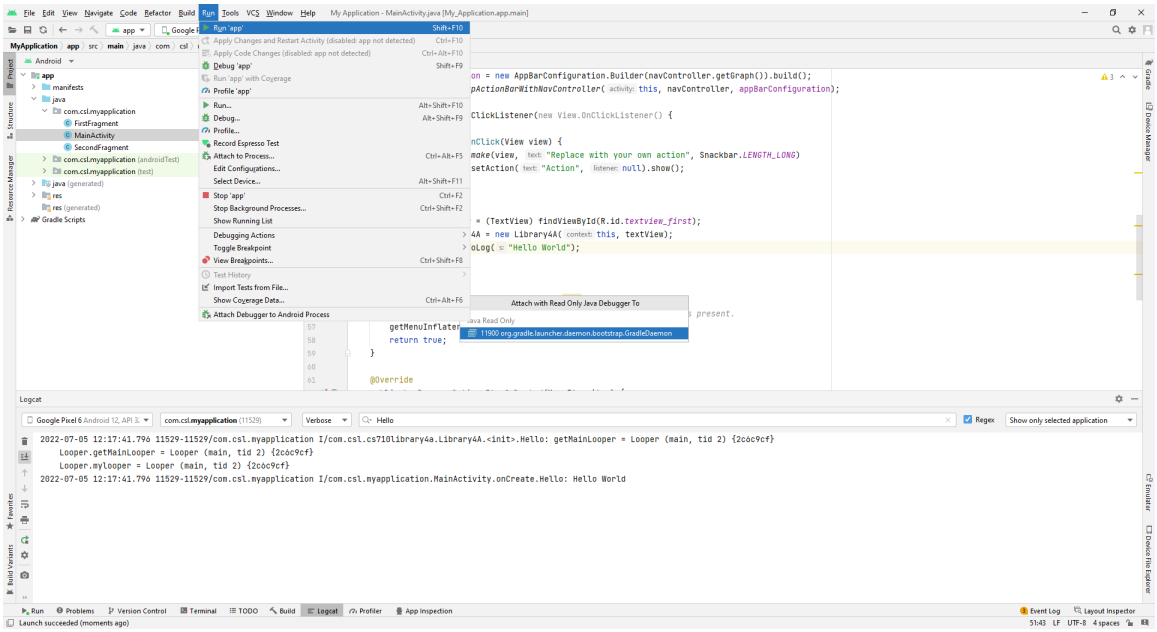
2. Select ‘Logcat’

   Set ‘Hello’ in the filter area

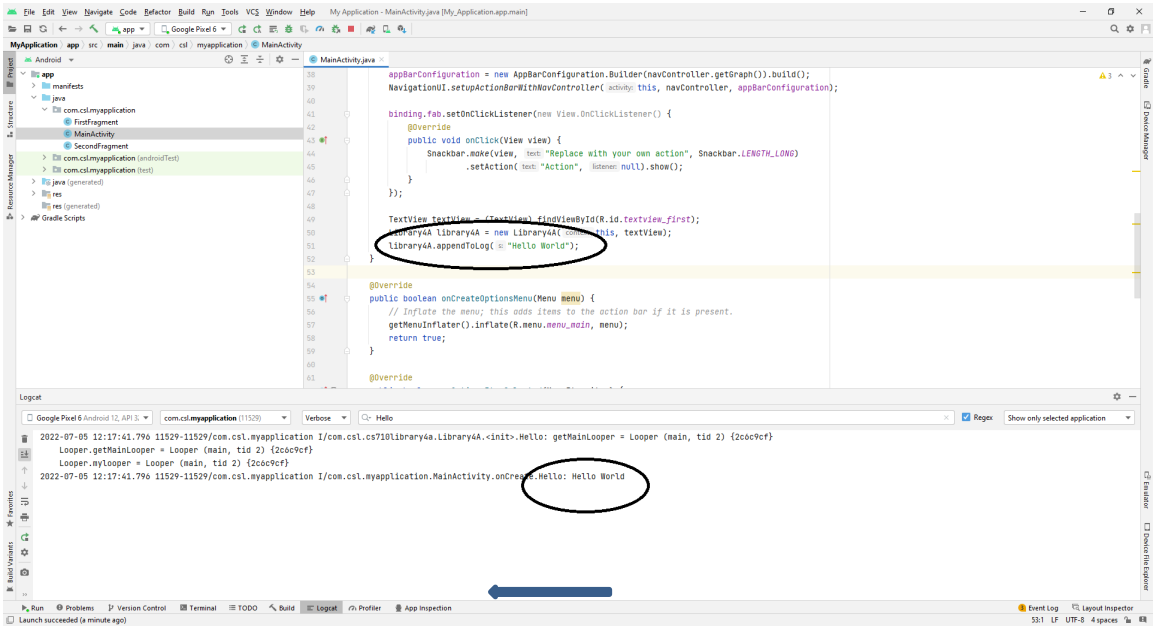


3. Download yourProject apps to an android device by selecting ‘Run’ -> ‘Run app’

Copyright Convergence Systems Limited 2022, All Rights Reserved



4. You should see the debug message which implies ok to use the library.



Copyright Convergence Systems Limited 2022, All Rights Reserved

3.6 How to Generate APK for Google Play

To generate apk for google play download and installation

1. Assumption: Google play console account has been setup. This requires account creation, payment and others. For details, please check for example https://support.shopgate.com/hc/en-us/articles/115004959928-Setting-up-a-Google-Play-Developer-Account

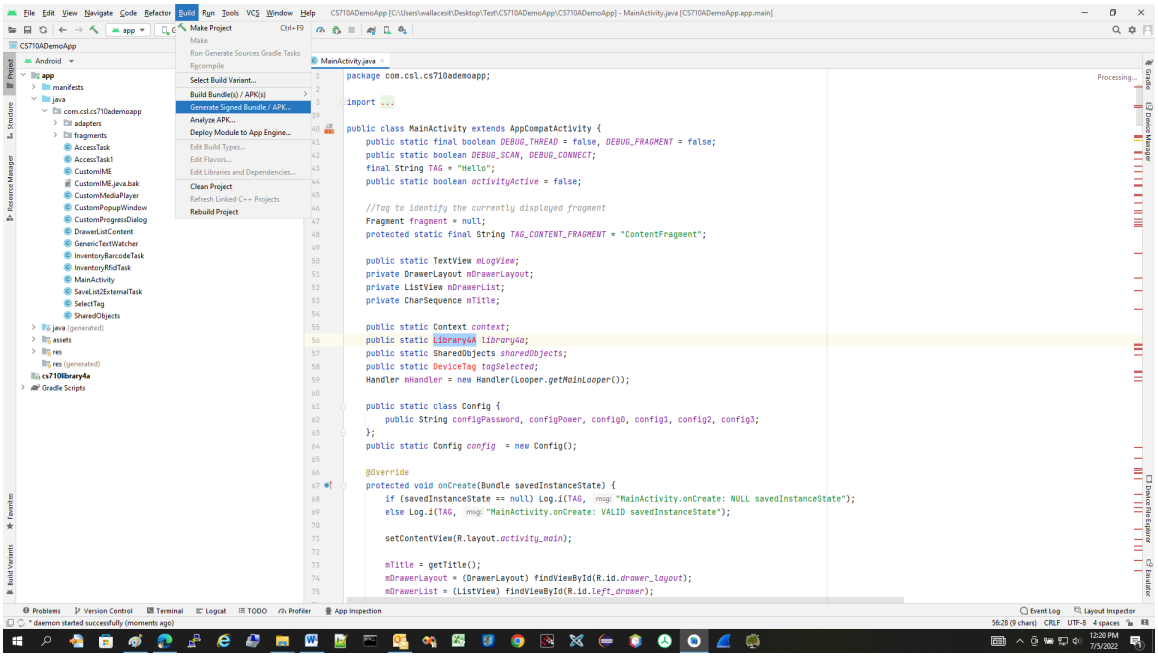
2. Source file preparation

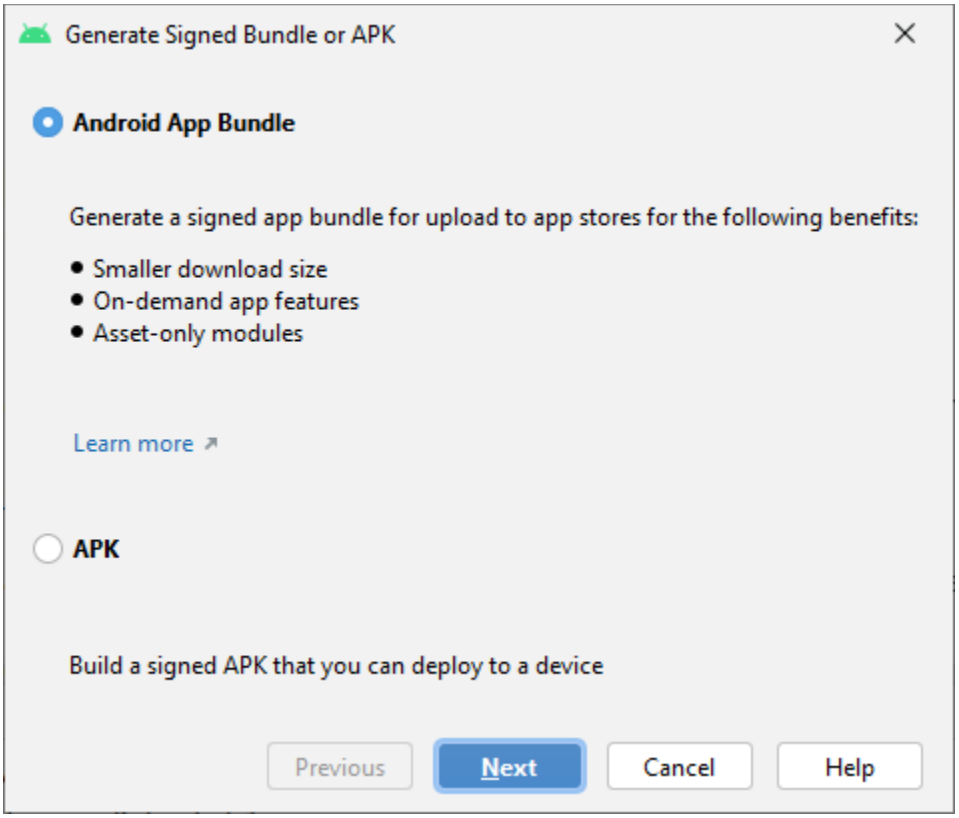
    i. If possible, remove debug message in the release source code.

    ii. Change the versionName value within build.gradle for the Module app. Change the versionCode value within build.gradle for the Module app VersionCode value is used by google play console to compare that of old app and to identify if new app is uploaded. VersionCode must be different and have value greater than the previous versionCode value.

Copyright Convergence Systems Limited 2022, All Rights Reserved

3. Select ‘Generate Signed Bundle/APK’ in ‘Build’ menu of Android Studio

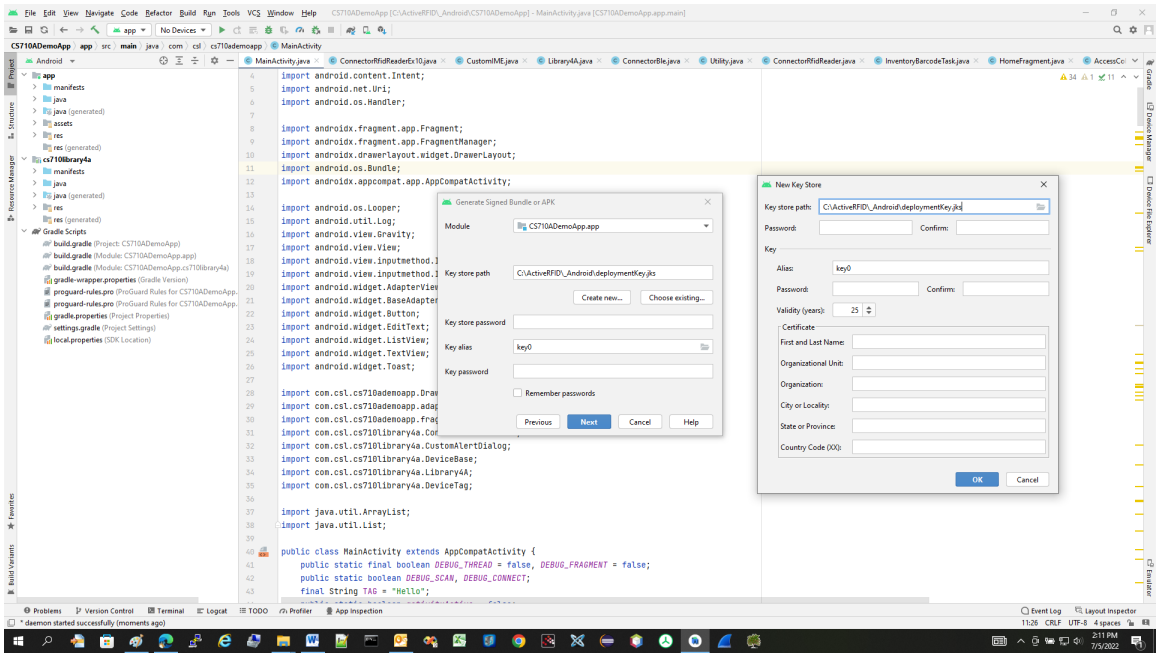




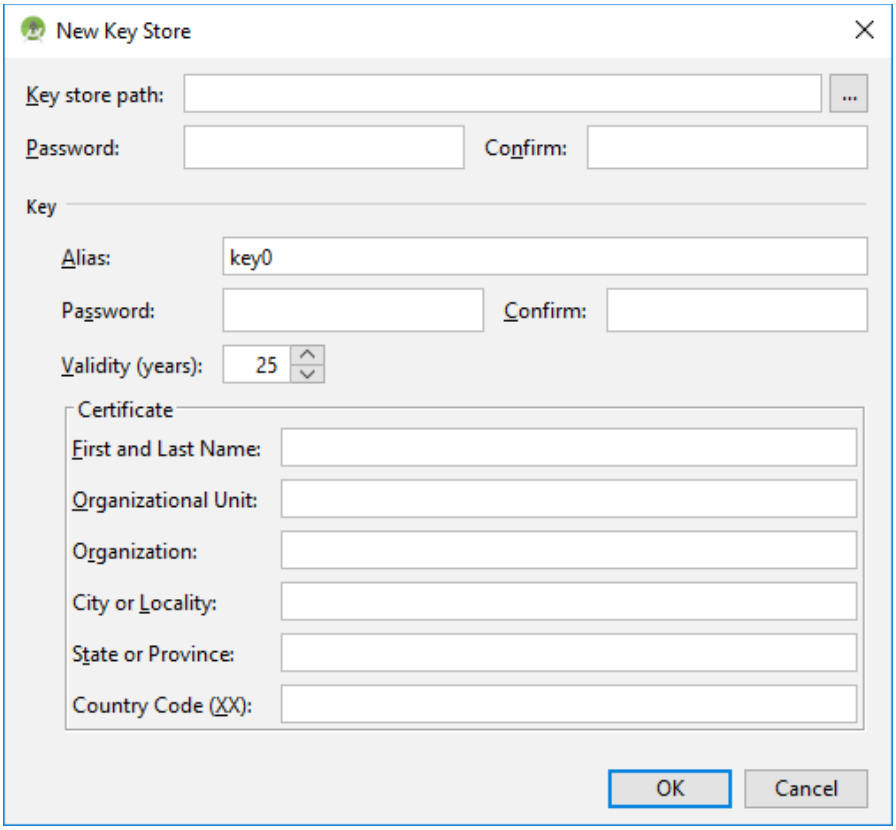
Select ether “bundle” or “apk”. “Bundle” will generate .aab file. “apk” will generate .apk file. Select “next”

