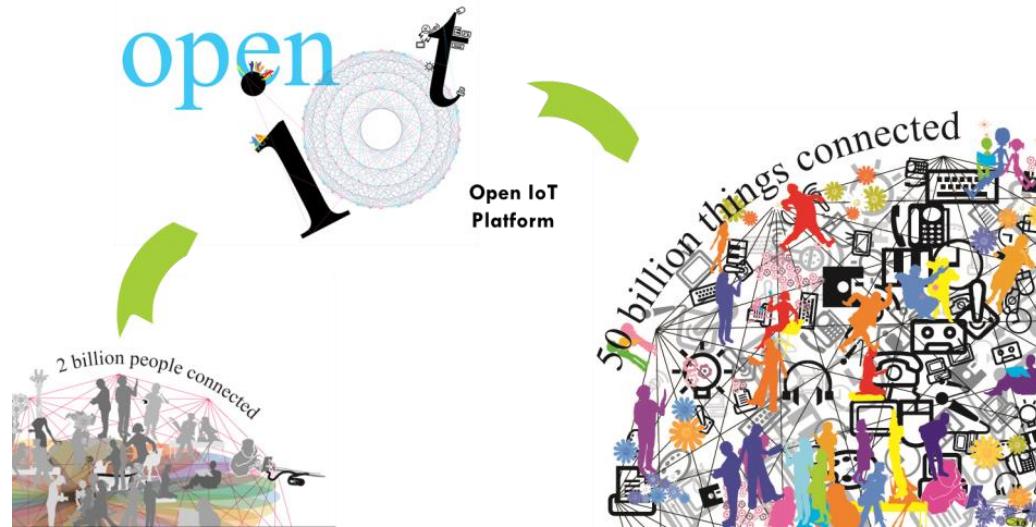


oneM2M Application AndroidSample UserGuide v0.6

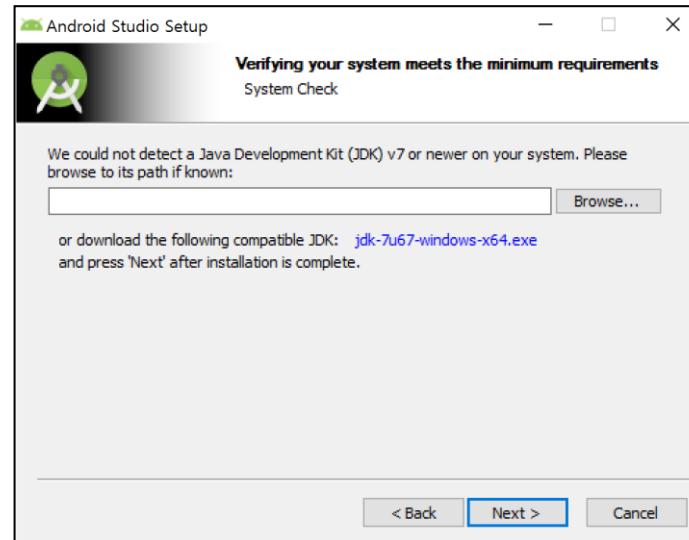
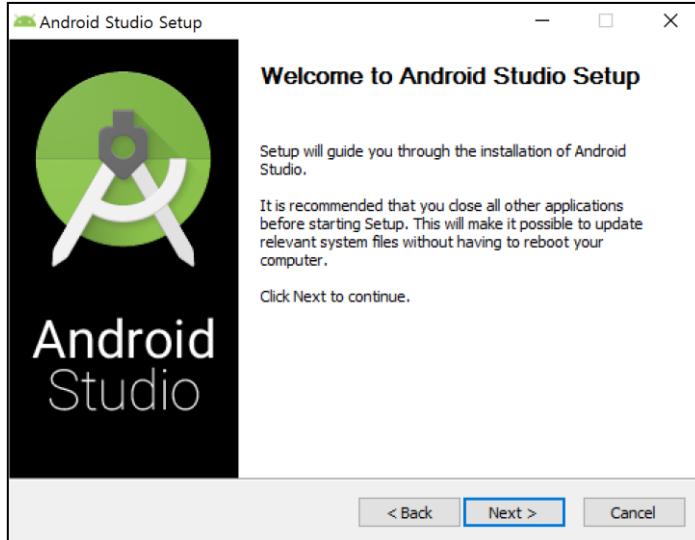


KETI (Korea Electronic Technology Institute)
(araha@keti.re.kr)

Android Studio Installation

■ Android Studio download and installation

<https://developer.android.com/sdk/index.html> (Android Studio Install ..., Java Development Kit 7u79 Install)



Java SE Development Kit 7u79

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement Decline License Agreement

Product / File Description	File Size	Download
Linux x86	130.4 MB	jdk-7u79-linux-i586.rpm
Linux x86	147.6 MB	jdk-7u79-linux-i586.tar.gz
Linux x64	131.69 MB	jdk-7u79-linux-x64.rpm