Copyright Convergence Systems Limited 2022, All Rights Reserved

4. For the first signed APK generation, select “Create new…” to generate the upload key

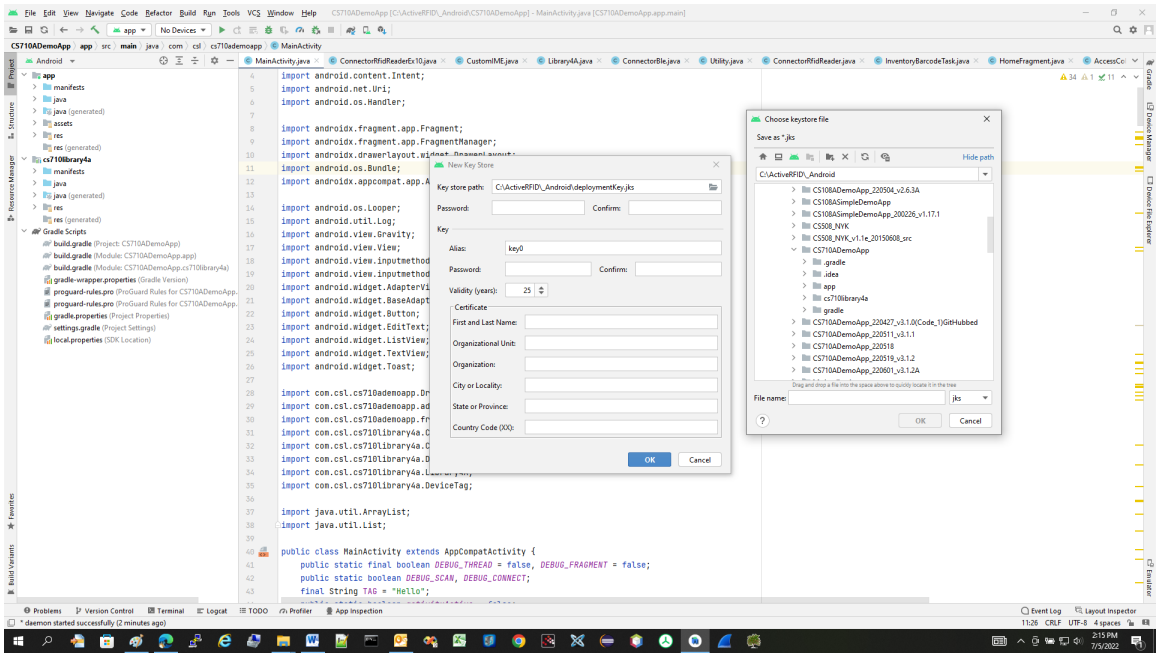


5. Select “….” In the row “Key store path”



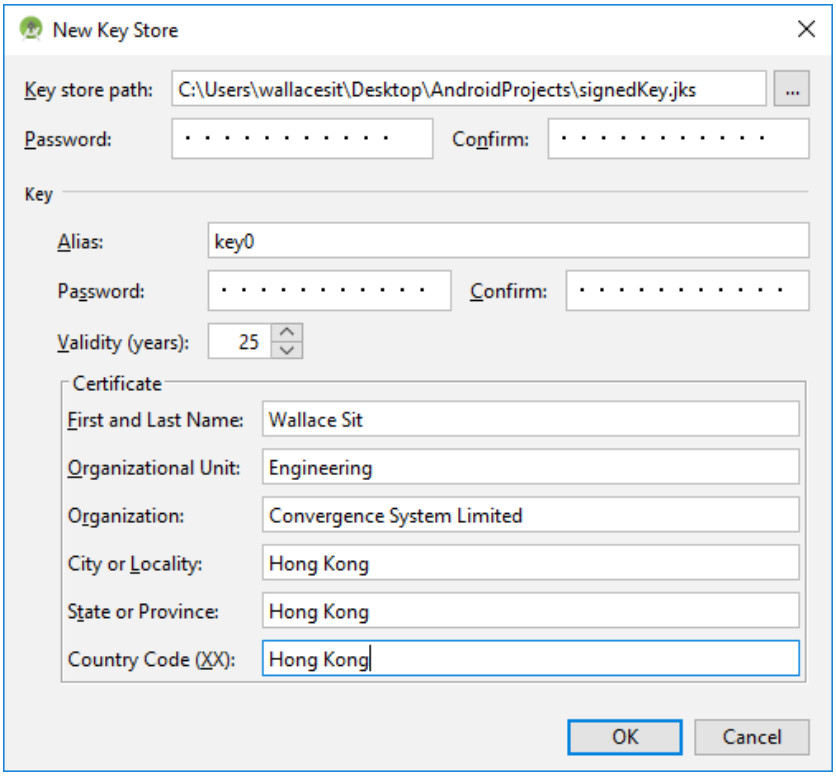
6. Select the location in the directory tree. Give name, such as signedKey in ‘File name’ row. Select ‘OK’

Copyright Convergence Systems Limited 2022, All Rights Reserved



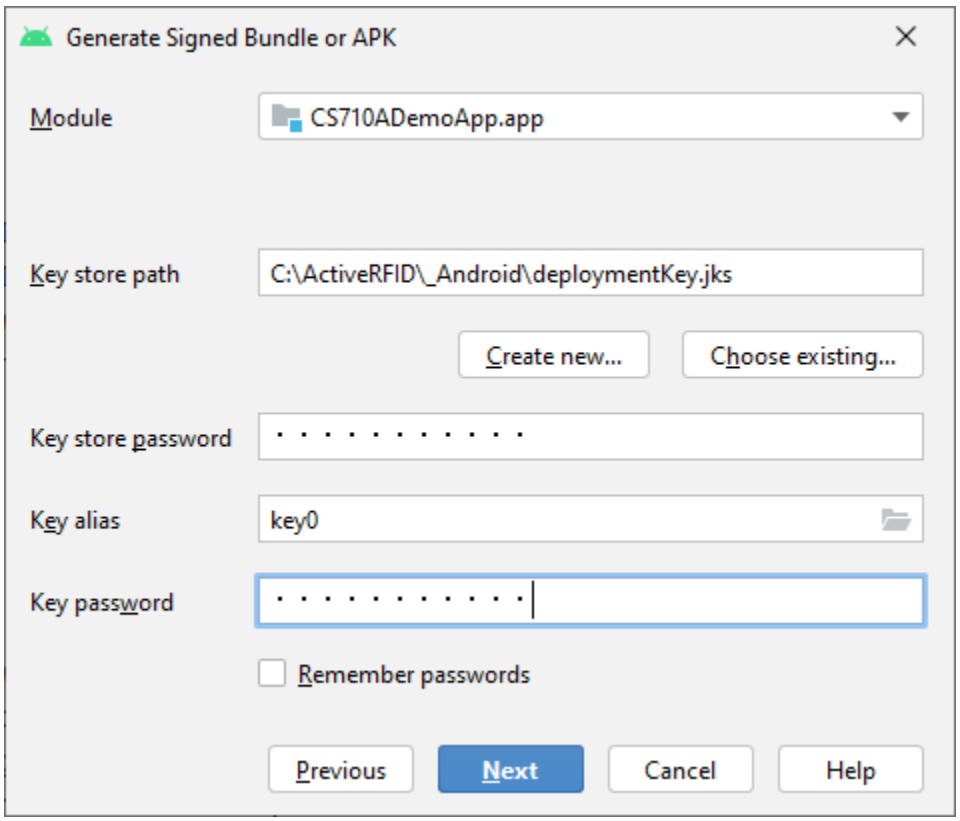
Copyright Convergence Systems Limited 2022, All Rights Reserved

7. Enter all fields and Select ‘OK’



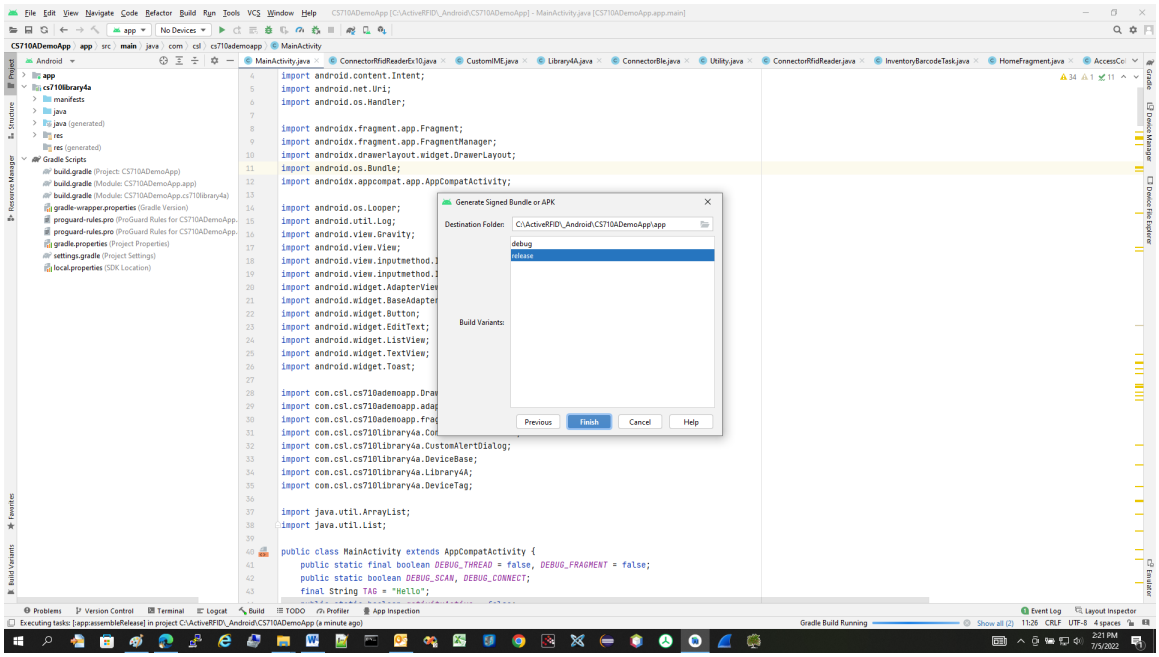
8. Enter passwords and Select ‘Next’

Copyright Convergence Systems Limited 2022, All Rights Reserved



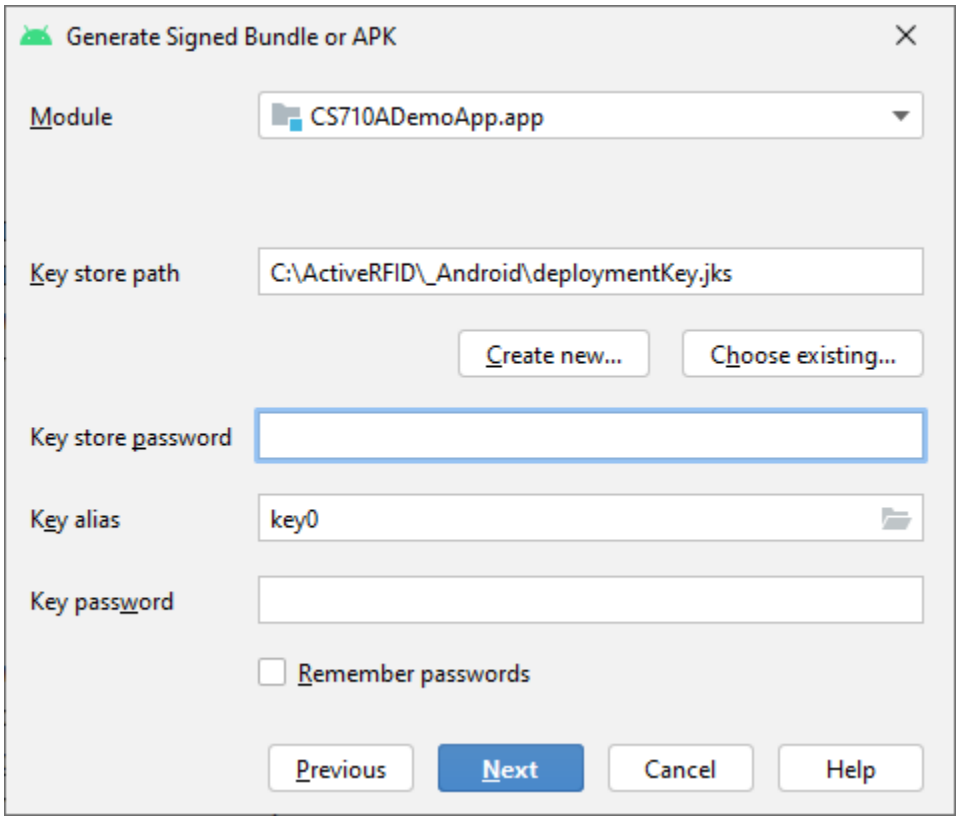
Copyright Convergence Systems Limited 2022, All Rights Reserved

9. Select “release” build variants. Then Select ‘Finish’



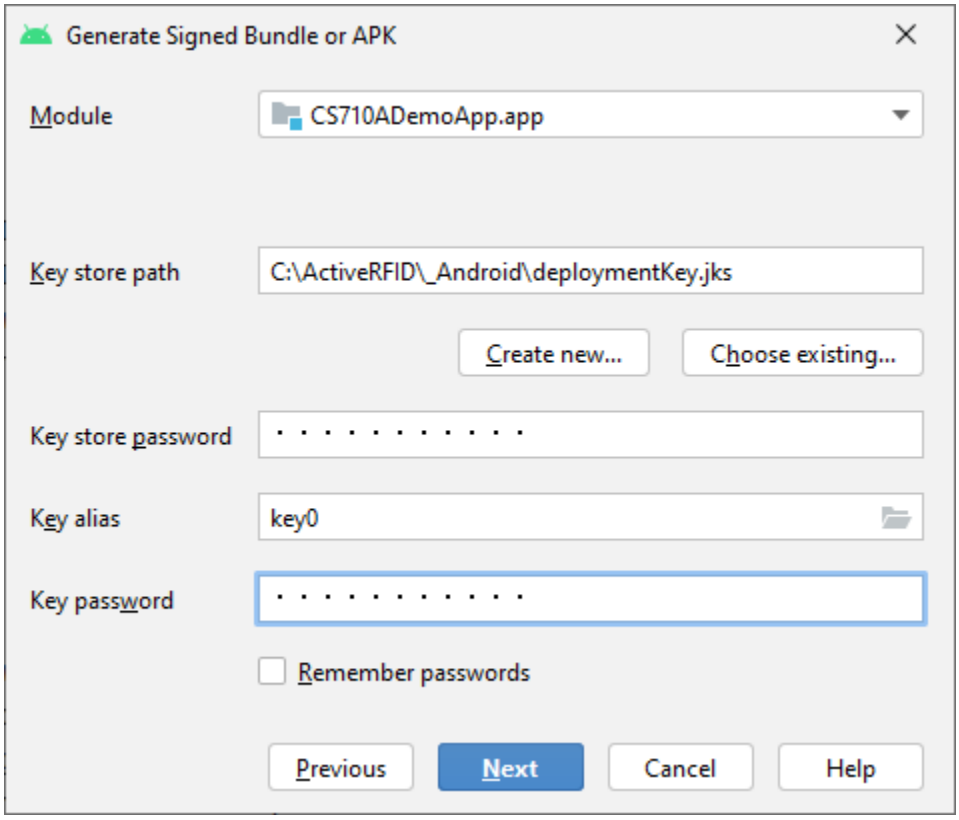
10. Wait for the end of ‘Gradle Build Running’ to create the signed APK

11. For the normal signed APK generation,



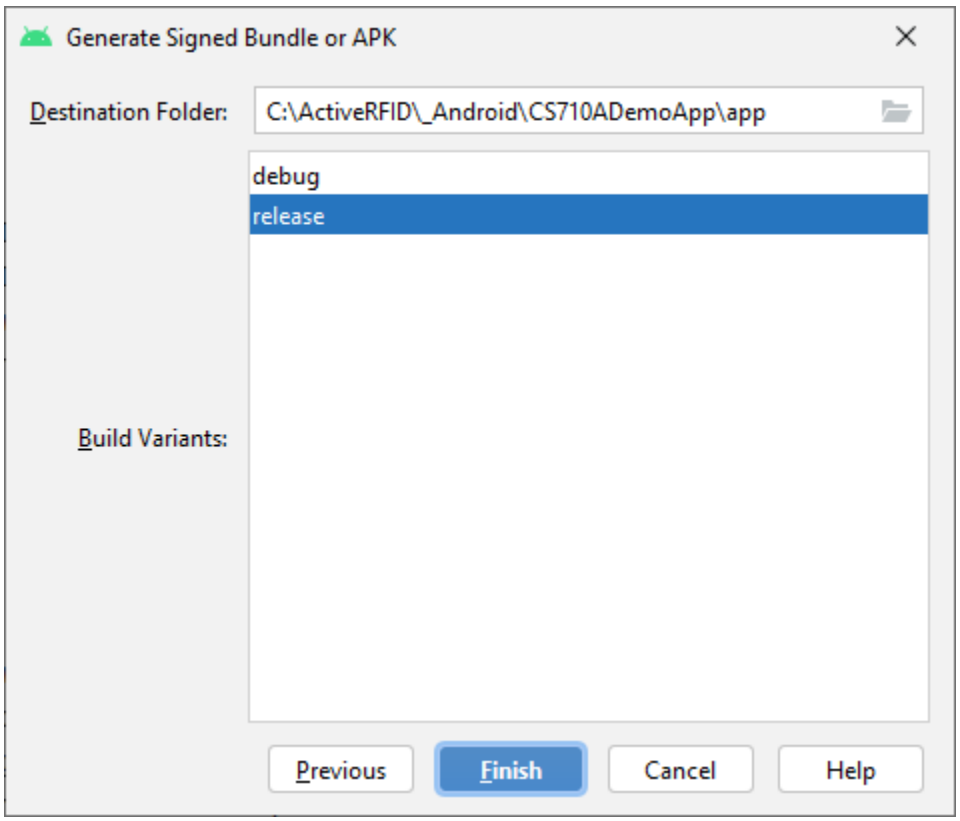
Copyright Convergence Systems Limited 2022, All Rights Reserved

12. Fill in the password fields and Select ‘Next’



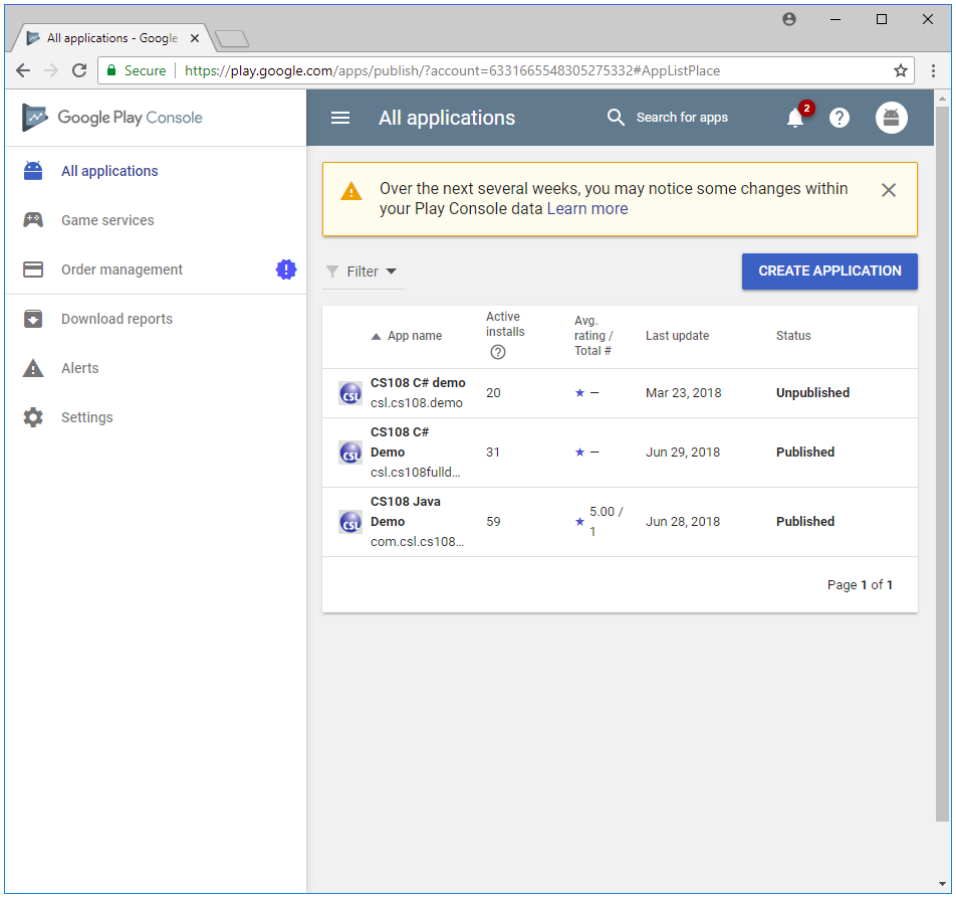
13. Select “release” build variant. Select ‘Finish’

Copyright Convergence Systems Limited 2022, All Rights Reserved



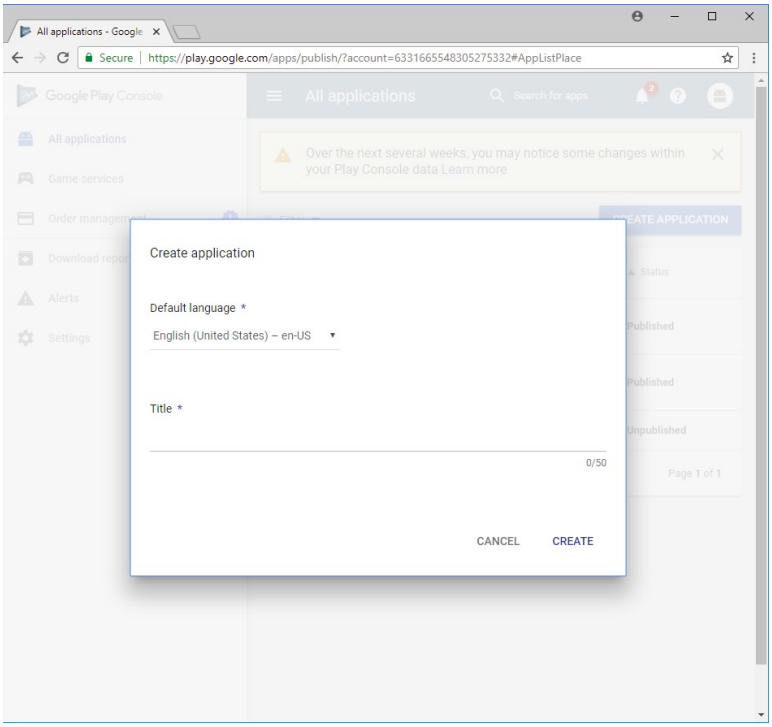
14. Wait the ending of 'Gradle Build Running' to create the signed APK
15. Signed APK file location: if successful, the file app-release.apk is in CS710AdemoApp\app\release.
16. File inventory: rename the app-release.apk to any name such as app-release-180718.apk with .apk extension.
17. File publish: within google play console. For the first time publish, select "Create application"

Copyright Convergence Systems Limited 2022, All Rights Reserved

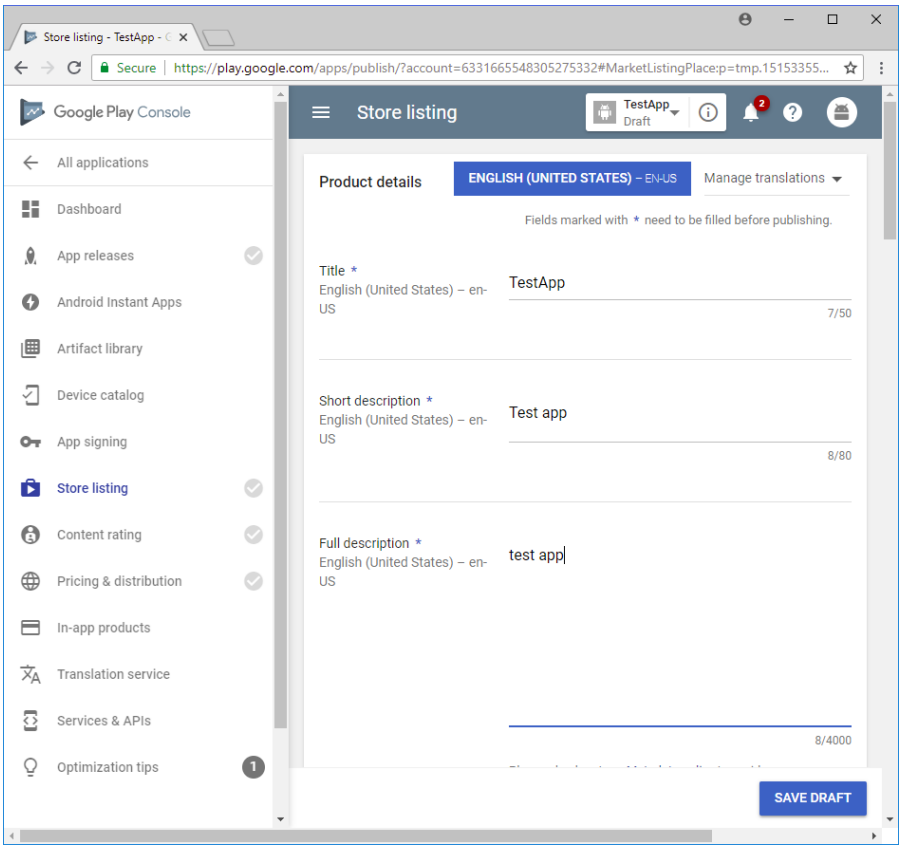


Copyright Convergence Systems Limited 2022, All Rights Reserved

18. Set language and Title. Then Select ‘Create’

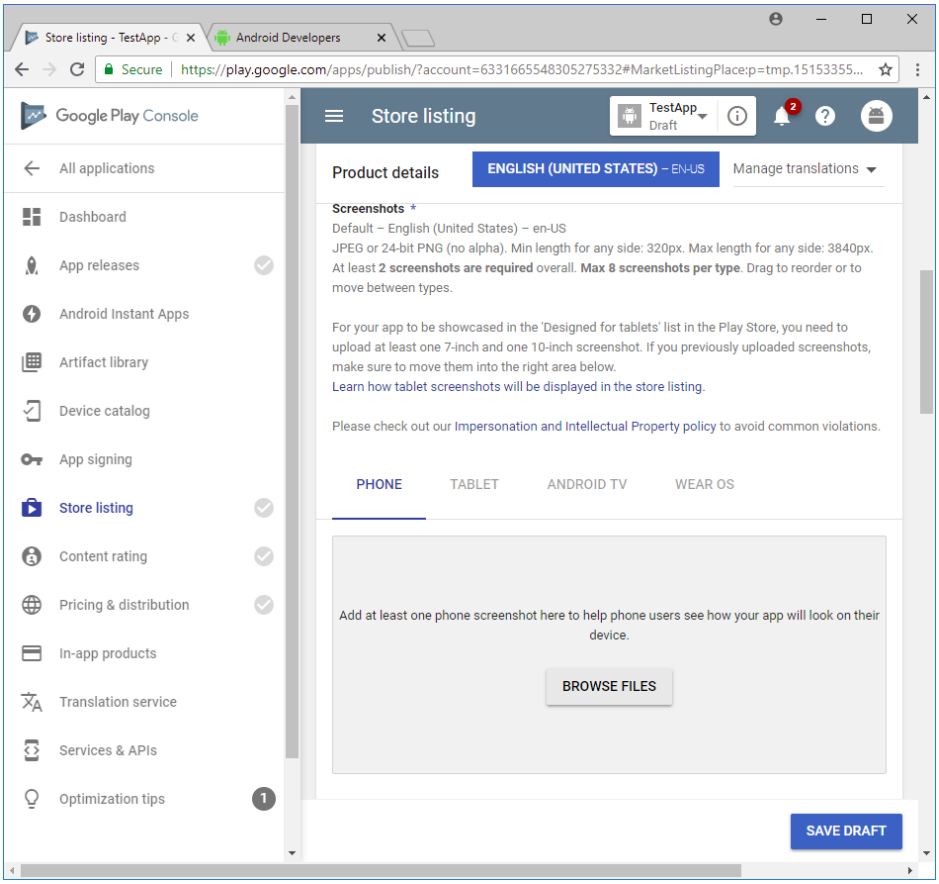


19. Fill in ‘Title’, ‘Short description’ and ‘Full description’.



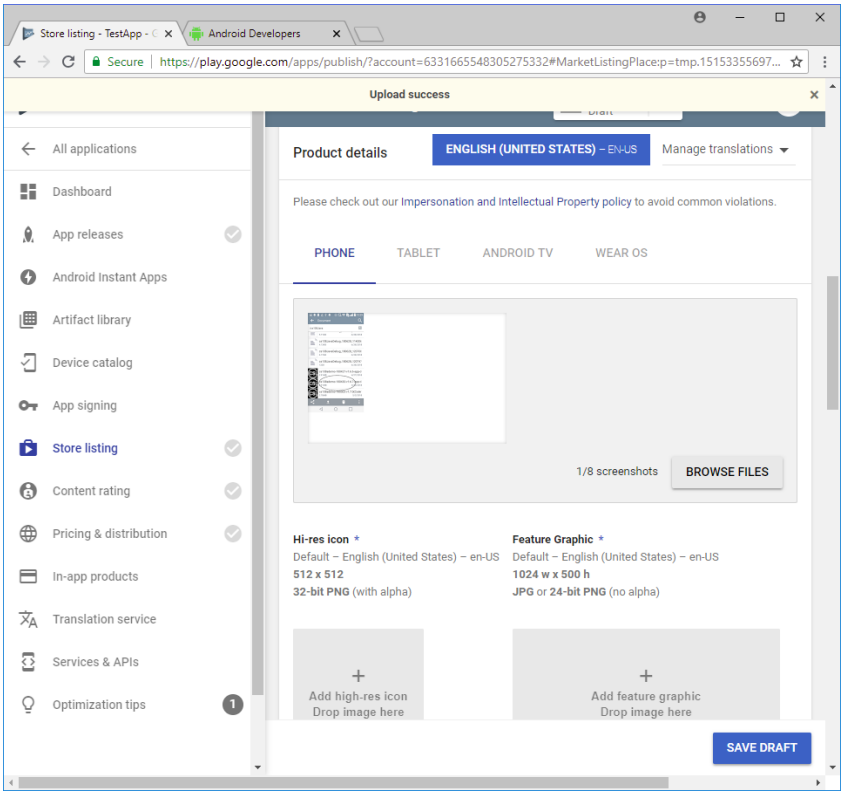
Copyright Convergence Systems Limited 2022, All Rights Reserved

20. Go downwards to find the Screenshots and Select ‘Browse Files’ to select some screen shot pictures.

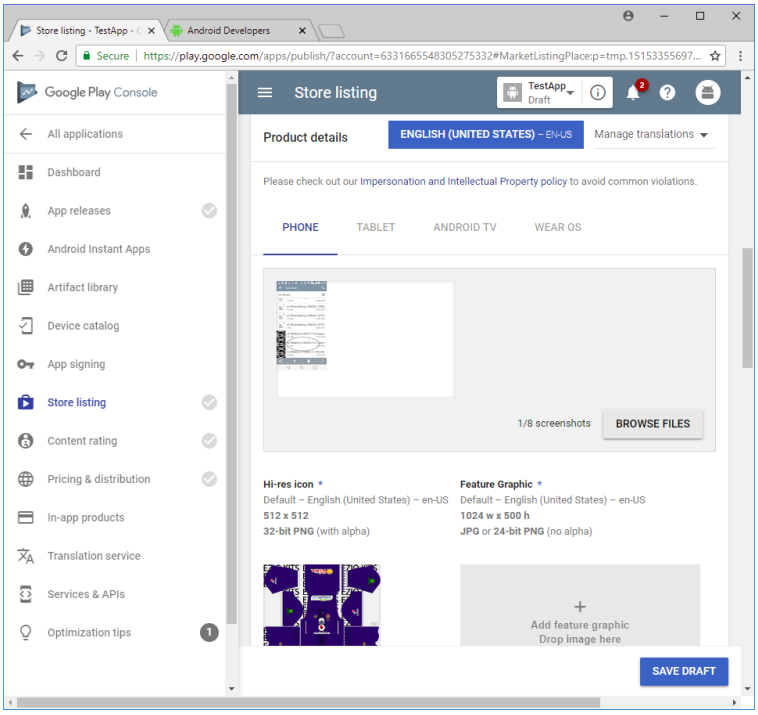


21. Select “Add high-res icons Drop image here” in ‘Hi-res icon’ section to select the picture with resolution 512x512.

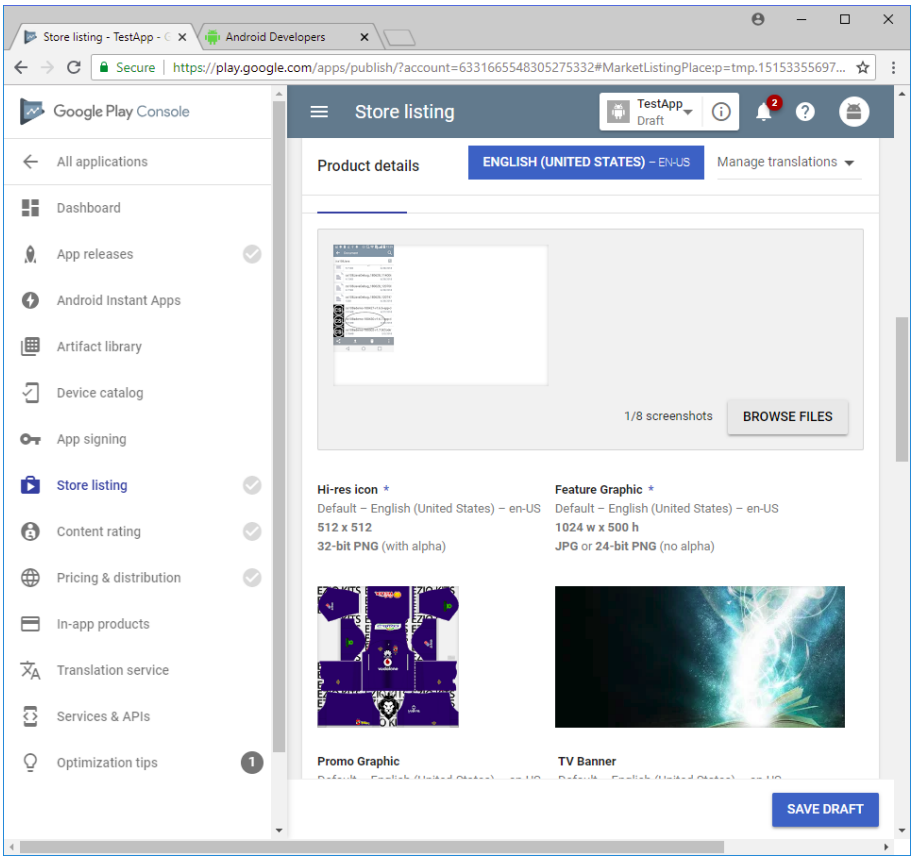
Copyright Convergence Systems Limited 2022, All Rights Reserved



22. Select ‘Add feature graphic. Drop image here’ icon in ‘Feature Graphic’ section to select the picture with resolution 1024x500.

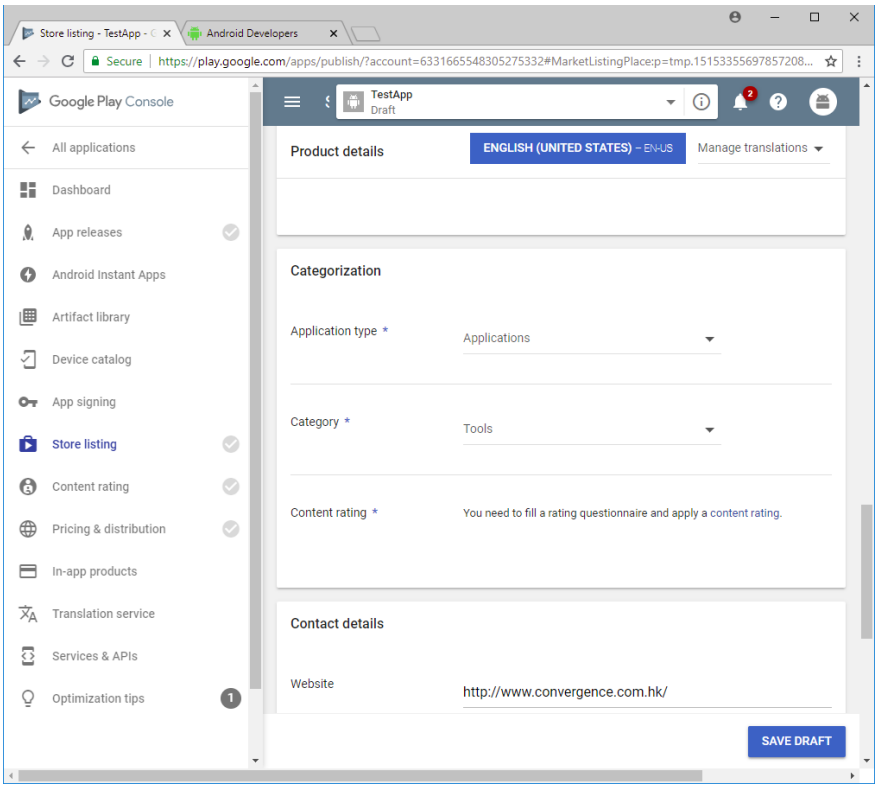


Copyright Convergence Systems Limited 2022, All Rights Reserved

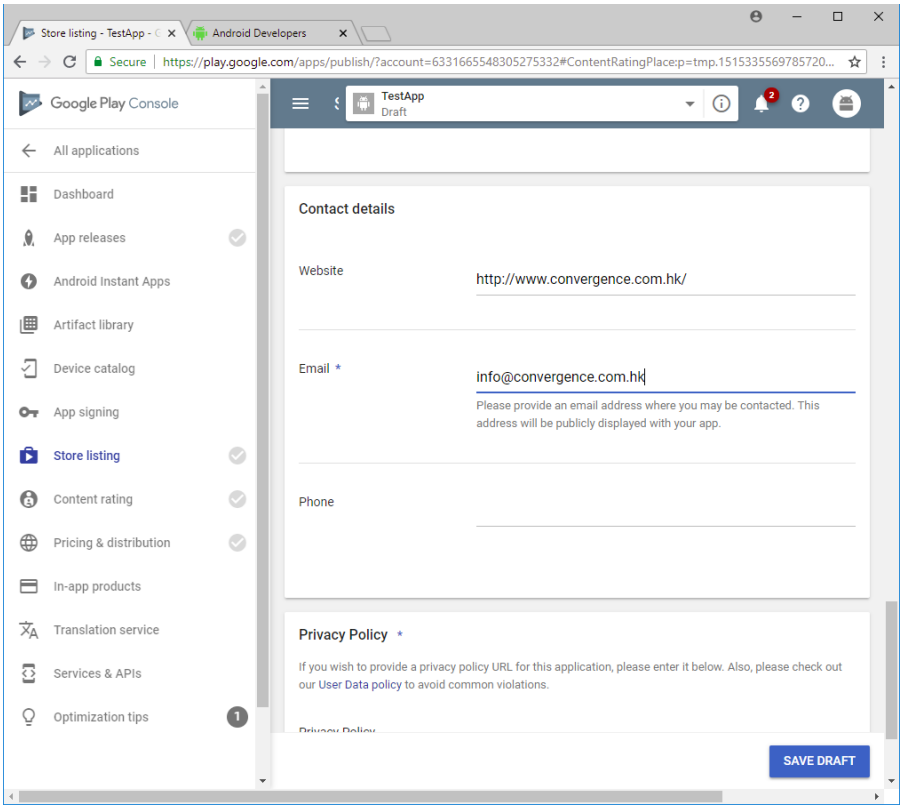


23. Go downwards to the ‘Categorization’ section. Select ‘Application type’, ‘Category’’ and then ‘content rating’.

Copyright Convergence Systems Limited 2022, All Rights Reserved



24. Go downwards to ‘Contact details’ and input ‘Email’ information

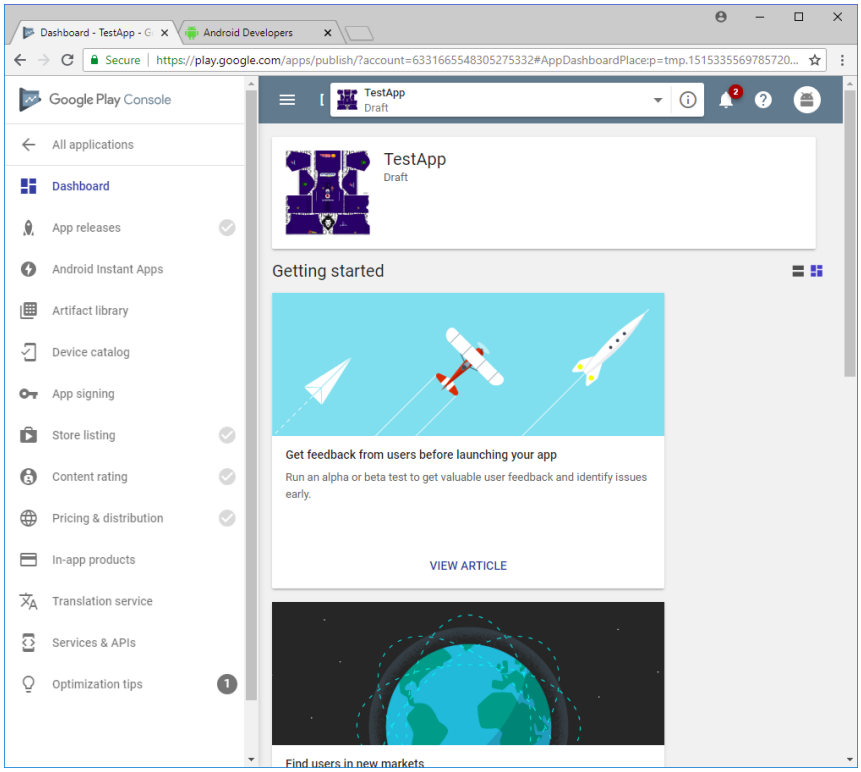


Copyright Convergence Systems Limited 2022, All Rights Reserved

25. Tick “Not submitting a privacy policy URL …” in the “Privacy Policy’ section.

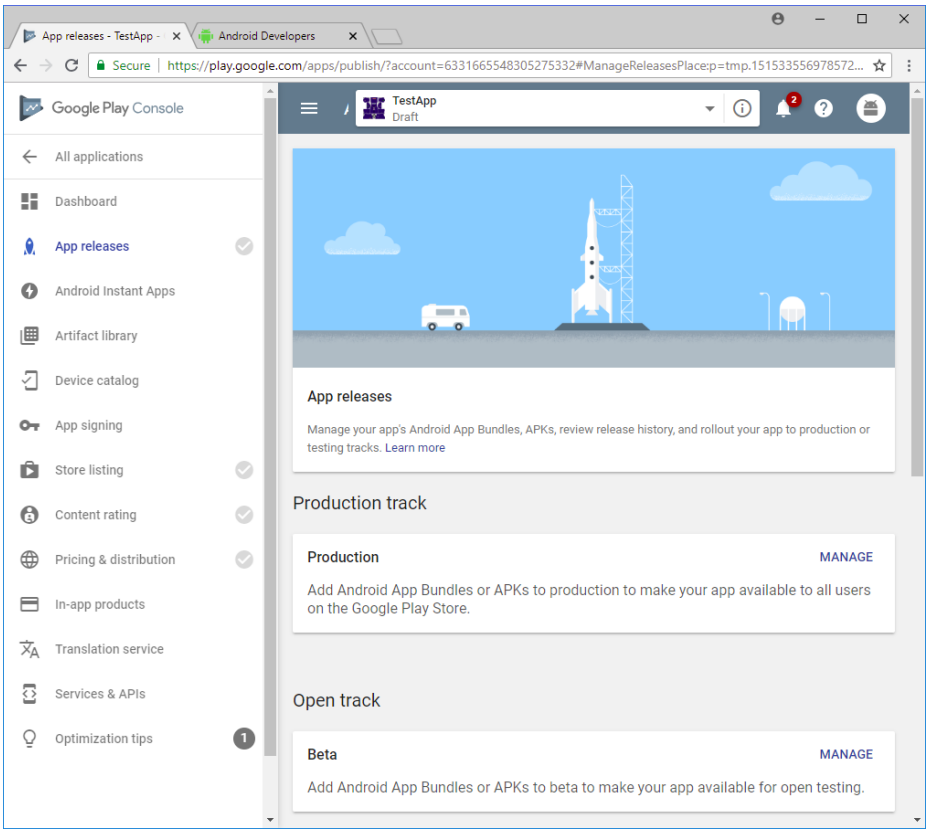
26. Press ‘Save Draft’

27. File publish: within google play console. Assume the application is created. Select the application, such as ‘TestApp’

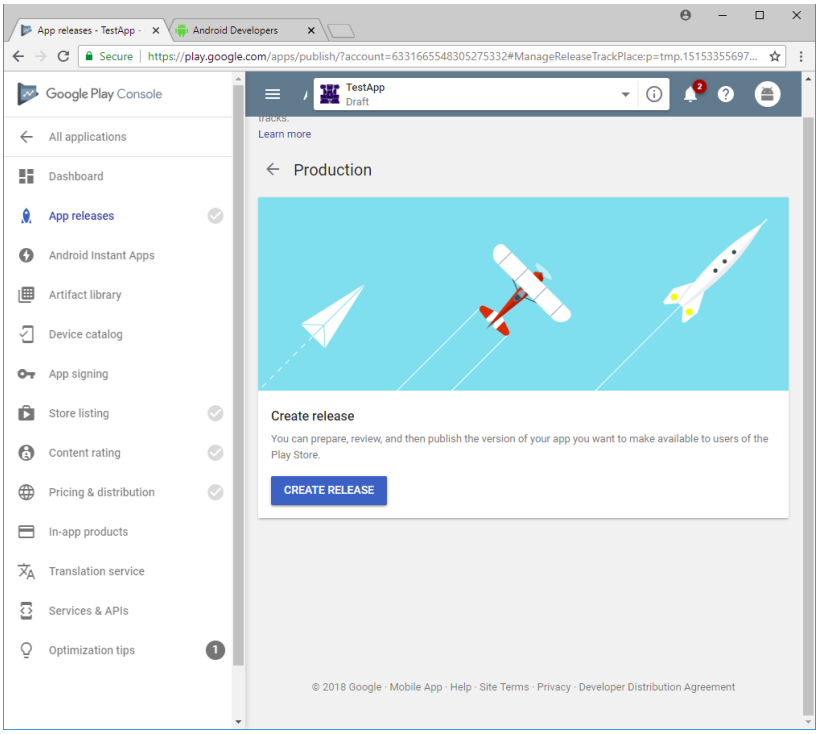


28. Select ‘App releases’ on the left hand side

Copyright Convergence Systems Limited 2022, All Rights Reserved

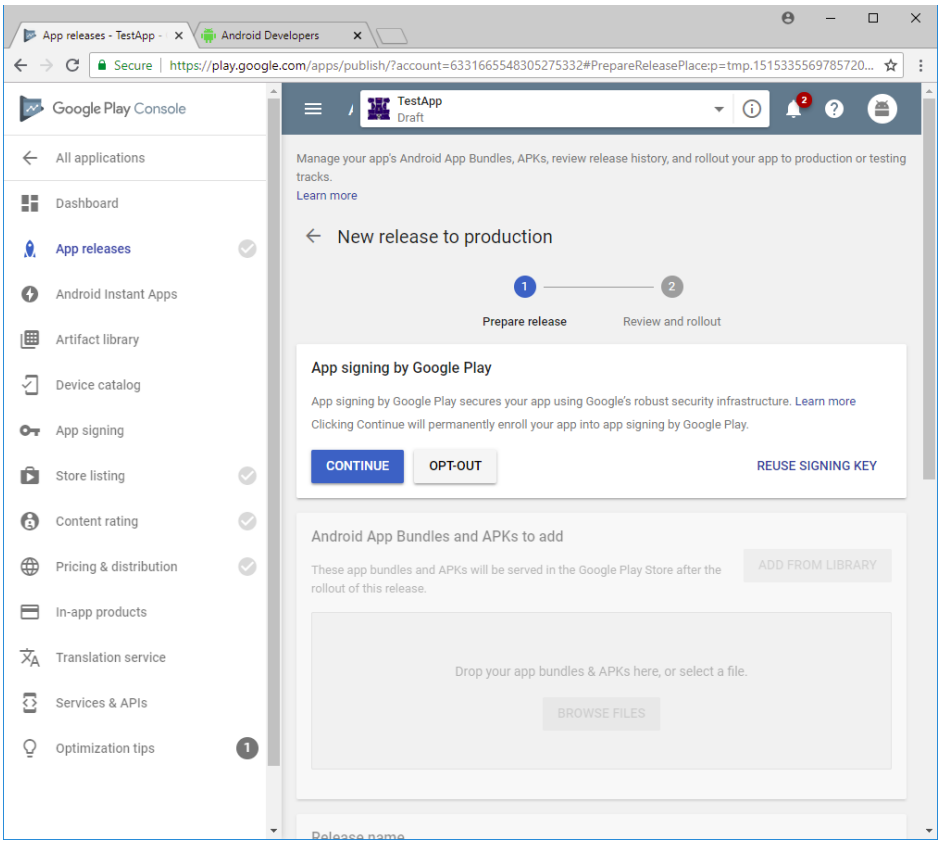


29. Select ‘Manage’ in ‘Production track’ -> ‘Production’ section

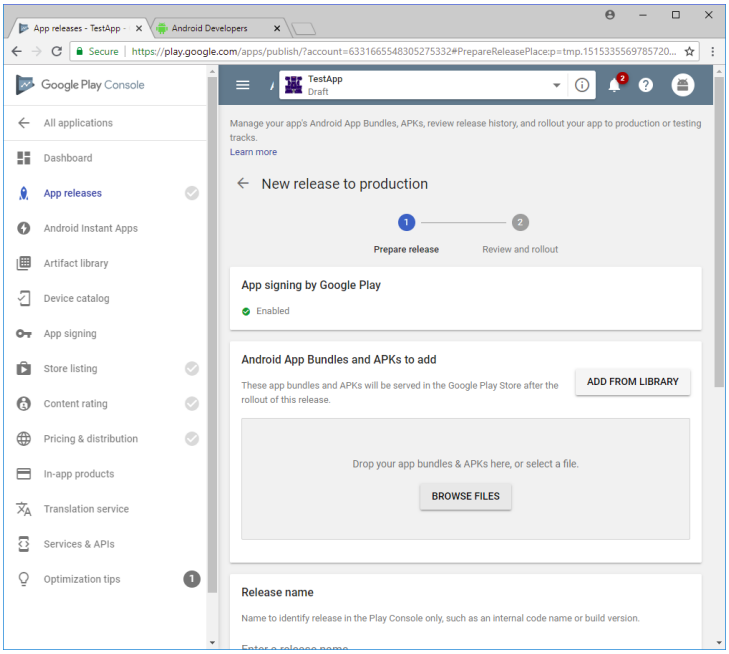


30. Select ‘Create Release’

Copyright Convergence Systems Limited 2022, All Rights Reserved



31. Select 'Continue'



32. Select 'Browse files' to select the signed APK file built previously.

33. Select 'Review' …. To finally publish the APK

Copyright Convergence Systems Limited 2022, All Rights Reserved

34. Apk installation from Google play: execute “Play” apps. Search and install your android apps.
35. End

Copyright Convergence Systems Limited 2022, All Rights Reserved

3.7 CS710S Android Demo App Classes

***** Note: the following information is updated as per Android Demo App 2.6.0**

Cs710ademoapp classes

Class name	Description
AccessTask	The class handles the AsyncTask to do RFID access operation
AccessTask1	The class is a superclass of AccessTask. It adds access retry, text formatting and deformatting and others.
CustomIME	The class handles the custom IME for keyboard wedge feature
CustomMediaPlayer	The class handles the custom media player to output sound
CustomPopupWindow	The class handles the custom popup window to display message
CustomProgressDialog	The class handles the custom progress dialog that are used for all Spinner UI component.
DrawerListContent	The class defines the drawer item list
GenericTextWatcher	The class handles the custom EditText UI component
InventoryBarcodeTask	The class handles the AsyncTask to do barcode inventory
InventoryRfidTask	The class handles the AsyncTask to do RFID inventory
MainActivity	This class is the entry point of the application
SaveList2ExternalTask	The class handles the task to save data to external files
SelectTag	The class handles the tag selected for further action (first used in AuraSense)
SharedObjects	This class stores the shared data of the application.

Fragment classes are called from MainActivity.

Fragment Class name	Description
AboutFragment	**The class handles the 'About' page UI**
AccessAuraSenseFragment	**The class handles the 'configuration' tab UI of 'AuraSense' page**
AccessAxzonConfigFragment	**The class handles the 'read' tab UI of 'Axzon' page**
AccessColdChainFragment	**The class handles the 'logging' tab UI with 'uEm4325' page.**
AccessEm4325PassiveFragment	**The class handles the 'One-shot' tab UI of 'uEm4325' page**
AccessFdmicroFragment	**The class handles the 'configuration' tab UI of 'FM13DT160' page**
AccessImpinjFragment	**The class handles the 'configuration' tab UI of 'Impinj FastID' page**

Copyright Convergence Systems Limited 2022, All Rights Reserved

AccessMicronFragment	**The class handles the 'configuration' tab UI with 'Axzon Magnus' page.**
AccessReadWriteFragment	**The class handles the 'Read/Write' page UI**
AccessReadWriteUserFragment	**The class handles the 'Large Sized memory read' page UI**
AccessRegisterFragment	**The class handles the 'Register Tag' page UI**
AccessSecurityFragment	**The class handles the 'Security' page UI**
AccessSecurityKillFragment	**The class handles the 'Kill' tab UI within 'Security' page**
AccessSecurityLockFragment	**The class handles the 'Lock' tab UI within 'Security' page**
AccessUcode8Fragment	**The class handles the 'configuration' tab UI of 'Ucode 8' page**
AccessUcodeFragment	**The class handles the 'configuration' tab UI of 'Ucode Dna' page**
AccessXerxesLoggerFragment	**The class handles the 'Logger' tab UI of 'Axzon' page**
AuraSenseFragment	**The class handles the 'AuraSense' page UI within 'Special Functions' page.**
AxzonFragment	**The class handles the 'Axzon' page UI within 'Special Functions' page.**
AxzonSelectorFragment	**The class handles the tag selection after selecting 'Axzon' or 'Axzon Magnus' within 'Special Functions' page and before going to AccessConfigFragment /AxzonFragment /MicronFragment**
ColdChainFragment	**The class handles the 'Cold Chain CS8300' page UI within 'Special Functions' page.**
CommonFragment	**The class is the base class of all fragment classes**
ConnectionFragment	**The class handles the 'Connection' page UI**
FdmicroFragment	**The class handles the 'FM13DT160' page UI**
HomeFragment	**The class handles the 'Main' page UI**
HomeSpecialFragment	**The class handles the 'Special Functions' page UI**
ImpinjFragment	**The class handles the 'Impinj FastID' page UI within 'Special Functions' page.**
InventoryBarcodeFragment	**The class handles the 'Barcode' tab UI of the 'Inventory' page.**
InventoryFragment	**The class handles the 'Inventory' page UI**
InventoryRfidMultiFragment	**The class handles the 'RFID' tab UI of the 'Inventory' page. The class also handles the 'Multi-bank inventory' page UI within 'Special Functions' page.**
InventoryRfidSearchFragment	**The class handles the 'Geiger Search' page UI**
InventoryRfidSimpleFragment	**The class handles the 'Simple Inventory' page**
MicronFragment	**The class handles the 'Axzon Magus' page UI within 'Special Functions' page.**
SettingAdminFragment	**The class handles the 'Administration' tab of the**

Copyright Convergence Systems Limited 2022, All Rights Reserved

	'Settings' page
SettingFilterFragment	**The class handles the 'Filter' page UI**
SettingFilterPostFragment	**The class handles the "Post-Filter' tab UI within 'Filter' page**
SettingFilterPreFragment	**The class handles the "Pre-Filter' tab UI within 'Filter' page**
SettingFilterRssiFragment	**The class handles the "Rssi-Filter' tab UI within 'Filter' page**
SettingFragment	**The class handles the 'Settings' page UI**
SettingOperateFragment	**The class handles the 'Operation' tab of the 'Settings' page**
Ucode8Fragment	**The class handles the 'Ucode 8' page UI within 'Special Functions' page.**
UcodeFragment	**The class handles the 'Ucode Dna' page UI within 'Special Functions' page.**
UtraceFragment	**The class handles the 'untrace' page UI**

Copyright Convergence Systems Limited 2022, All Rights Reserved

Chapter 4: Library Class and Method

Class Name: com.csl.cs710library4a.AesCmac

AesCmac(int length)	Initialize Aes calculation
init(Key key)	Initialize with the Aes key
void updateBlock(byte[] data)	Update Aes value
final byte[] doFinal()	Finalize Aes value

Class Name: com.csl.cs710library4a.ConnectorRfidReader

Class used by AccessTask in application

Class Name: com.csl.cs710library4a.CustomAlertDialog

boolean Confirm(Activity act, String Title, String ConfirmText, String OkBtn, Runnable okProcedure, String CancelBtn, Runnable cancelProcedure)	Start the custom dialog

Class Name: com.csl.cs710library4a.DeviceBase

Class Name: com.csl.cs710library4a.DeviceCsReader

Class Name: com.csl.cs710library4a.DeviceTag

DeviceBase implements Comparable<DeviceBase> (for barcode data)	DeviceBase(String name, String identity, boolean selected, int count)	Initialize device data record
	DeviceBase(String identity, boolean selected, int count)	Initialize device data record
	String getName()	Get name of data record
	void setName(String name)	Set name of data record
	String getIdentity()	Get address of data record

Copyright Convergence Systems Limited 2022, All Rights Reserved

	void setIdentity(String identity)	Set address of data record
	boolean getSelected()	Get selected status of the data record
	void setSelected(boolean selected)	Set selected status of the data record
	int getCount()	Get count of data record
	void setCount(int count)	Set count of data record
	int compareTo(ReaderDevice other)	Compare routine for sorting sharedObjects
DeviceCsReader extends DeviceBase (for scanning data – to be simplified)	DeviceCsReader(String name, String address, boolean selected, String details, int count, double rssi, int serviceUUID2p1)	Initialize device data record
	String getDetails()	**Get details of data record**
	void setDetails(String details)	**Set details of data record**
	String getPc()	**Get Pc of data record**
	String getXpc()	**Get Xpc of data record**
	setXpc(String strXpc)	**Set Xpc of data record**
	String getRes()	**Get Res of data record**
	String getRes2()	**Get Res2 of data record**
	String getEpc()	**Get Epc of data record**
	String getTid()	**Get Tid of data record**
	String getUser()	**Get User of data record**
	String getMdid()	**Get Mdid of data record**
	double getRssi()	**Get rssi of data record**
	void setRssi(double rssi)	**Set rssi of data record**
	int getPhase();	**Get phase of data record**
	void setPhase(int phase)	**Set phase of data record**
	int getChannel()	**Get channel of data record**
	void setChannel(int channel)	**Set channel of data record**
	int getPort()	**Get port of data record**
	void setPort(int port)	**Set port of data record**
	int getStatus()	**Get status of data record**
	void setStatus(int status)	**Set status of data record**
	int getBackport1()	**Get backport1 of data record**
	void setBackport1(int backport1)	**Set backport1 of data record**
	int getBackport2()	**Get backport2 of data record**
	void setBackport2(int	**Set backport2 of data**

Copyright Convergence Systems Limited 2022, All Rights Reserved

	backport1)	**record**
	int getCodeSensor()	**Get codesensor of data record**
	setCodeSensor(int codeSensor)	**Set codesensor of data record**
	int getCodeSensorMax()	**Get codesensorMax of data record**
	setCodeSensorMax(int codeSensorMax)	**Set codesensorMax of data record**
	int getCodeRssi()	**Get CodeRssi of data record**
	void setCodeRssi(int codeRssi)	**Set CodeRssi of data record**
	float getCodeTempC()	**Get CodeTempC of data record**
	void setCodeTempC(float codeTempC)	**Set CodeTempC of data record**
	String getBrand()	**Get Brand of data record**
	void setBrand(String brand)	**Set Brand of data record**
	int getSensorData()	**Get SensorData of data record**
	void setSensorData(int sensorData)	**Set SensorData of data record**
	String getstrExtra1()	**Get the string of extra bank 1**
	setExtra1(String strExtra1, int extra1Bank, int extra1Offset)	**Set the string of extra bank 1**
	String getstrExtra2()	**Get the string of extra bank 2**
	setExtra2(String strExtra2, int extra2Bank, int extra2Offset)	**Set the string of extra bank 2**
	void setExtra(String strExtra1, int extra1Bank, int extra1Offset, String strExtra2, int extra2Bank, int extra2Offset)	**Set ExtraData of data record**
	boolean isConnected()	**Check connection status of data record**
	void setConnected(boolean isConnected)	**Set connection status of data record**
	String getTimeOfRead()	**Get TimeOfRead of data record**
	String getTimeZone ()	**Get TimeZone of data**

Copyright Convergence Systems Limited 2022, All Rights Reserved

		record
	String getLocation ()	**Get Location of data record**
	void setLocation(String location)	**Set Location of data record**
	String getCompass ()	**Get Compass of data record**
	void setCompass(String compass)	**Set Compass of data record**
DeviceTag extends DeviceBase (for tag data – to be extends to DeviceCsReader, currently same as DeviceCsReader)	DeviceTag(String name, String address, boolean selected, String details, String strPc, String strXpc, String strCrc16, String strMdid, String strExtra1, int extra1Bank, int extra1Offset, String strExtra2, int extra2Bank, int extra2Offset, String strTimeOfRead, String strTimeZone, String strLocation, String strCompass, int count, double rssi, int phase, int channel, int port, int status, int backPort1, int backPort2, int codeSensor, int codeRssi, float codeTempC, String brand, int sensorData)	
	DeviceTag(String name, String address, boolean selected, String details, int count, double rssi, int serviceUUID2p1)	
	DeviceTag(String address, boolean selected, int count)	

Class Name: com.csl.cs710library4a.Library4A.AsyncTasknAsync

void cancel(boolean bCancel)	Cancel the task
boolean isCancelled()	Check if the task is cancelled
boolean isRunning()	Check if the task running
void execute()	Start the task
void publishProgress	Put the result to the UI material

Copyright Convergence Systems Limited 2022, All Rights Reserved

Class Name: com.csl.cs710library4a.Library4A.MultiBankData

int extra1Bank; int extra2Bank; int extra1Count; int extra2Count; int extra1Offset; int extra2Offset;	This is multibank register in Atmel controller

Class Name: com.csl.cs710library4a.Library4A

This library class provides some high-level function calls for CS710S Android Demo App. The description is updated for library cs710library4a version 3.1.3.

Public Constants

UplinkPacketTypes	TYPE_UPLINK_RESPONDED, TYPE_UPLINK1_RESPONDED, TYPE_BARCODE_GOODREAD, TYPE_COMMAND_BEGIN, TYPE_COMMAND_END, TYPE_18K6C_INVENTORY, TYPE_18K6C_INVENTORY_COMPACT, TYPE_18K6C_TAG_ACCESS, TYPE_ANTENNA_CYCLE_END, TYPE_ERRORCODE, TYPE_BARCODE_INVENTORY
HostCommands	NULL, CMD_WROEM, CMD_RDOEM, CMD_ENGTEST, CMD_MBPRDREG, CMD_MBPWRREG, CMD_18K6CINV, CMD_18K6CREAD, CMD_18K6CWRITE, CMD_18K6CLOCK, CMD_18K6CKILL, CMD_SETPWRMGMTCFG, CMD_UPDATELINKPROFILE, CMD_18K6CBLOCKWRITE, CMD_CHANGEEAS, CMD_GETSENSORDATA,

Copyright Convergence Systems Limited 2022, All Rights Reserved

	CMD_AUTHENTICATE, CMD_READBUFFER, CMD_UNTRACEABLE, CMD_FDM_RDMEM, CMD_FDM_WRMEM, CMD_FDM_AUTH, CMD_FDM_GET_TEMPERATURE, CMD_FDM_START_LOGGING, CMD_FDM_STOP_LOGGING, CMD_FDM_WRREG, CMD_FDM_RDREG, CMD_FDM_DEEP_SLEEP, CMD_FDM_OPMODE_CHECK, CMD_FDM_INIT_REGFILE, CMD_FDM_LED_CTRL, CMD_18K6CINV_SELECT, CMD_18K6CINV_COMPACT, CMD_18K6CINV_COMPACT_SELECT, CMD_18K6CINV_MB, CMD_18K6CINV_MB_SELECT
CsvColumn	RESERVE_BANK, EPC_BANK, TID_BANK, USER_BANK, PHASE, CHANNEL, TIME, TIMEZONE, LOCATION, DIRECTION, OTHERS
	INVALID_STATUS INVALID_BACKPORT INVALID_CODESENSOR INVALID_CODERSSI INVALID_SENSORDATA INVALID_CODETEMPC iNO_SUCH_SETTING byNO_SUCH_SETTING, strNO_SUCH_SETTING
TaskCancelReason	NULL, SAME_SETTING, INVALD_REQUEST, DESTORY, STOP, BUTTON_RELEASE,

Copyright Convergence Systems Limited 2022, All Rights Reserved

	ERROR, TIMEOUT, RFID_RESET
TagTypes	TAG_NULL, TAG_IMPINJ4, TAG_UCODE8, TAG_UCODEDNA, TAG_BAPCARD, TAG_COLDCHAIN, TAG_AURASENSE, TAG_MAGNUS, TAG_MAGNUS_S1, TAG_MAGNUS_S2, TAG_MAGNUS_S3, TAG_XERXES, TAG_FDMICRO, TAG_CTESIUS

Public data classes

CsReaderScanData	String deviceName, deviceAddress; int deviceBondState	Scanned device object
	int rssi	device rssi
	byte[] scanRecord	advertisement data
	String getDeviceName()	Get device name
	String getDeviceAddress()	Get device address
	int getDeviceBondState()	Get device bond state
	byte[] getScanRecord()	Get the device advertisement data
UplinkPacket	UplinkPacketTypes decodeDataType	E710 response package type
	byte[] dataValues	E710 response package data
	long decodedTime	decoded time field within response data
	double decodedRssi	decoded rssi field within response data
	int decodedPhase	decoded phase field within response data
	int decodedChidx	decoded channel index field within response data
	int decodedPort	decoded port field within response data
	byte[] decodedPc	decoded PC field within response data

Copyright Convergence Systems Limited 2022, All Rights Reserved

	byte[] decodedEpc	decoded EPC field within response data
	byte[] decodedCrc	decoded CRC field within response data
	byte[] decodedData1	Decoded data1 field within response data
	byte[] decodedData2	Decoded data2 field within response data
	String decodedResult	Decoded string available for user application if Ok
	String decodedError	Debug string to be displayed for user application if error
TagData	String strCrc16; String strPc, strXpc; String strEpc, strAddress; int portstatus; int backport1, backport2, codeSensor, codeRssi; String brand; float codeTempC; int iSensorData; String strExtra1, strExtra2; Library4A.MultiBankData multiBankData;	This is part of data object of DeviceTag. This is displayed as a list during inventory.

Public constructors

Library4A (Context context, TextView mLogView)	Initialize the class with parameters: Context: context, such as getActivity(), of the calling object mLogView: the textView in the calling object for debugging purpose.

Public interfaces

NotificationListener	Callback for the trigger button press or release

Public methods

General:

String getlibraryVersion()	Get the version of library

Copyright Convergence Systems Limited 2022, All Rights Reserved

boolean isVersionGreaterEqual (String version, int majorVersion, int minorVersion, int buildVersion)	check if the version is above specific version values.
void appendToLogView(String s)	Show the debug string to the EditText of the right hand side drawer.
void appendToLog(String s)	Show the debug string to the android studio LogCat window
String byteArrayToString(byte[] packet)	Convert the byte array to String
float decodeMicronTemperature(int iTag35, String strActData, String strCalData)	Decode the temperature in degree C from different Micron tag action data and calibration data
String strFloat16toFloat32(String strData)	Change the 16bit float data to normal 32bit float data
String str2float16(String strData)	Change the string to 16bit float data
String temperatureC2F (String strValue)	Convert temperature from degree C to F
String temperatureF2C (String strValue)	Convert temperature from degree F to C
void setSameCheck(boolean sameCheck1)	set if parameter setting check is same as stored value and skips repeated storage
void saveSetting2File()	save the user configurable parameters to file.
int getRssiDisplaySetting()	Get the rssi display type with output: 0 for dBuV, 1 for dBm
boolean setRssiDisplaySetting(int rssiDisplaySelect)	Set the rssi display type with parameter: 0 for dBuV, 1 for dBm
byte getPopulation2Q(int population)	get the Q value from population
int getPopulation()	get the tag population set
boolean setPopulation(int population)	set the tag population
public byte getQValue()	get the Q Value to be set
boolean setQValue(byte byteValue)	set the Q value
int getBeepCount()	Get beep delay count for tag received
boolean setBeepCount(int beepCount)	set beep delay count for tag received
boolean getInventoryBeep()	Get if beep is needed for RFID inventory
boolean setInventoryBeep(boolean inventoryBeep)	Set if beep is needed for RFID inventory with parameter: true for enable, false for disable
boolean getInventoryVibrateEnable()	Get if vibrator is needed for RFID inventory
boolean setInventoryVibrateEnable(boolean inventoryVibrate)	Set if vibrator is needed for RFID inventory with parameter: true for enable, false for disable
int getVibrate4AllSetting()	Get the vibrator operation mode with output: 0 for vibrate for new tag, 1 for vibrate for all tag
boolean setVibrate4AllSetting(int vibrateModeSelect)	Get the vibrator operation mode with parameter: 0 for vibrate for new tag, 1 for vibrate for all tag
int getVibrateTime()	Get vibration time in ms for RFID inventory
boolean setVibrateTime(int vibrateTime)	Set vibration time in ms for RFID inventory
int getVibrateWindow()	Get the vibration window in second for RFID inventory
boolean setVibrateWindow(int vibrateWindow)	set the vibration window in second for RFID

Copyright Convergence Systems Limited 2022, All Rights Reserved

	inventory
boolean setVibrateOn(boolean on)	turn the vibrator on or off parameter: on: true for on. False for off.
boolean getSaveFileEnable()	Get if inventory data is saved to external file
boolean setSaveFileEnable(boolean saveFileEnable)	Set if inventory data is saved to external file with parameter: true for enable, false for disable
boolean getSaveCloudEnable()	Get if inventory data is saved to cloud server
boolean setSaveCloudEnable(boolean saveCloudEnable)	Set if inventory data is saved to cloud server with parameter: true for enable, false for disable
boolean getSaveNewCloudEnable()	Get if inventory NEW data is saved to cloud server
boolean setSaveNewCloudEnable(boolean saveNewCloudEnable)	Set if inventory NEW data is saved to cloud server with parameter: true for enable, false for disable
boolean getSaveAllCloudEnable()	Get if inventory ALL data is saved to cloud server
boolean setSaveAllCloudEnable(boolean saveNewCloudEnable)	Set if inventory ALL data is saved to cloud server with parameter: true for enable, false for disable
String getServerLocation()	Get the cloud server location
boolean setServerLocation(String serverLocation)	Set the cloud server location
int getServerTimeout()	Get the cloud connection timeout in second
boolean setServerTimeout(int serverTimeout)	Set the cloud connection timeout in second
int getSavingFormatSetting()	Get the saving format
boolean setSavingFormatSetting(int savingFormatSelect)	Set the file saving format
int getCsvColumnSelectSetting()	Get columns selected in csv format
boolean setCsvColumnSelectSetting(int csvColumnSelect)	Set the columns selected in csv format
boolean getTriggerReporting()	Check if trigger auto reporting is enabled
boolean setTriggerReporting(boolean triggerReporting)	Enable/Disable trigger auto reporting
short getTriggerReportingCount()	Get the period of trigger auto reporting
boolean setTriggerReportingCount(short triggerReportingCount)	Set the period of trigger auto reporting
String checkVersion()	Check different firmware versions.
int getTriggerCount()	Get the number of trigger received
double dBm_to_dBuV(double value)	Conversion from dBm to dBuV
double dbuV_to_dBm(double value)	Conversion from dBuV to dBm
String deformatWriteAccessData(String strIn)	Routine to deformat the format in
boolean isWriteExtStoragePermitted(boolean requestPermission)	Check if write external file is permitted
String openWriteFile(String strFileNameHeader)	Open a file for writing
File getWriteFileOpened()	Get the opened file for writing data
boolean isModifiedEntry()	Check if the text box in Setting page is modified before
void resetModifiedEntry()	Reset the modified status of the textbox

Copyright Convergence Systems Limited 2022, All Rights Reserved

void showConfigFile()	Show the configuration parameters saved in Setting page
void showRxGainChangeLog()	Show the log data when the RxGain is changed before in Setting page

Bluetooth related:

String getBluetoothICFirmwareVersion()	get the bluetooth firmware version in string, in sequence of major, minor and build versions separated by dot character.
String getLinkedDeviceName()	get the bluetooth name
boolean setLinkedDeviceName(String name)	set the bluetooth name
boolean isScanning()	check if scanning bluetooth devices. Return: true implies scanning
boolean scanLeDevice(final boolean enable)	Start/stop scanning Bluetooth devices with parameters: enable: true to start scanning. False to stop scanning
boolean isConnected()	check if the Bluetooth is connected return: true implies connected.
boolean connect(DeviceCsReader tagDevice)	start Bluetooth connection with parameter: readerDevice: the selected scanned bluetooth device
boolean connectAgain()	Do disconnect and connect again
void disconnect(boolean tempDisconnect)	start bluetooth disconnection with parameter: tempDisconnect: true to disconnect temporarily. False to disconnect permanently.
String getLinkedDeviceAddress()	get the mac address of the connected bluetooth device
int getLinkedDevicePortNumber()	get the name of the connected bluetooth device
int getRssi()	get the rssi signal strength of the bluetooth connection in unit –dbm.
long getStreamInRate()	get the data rate of incoming data in byte per second.
CsReaderScanData getNewDeviceScanned()	Get new scanned device information with output: null for nothing.
int getMtu()	Get the MTU value of the connection
void turnOnLocationDevice(boolean onStatus)	turn on location device
void turnOnSensorDevice(boolean onStatus)	turn on Ecompass sensor device

Copyright Convergence Systems Limited 2022, All Rights Reserved

String getTimeStamp()	Get the time stamp
String getLocation()	Get the global location data
String getEcompass()	Get the Ecompass data

Host Processor Firmware related:

String getHostProcessorICSerialNumber()	get the serial number of the reader
String getHostProcessorICBoardVersion()	get the main board version
String hostProcessorICGetFirmwareVersion()	Get the Firmware version of the host processor

Notification related:

boolean batteryLevelRequest()	request to update the battery level immediately. Return: true for success request.
boolean setAutoRFIDAbort(boolean enable)	set automatic RFID inventory Abort
boolean getAutoRfidAbortRequest()	Get status of automatic RFID inventory Abort
boolean setAutoBarStartSTop (boolean enable)	set automatic Barcode start/stop
getAutoBarStartSTopRequest()	Get the status of automatic Barcode start/stop
String getBatteryDisplay(boolean voltageDisplay)	Get the display information about battery status with parameter: true for voltage display. False for percentage display
public String isBatteryLow()	Check if battery is low. If output string is null, Ok. Otherwise, it shows the battery level percentage.
int getBatteryCount()	get the battery reported index which is used to check if battery is updated.
int getBatteryDisplaySetting()	get the battery display type. 0 for voltage display, 1 for percentage display
boolean setBatteryDisplaySetting(int batteryDisplaySelect)	set the battery display type with parameters: 0 for voltage display, 1 for percentage display
boolean getTriggerButtonStatus()	get the latest trigger button status received from the reader.
boolean getTriggerStatusRequest()	Get the immediate trigger status
void setNotificationListener(NotificationListener listener)	set the listener routine for the trigger button.
Library4A.UplinkPacket readCsReaderUplinkPacketDecoded()	Read the RFID uplink data

Copyright Convergence Systems Limited 2022, All Rights Reserved

Barcode detector related:

boolean isBarcodeFailure()	check if barcode module is failure
String getBarcodeSerial()	get the serial number of the barcode module
boolean getBarcodeOnStatus()	get the on status of the barcode device
boolean setBarcodeOn(boolean on)	turn the barcode detector on or off parameter: on: true for on. False for off.
boolean barcodeSendCommandTrigger()	send command data to set trigger mode in the barcode module
boolean barcodeSendCommandSetPreSuffix()	send command data to set a pre-defined prefix and suffix in the barcode module
boolean barcodeSendCommandResetPreSuffix()	send command data to clear the prefix and suffix in the barcode module
boolean barcodeSendCommandConinuous()	send command data to set continuous mode
boolean barcodeInventory(boolean start)	start/stop barcode inventory
Library4A.UplinkPacket readCsReaderUplinkPacketDecoded()	Get new barcode inventoried information with output: null for nothing.
String getBarcodeVersion()	Get barcode firmware version
String getBarcodeESN()	Get ESN of barcode
String getBarcodeDate()	Get firmware date of the barcode

RFID detector related:

boolean getRfidOnStatus()	check if RFID detector is on or off. Return: true if on
boolean isRfidFailure()	check if barcode module is failure
int mToWriteSize()	get the length of the data queuing to send to RFID detector.
String getRadioSerial()	get the serial number of the radio module
String getRadioChipSerial()	Get the serial number of the E710 chip
String getRadioBoardVersion()	get the board version of the radio module
String getMacVer()	get the Mac version in string, in sequence of major, minor and build version, separated by dot character.
String getModelNumber()	get the model number of the radio

Copyright Convergence Systems Limited 2022, All Rights Reserved

	module
void setImpinJExtension(boolean tagFocus, boolean fastId)	void setImpinJExtension(boolean tagFocus, boolean fastId)
int getAntennaSelect()	Get the antenna port selected for the csModel
boolean setAntennaSelect(int number)	Set the antenna port selected for the csModel
boolean getAntennaEnable()	Get the antenna enable status for the csMode
boolean setAntennaEnable(boolean enable)	Enable/Disable the antenna for the csModel
long getAntennaDwell()	get the antenna dwell time
boolean setAntennaDwell(long antennaDwell)	set the antenna dwell time parameter: antennaDwell in unit of 1ms.
long getAntennaPower()	get the antenna power level
boolean setAntennaPower(long pwrlevel)	set the antenna power level parameter: pwrlevel in unit of 0.1dbm, up to 300.
int getQueryTarget()	get the query target value set for inventory.
int getQuerySession()	get the query session value set for inventory.
int getQuerySelect()	get the query select value set for inventory.
boolean setTagGroup(int sL, int session, int target1)	set query group parameters parameters: sL: set the query select value for the inventory. From 0 to 3. Session: set the query session for the inventory. From 0 to 3 Flag: set the query target for the inventory. 0 for A. others for B.
int getTagFocus()	get the tag focus value set for inventory.
int getFastId()	get the tag fastId value set for inventory.
boolean setTagFocus(boolean tagFocusNew)	set the tag focus value set for inventory.
boolean setInvBrandId(boolean invBrandId)	Set the brand ID for brand checking of the tag information.
boolean getDynamicAlgo()	get the query algorithm set for the inventory.

Copyright Convergence Systems Limited 2022, All Rights Reserved

boolean setInvAlgo(boolean dynamicAlgo)	set the query algorithm for the inventory. Parameter: invAlgo: false implies fixed. True implies dynamic algorithm.
List<String> getProfileList()	Get the list of available profiles
int getCurrentProfile()	get the query profile set for the inventory.
boolean setCurrentLinkProfile(int profile)	set the query profile for the inventory.
String getEnvironmentalRSSI()	Get the environmental RSSI read in the RFID module.
void resetEnvironmentalRSSI()	Reset the environmental RSSI read in the RFID module.
int getHighCompression()	Get High Compression value for inventory
int getRflnaGain()	Get RF LNA Gain value for inventory
int getIflnaGain ()	Get IF LNA Gain value for inventory
int getAgcGain ()	Get AGC Gain value for inventory
int getRxGain ()	Get RX Gain value for inventory
boolean setRxGain(int highCompression, int rflnagain, int iflnagain, int agcgain)	Set HighCompression, RF LNA Gain, IF LNA Gain, AGC Gain values for inventory
byte getTagDelay()	get the tag delay time (in ms) before next tag coming.
boolean setTagDelay(byte tagDelay)	set the tag delay before next tag coming in unit of ms.
byte getSelectDelay()	Get Select delay register in Atmel controller
boolean setSelectDelay(byte tagDelay)	Set Select delay register in Atmel controller
List<String> getRxAttenList()	Get the enum list of RxAtten register in E710
List<String> getMixerList()	Get the enum list of Mixer register in E710
List<String> getPga1List()	Get the enum list of Pga1 register in E710
List<String> getPga2List()	Get the enum list of Pga2 register in E710
List<String> getPga3List()	Get the enum list of Pga3 register in E710
short getRxAttenGain()	Get the value of RxAtten register in E710
int getMixerGain()	Get the value of Mixer register in

Copyright Convergence Systems Limited 2022, All Rights Reserved

	E710
int getPga1Gain()	Get the value of Pga1 register in E710
int getPga2Gain()	Get the value of Pga2 register in E710
int getPga3Gain()	Get the value of Pga3 register in E710
boolean setRxGain(int rxAttenGain, int mixerGain, int pga1Gain, int pga2Gain, int pga3Gain)	Set the value of rxAtten, mixerGain, pga1, pga2, pga3 of E710
long getCycleDelay()	get the tag delay before running next inventory cycle in unit of ms
boolean setCycleDelay(long cycleDelay)	set the tag delay before running next inventory cycle in unit of ms
int getIntraPacketDelay()	Get intraPacket delay in Atmel controller
boolean setIntraPacketDelay(int intraPkgDelay)	Set intraPacket delay in Atmel controller
void getAuthenticateReplyLength()	get the length of the authentication reply data
boolean setTam1Configuration(int keyId, String matchData)	Set the Tam1 data for the tam1 authentication check
boolean setTam2Configuration(int keyId, String matchData, int profile, int offset, int blockId, int protMode)	Set the Tam2 data for the tam2 authentication check
int getUntraceableEpcLength()	get the length of the untraceable Epc
boolean setUntraceable(boolean bHideEpc, int ishowEpcSize, int iHideTid, boolean bHideUser, boolean bHideRange)	set the untraceable parameters
boolean setUntraceable(int range, boolean user, int tid, int epcLength, boolean epc, boolean uxpc)	set the untraceable parameters
Int getQValue()	get the start Q value set for dynamic algorithm.
int getMaxQ()	get the maximum Q value set for dynamic algorithm.
boolean setMaxQ(int maxQ)	Set max Q in the E710 register
int getMinQ()	get the minimum Q vale set for dynamic algorithm.
boolean setMinQ(int minQ)	Set min Q in the E710 register
byte getNumMinQcycles()	Get number of min Q cycles in E710 register
boolean setNumMinQCycles(int numMinQCycles)	Set number of min Q cycles in E710 register
byte getQincreaseUseQuery()	Get Q value increase use Query in E710 register
boolean setQincreaseUseQuery(boolean qIncreaseUseQuery)	Set Q value increase use Query in E710 register
getQdecreaseUseQuery()	Get Q value decrease use Query in

Copyright Convergence Systems Limited 2022, All Rights Reserved

	E710 register
setQdecreaseUseQuery(boolean qDecreaseUseQuery)	Set Q value decrease use Query in E710 register
long getMaxQueries()	Get Max queries in E710 register
boolean setMaxQueries(long maxQueries)	Set Max queries in E710 register
int getRetryCount()	get the retry count set for dynamic algorithm.
boolean setRetryCount(int retryCount)	Set the retry count set for dynamic algorithm.
public int getInvSelectIndex()	get the inventory select index
boolean getSelectEnable()	get the enable status of the inventory select index.
int getSelectTarget()	get the target selected for the inventory select index.
int getSelectAction()	get the action selected for the inventory select index.
int getSelectMaskBank()	get the mask bank for the inventory select index.
int getSelectMaskOffset()	get the mask offset for the inventory select index.
String getSelectMaskData()	get the mask data for the inventory select index.
boolean setInvSelectIndex(int invSelect)	set inventory select index for the inventory select index. Parameter: invSelect from 0 to 7.
boolean setSelectCriteriaDisable()	Disable the select of the select index for inventory
boolean setSelectCriteria(int index, boolean enable, int target, int action, int bank, int offset, String mask, boolean maskbit)	set the select criteria parameters: enable: the enable status of the inventory select index. target: the target selected for the inventory select index. action: the action selected for the inventory select index. bank: the mask bank for the inventory select index. offset: the mask offset for the inventory select index. mask: the mask data for the inventory select index. maskbit: indicate the mask string is binary or hexadecimal String.
boolean setSelectCriteria(boolean enable, int target, int action, int bank, int offset, String mask)	set the select criteria parameters:

Copyright Convergence Systems Limited 2022, All Rights Reserved

	enable: the enable status of the inventory select index. target: the target selected for the inventory select index. action: the action selected for the inventory select index. bank: the mask bank for the inventory select index. offset: the mask offset for the inventory select index. mask: the mask data for the inventory select index.
boolean setSelectCriteria(int index, boolean enable, int target, int action, int bank, int offset, String mask, int maskblen)	set the select criteria parameters: enable: the enable status of the inventory select index. target: the target selected for the inventory select index. action: the action selected for the inventory select index. bank: the mask bank for the inventory select index. offset: the mask offset for the inventory select index. mask: the mask data for the inventory select index. Maskblen: indicate valid bit of the mask string.
boolean getRssiFilterEnable()	Check if RSSI filtering is enabled or not
int getRssiFilterType()	Check the Filter type
int getRssiFilterOption()	Check the Filter Option
boolean setRssiFilterConfig(boolean enable, int rssiFilterType, int rssiFilterOption)	Set RSSI filtering parameters
double getRssiFilterThreshold1()	Get RSSI filter threshold 1
getRssiFilterThreshold2()	Get RSSI filter threshold 2
boolean setRssiFilterThreshold(double rssiFilterThreshold1, double rssiFilterThreshold2)	Set RSSI filter thresholds
long getRssiFilterCount()	Check RSSI filter count
setRssiFilterCount(long rssiFilterCount)	Set RSSI filter count
boolean getInvMatchEnable()	get the enable status set for post-filter
boolean getInvMatchType()	get the match type set for the post-filter.
int getInvMatchOffset()	get the match offset for the post-filter.
String getInvMatchData()	get the mask data for the post-filter.
boolean setPostMatchCriteria(boolean enable, boolean target,	set the post filter match criteria

Copyright Convergence Systems Limited 2022, All Rights Reserved

int offset, String mask)	parameters: enable: the enable status set for post-filter target: the match type set for the post-filter.. True to filter matched ones. offset: the match offset for the post-filter. mask: the mask data for the post-filter
int getCountryNumberInList()	get logical channel number used for fixed channel usage.
String[] getCountryList()	Get the list of country selectable for the radio device.
boolean setCountryInList(int countryInList)	set the country in the selectable list for the radio device
boolean getChannelHoppingStatus ()	get frequency hopping order. 0 for hopping, 1 for fixed frequency.
boolean setChannelHoppingStatus(boolean channelOrderHopping)	Set frequency hopping order. 0 for hopping, 1 for fixed frequency.
int getChannel()	get the channel used
boolean setChannel(int channelSelect)	select the channel to be used
int FreqChnCnt()	Get the number of the frequency used
double getLogicalChannel2PhysicalFreq(int channel)	get actual physical channel number from logical channel number
int getDuplicateEliminationTime()	Get duplication elimination time
boolean setDuplicateEliminationTime(int duplicateEliminationTime)	Set duplication elimination time
boolean setEventPackageUplinkEnable(byte bEnable)	Enable event package uplink
boolean setMultibankReadConfig(int iSet, byte[] byteArrayData)	Set event packet uplink
boolean setInvModeCompact(boolean invModeCompact)	set inventory mode as compact mode if true, normal mode if false.
boolean setKillPassword(String password)	set the kill password
boolean setAccessPassword(String password)	set the access password
boolean setAccessRetry(boolean accessVerify, int accessRetry)	set the access retry parameters: accessVerify: need to verify or not after access accessRetry: number of retry if access failure
boolean setAccessLockAction(int accessLockAction, int accessLockMask)	set lock configuration for lock operation.
void restoreAfterTagSelect()	restore user configured parameter after selected operation

Copyright Convergence Systems Limited 2022, All Rights Reserved

boolean setSelectedTag(String strTagId, int selectBank, long pwrlevel)	set the selected tag to be selected or matched for the RFID operation. Parameter: strTagId: the Epc of the matching tag. selectBank: the bank of the matching tag. Pwrlevel: power level
boolean setSelectedTag(String selectMask, int selectBank, int selectOffset, long pwrlevel, int qValue, int matchRep)	set the selected tag to be selected or matched for the RFID operation. Parameter: selectMask: the mask of the matching tag. selectBank: the bank of the matching tag selectOffset: the mask offset of matching tag. Pwrlevel: power level qValue: the qValue matchRep: the match repeat count
boolean startRfidInventory()	start RFID inventory.
boolean startRfidSearching()	Start RFID searching
Library4A.MultiBankData startRfidInventoryWithMultiBank(Library4A.MultiBankData multiBankData, String mDid, boolean bMultiBank, boolean needTagFiltering)	Start RFID multibank inventory
boolean abortRfidOperation()	stop the operation.
void resetInvalidata()	Reset invalidate
getInvalidata()	Get invalidate
int getInvalidUpdata()	Get invalidUpdata
int getValidata()	Get validate
long getTagRate()	Get tagRate
long getcrcErrorTagRate()	Get crcErrorTagRate
boolean sendHostRegRequestHST_CMD(HostCommands hostCommand)	set operation command to be executed in StartOperation command
Library4A.UplinkPacket readCsReaderUplinkPacketDecoded()	Get new RFID inventoried tag information with output: null for nothing. Otherwise it returns the package data inventoried.
UplinkPacket readUplinkPacket()	This is packet without decoding
ConnectorRfidReader.TagData decodeXerxesTagData(UplinkPacket uplinkPacket, String strMdid, Library4A.MultiBankData multiBankData)	Decode the uplink RFID data

Copyright Convergence Systems Limited 2022, All Rights Reserved

void setRfidReaderDefault()	Set the default parameters in Settings
void settagTypeExpected(TagTypes tagType)	Set the expected Tag Type for the inventory
TagTypes gettagTypeExpected()	Get the expected Tag type set in the library
void setselectFor(int selectFor)	Set the expected operation for the expected tag type for the inventory
int getselectFor()	Get the expected operation for the expected tag type for the inventory
void setmDid(String mDid)	Set no expected tag type for the inventory
void startSettingTask(Context context, boolean sameSetting, String invalidRequest)	Start the Setting task
boolean isSettingTaskRunning()	Check if the setting task is finished or not
void stopSettingTask()	Stop the setting task
void resetSelectData()	Reset all tag select registers
void setselectHold(int selectHold)	Set the select hold time
boolean disableMultibankReadConfig()	Disable multibank registers
set_em4245_commandConfig()	Set em4245 register value
boolean setKillPassword(String password)	Set the killPassword register in E710 Atmel controller
boolean setAccessPassword(String password)	Set AccessPassword register in E710 Atmel controller
setAccessRead(int accessBank, int accessOffset, int accessCount)	Set AccessRead register in E710 Atmel controller
boolean setAccessWrite(int accessBank, int accessOffset, int accessCount, String dataInput)	Set AccessWrite Register in E710 Atmel controller

Copyright Convergence Systems Limited 2022, All Rights Reserved

Chapter 5: Example Command Sequences

5.1 Introduction

This chapter gives some example sequences of commands of common operations on CS710S

1. Scan and Connect Bluetooth
2. Start RFID Tag/Label Inventory (Simplest – using default configuration)
3. Start RFID Tag/Label Inventory (with non-default Configuration)
4. Start Barcode Inventory
5. Search an RFID Tag/Label (Geiger Search)
6. Read an RFID Tag/Label TID Bank
7. Write an RFID Tag/Label EPC Bank

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.2 Scan and Connect Bluetooth

1. scanLeDevice(true): start scanning
2. scanLeDevice(false): stop scanning
3. isScanning(): check if scanning
4. getNewDeviceScanned(): to get possible scanned device data
5. connect(readerDevice): connect with one of the readerDevice scanner
6. disconnect(false): disconnect the bluetooth connection.
7. isConnected(): check bluetooth connection

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.3 Start RFID Tag/Label Inventory using Default Configuration

1. startRfidInventory(): start inventory
2. readCsReaderUplinkPacketDecoded(): to get possible inventoried package data
3. abortRfidOperation(): to stop inventory (do this when you are done reading RFID tags)

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.4 Start RFID Tag/Label Inventory using Non-Default Configuration

1. abortRfidOperation(): to stop any previous RFID inventory and check RFID module health
2. setAntennaPower(300): Set Power = 300
3. setAntennaDwell(0): Set Antenna Dwell=0
4. setCurrentLinkProfile (1): Set Link Profile = 1
5. setTagGroup(0, 0, 0): Set select = all, Session = S0, target = A
6. setInvAlgo(true): Set Algorithm true for Dynamic, false for Fixed
7. setQValue(8), setMinQ(0), setMaxQ(15), setInvModeCompact(false): set Q, MinQ, MaxQ for the algorithm, set non-Compact mode inventory
8. startRfidInventory(): start inventory
9. readCsReaderUplinkPacketDecoded(): to get possible inventoried package data
10. abortRfidOperation(): to stop RFID inventory (do this when you are done reading RFID tags)

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.5 Start Barcode Inventory

1. setBarcodeOn(true): start barcode
2. barcodeInventory(true): start inventory
3. readCsReaderUplinkPacketDecoded(): get possible inventoried barcode data
4. barcodeInventory(false): stop inventory
5. setBarcodeOn(false): stop barcode (do this when you are done reading barcodes)

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.6 Search an RFID Tag/Label (Geiger Search)

1. abortRfidOperation(): to stop any previous RFID inventory and check RFID module health
2. setSelectedTag("2017000000000000000000001", 1, 300): set target tag ID, such as 2017000000000000000000001, bank selected = 1, and power = 300
3. setCurrentLinkProfile (103): Set Link Profile = 103
4. setTagGroup(0, 1, 2): Set select = all, Session = S1, target = A/B toggle
5. setInvAlgo(0): Set Algorithm = Fixed
6. setQValue(0), setInvModeCompact(false): Q=0 for fixed algorithm, compact mode as false
7. startRfidInventory(): start inventory
8. readCsReaderUplinkPacketDecoded(): to get possible inventoried package data
9. abortRfidOperation(): to stop RFID inventory (do this when you are done reading RFID tags)

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.7 Read an RFID Tag/Label TID Bank

1. abortRfidOperation(): to stop any previous RFID inventory and check RFID module health
2. setAccessRead(2, 0, 1): to access read TID bank, offset as 0, count as 1
3. setAccessPassword("0000"): to set the access password. "0000" means no password.
4. setAccessRetry(true, 7): to enable retry and set retry count as 7
5. setSelectedTag("20170000000000000000001", 1, 300): set target tag ID, such as 20170000000000000000001, bank selected as 1, and power = 300
6. setCurrentLinkProfile(1): Set Link Profile = 1
7. setTagGroup(0, 1, 2): Set select = all, Session = S1, target = A/B toggle
8. setInvAlgo(0): Set Algorithm = Fixed
9. setQValue(0), setInvModeCompact(false): Q=0 for fixed algorithm, compact mode as false
10. sendHostRegRequestHST_CMD(CMD_18K6CREAD): to start read operation.
11. readCsReaderUplinkPacketDecoded(): to get possible command response package data. The data to be read will come back in this Event.
12. abortRfidOperation(): to stop RFID inventory (do this when you are done reading RFID tags)

Copyright Convergence Systems Limited 2022, All Rights Reserved

5.8 Write an RFID Tag EPC Bank

1. abortRfidOperation(): to stop any previous RFID inventory and check RFID module health
2. setAccessWrite(1, 2, 1, “2018”): to access read EPC bank, offset as 2, count as 1, data as ‘2018’
3. setAccessPassword(“0000”): to set the access password. “0000” means no password.
4. setAccessRetry(true, 7): to enable retry and set retry count as 7
5. setSelectedTag(“20170000000000000000001”, 1, 300): set target tag ID, such as 20170000000000000000001, bank=1, and power = 300
6. setCurrentLinkProfile (1): Set Link Profile = 1
7. setTagGroup(0, 1, 2): Set select = all, Session = S1, target = A/B toggle
8. setInvAlgo(0): Set Algorithm = Fixed
9. setQValue(0), setInvModeCompact(false): Q=0 for fixed algorithm, compact mode as false
10. sendHostRegRequestHST_CMD(CMD_18K6CWRITE): to start write operation.
11. readCsReaderUplinkPacketDecoded(): to get possible command response package data. decodedResult will come back if successful. decodedError will come back if failed.
12. abortRfidOperation(): to stop RFID inventory (do this when you are done reading RFID tags)

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix A: Reader Modes (Link Profiles)

There are 11 link profiles in CS710S. Only 1 profile is active at any time in CS710S. The purpose of each link profile is explained below. These purposes correspond to different business and physical scenarios. The user should try out each profile to see which one gives best performance.

Mode ID	Mode Optimization	Intended Region	Forward Link Modulation	Tari (µs)	BLF (kHz)	Reverse Link Modulation	Chip RX Sensitivity (dBm)	Maximum Read Rate (tags/s)
103	Read Rate	FCC	DSB	6.25	640	Miller M=1	**-78**	**1000**
120	Read Rate	FCC	DSB	6.25	640	Miller M=2	**-81**	**700**
345	Read Rate	FCC, ETSI UB	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
302	Read Rate	All	PR_ASK	7.5	640	Miller M=1	**-78**	**800**
323	Read Rate	All	PR_ASK	7.5	640	Miller M=2	**-81**	**550**
344	ETSI UB DRM	All	PR_ASK	7.5	640	Miller M=4	**-84**	**400**
223	ETSI LB	ETSI	PR_ASK	15	320	Miller M=2	**-84**	**300**
222	ETSI LB	ETSI	PR_ASK	20	320	Miller M=2	**-84**	**250**
241	ETSI LB DRM	ETSI	PR_ASK	20	320	Miller M=4	**-87**	**200**
244	FCC DRM	FCC	PR_ASK	20	250	Miller M=4	**-88**	**150**
285	Sensitivity	All	PR_ASK	20	160	Miller M=8	**-93**	**50**

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix B: Sessions

Session is a concept of EPC to allow a tag to respond to multiple readers inventorying it at the same time, each using a different session number.

There are 4 possible sessions: S0, S1, S2, S3.

The user however has to be careful because these 4 sessions have different behavior, notably how the tag flag "persist" in time. A tag, before inventory or when just after power on, has a flag of State A. When it is inventoried, the flag will go to State B. The tag flag will stay in State B until the tag powers off or the persistence time is up.

A reader can declare it only wants to inventory flag A, so that after a tag is inventoried and its flag gone to State B, it will no longer respond to further inventory rounds – until the end of the persistence time.

Now for S0, S1, S2 and S3, the persistence times are DIFFERENT! Because of that, one has to be very careful in choosing which session to use.

Session	**Tag Flags Persistence Time**
S0	Tag Energized: indefinite Tag Not Energized: none
S1	Tag Energized: 0.5 second < Persistence Time < 5 seconds Tag Not Energized: 0.5 second < Persistence Time < 5 seconds

Copyright Convergence Systems Limited 2022, All Rights Reserved

S2	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time
S3	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix C: Tag Population and Q

Tag Population is the RFID tag population that is to be inventoried. To be more precise, it is the population of tags that can be "seen" by the RFID reader.

Q is an EPC concept related to the way a group of tags is inventoried. When a reader broadcast its desire to inventory tags, it sends out a Q value. The tag will, based on that Q, calculate a certain number and define that as the number of repeated inventories the reader will do. Basically, the relationship of Inventory Repeats and Q is:

Inventory Repeats = 2^Q

The tag will then choose by random a certain number less than this Inventory Repeats. When the reader starts doing inventory, the tag will then respond at that repeat number.

In other words, the Inventory Repeats should correspond to Tag Population:

Tag Population = Inventory Repeats = 2^Q

For example, if there are 8 tags, then in theory the Q can be 3, and if each tag chooses a number different from that of the other 7 (miraculously, of course), then the 8 tags will be inventoried in an orderly manner in turn.

Of course this will never happen, as the tags will easily choose a number the same as that of another one, and a collision will happen.

Therefore, it is a normal practice to have a bigger Q, such as 4 in this case, so that the 8 tags would have a lower chance of choosing the same number.

Copyright Convergence Systems Limited 2022, All Rights Reserved

Therefore, reversing the equation, ideally, we can have:

$Q = INTEGER(LOG_2(Tag\ Population))$

But in reality, we need some headroom, so that:

$Q = INTEGER(LOG_2(Tag\ Population \times 2)+1$

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix D: Query Algorithm

There are 2 types of Query Algorithm: Fixed Q and Dynamic Q.

For Fixed Q, the Q value does not change. In other words, the expected Tag Population does not change.

For Dynamic Q, the Q value changes adaptively: when there are a lot of inventory repeats where no tags respond, the reader will interpret that there are not that many RFID tags in the front, and hence it is more efficient to change the Q to a smaller value. When there are a lot of inventory repeats where the reader receive data but they do not satisfy checksum, meaning there is heavy collision, then the reader will interpret that there are too many RFID tags in the front of the reader, and hence it is better to increase the value of Q. Dynamic Q algorithm is a way to allow the RFID reader to adapt to different amount of RFID tags being seen by the reader. The idea is that if there are not so many tags, then the Q can be reduced and the reader can collect all the tag data faster.

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix E: Target

Target here actually refers to the target flag that the reader wants to inventory. There are 2 possible flags of an RFID tag: State A and State B.

When an RFID tag is first powered up, it has a flag of State A. After it is inventoried, the state of the flag becomes State B.

The tag will only go back to State A if either it is powered off and powered on again, or if its persistence time has run up (See Appendix B).

For each round of inventory, the reader sends out notification to the world which tag flag state it wants to inventory. It can keep on inventory State A, or it can inventory State A and State B alternatively from one round of inventory to the next round of inventory.

In theory, it is a good thing to inventory only State A. The reason being that those tags that have been inventoried should not respond again, and will hence quickly reduce the amount of collision between tags. So in general if you set inventory to State A only, the inventory of large amount of tags can be very fast.

The only catch is that when a tag responds to the reader, it does not know another tag is colliding with it. It sends out the response and thinks it has done the job, hence transitioning to flag State B. So in such case, the tag will not respond to further inventory, even though its response has been lost due to collision. Because of that, sometimes the user will set the inventory to target State A in one inventory round, and then State B in the next round, and vice versa, and so on. This is called A/B Toggle or A & B Dual Target or simply Dual Target.

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix F: Security

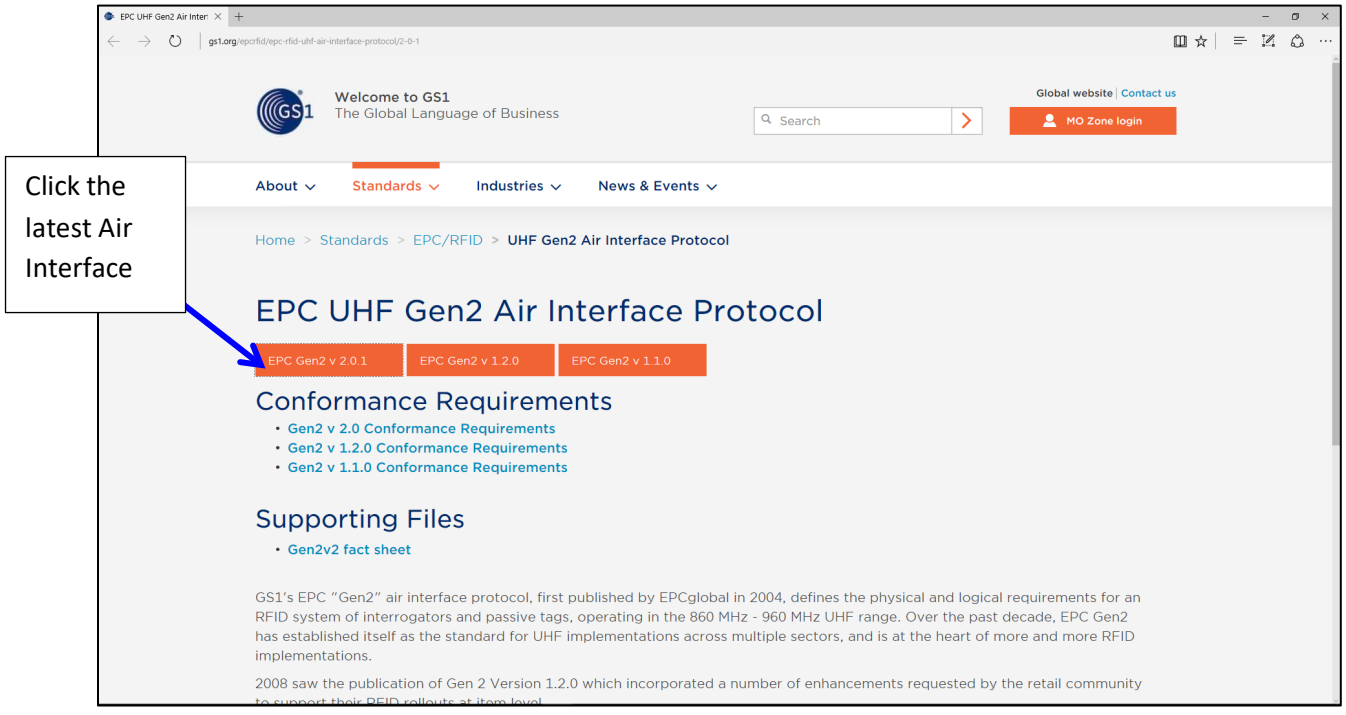
There are 4 actions you can apply on the memory inside an RFID tag:

1) Lock
2) Unlock
3) Permanent Lock
4) Permanent Unlock

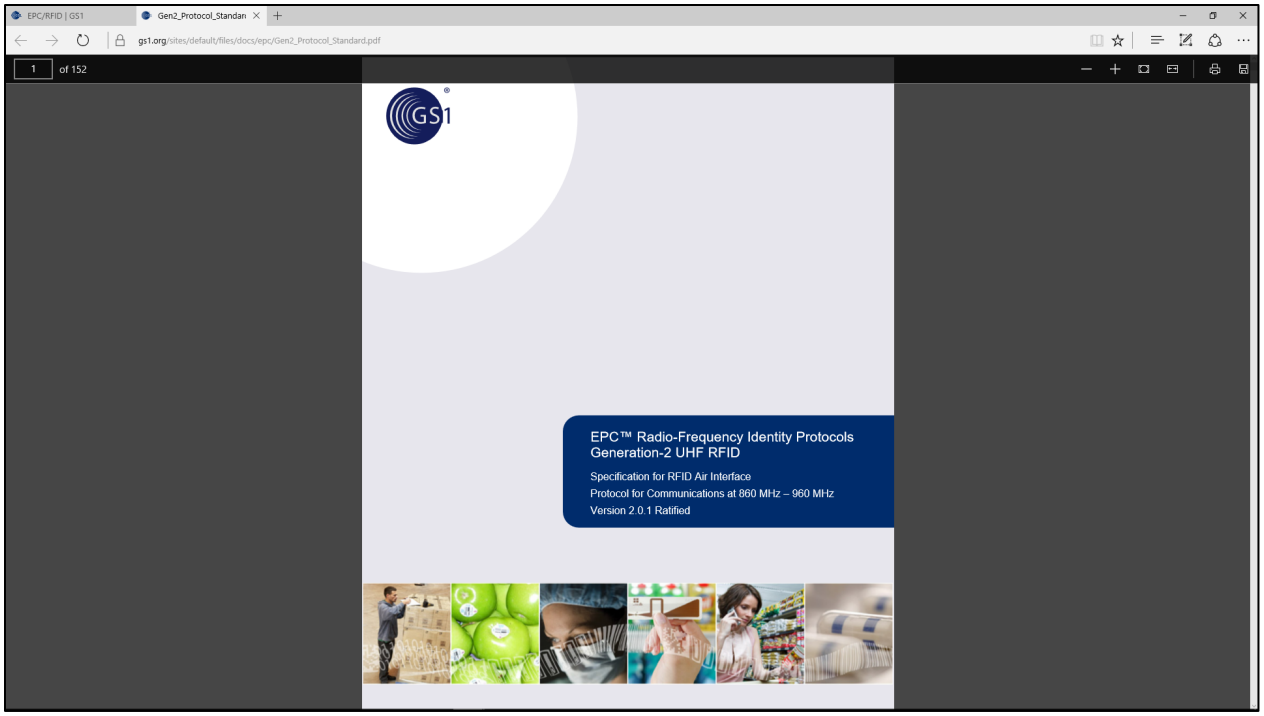
EPC document which can be found from EPC website:

https://www.gs1.org/epcrfid/epc-rfid-uhf-air-interface-protocol/2-0-1.

Once there, press the button showing the latest air interface and mouse click to get the pdf file.



Copyright Convergence Systems Limited 2022, All Rights Reserved



For Access Password and Kill Password the security locking affects both reading and writing.

For EPC bank and User bank, the security locking affects only writing.

For TID bank, since we are the user and not the manufacturing vendor, security action has no effect. It has been permanently unlocked in the factory anyway.

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix G: Models & Regulatory Region

There are various models, denoted by the alphanumeric right after the “CS710-“, here denoted by “**N**”. The applicable regulatory regions for each model are described to the right of it below:

N=1:	865-868 MHz for Europe ETSI, Russia, Mid-East countries,
	865-867 MHz for India
N=2:	902-928 MHz, FCC, for USA, Canada and Mexico. Hopping frequencies locked
N=2 AS:	920-926 MHz, Australia. Hopping frequencies locked
N=2 NZ:	921.5-928 MHz, New Zealand. Hopping frequencies locked
N=2 OFCA:	920-925 MHz, Hong Kong. Hopping frequencies locked
N=2 RW:	920-928 MHz, Rest of the World, e.g. Philippines, Brazil, Peru, Uruguay, etc.
N=4:	922-928 MHz, Taiwan
N=7:	920-925 MHz, China
N=8:	916.7-920.9 MHz, Japan
N=9:	915-921 MHz, Europe Upper Band

Copyright Convergence Systems Limited 2022, All Rights Reserved

Appendix H: Technical Support

All technical support should be sent to the following email:

info@convergence.com.hk

Copyright Convergence Systems Limited 2022, All Rights Reserved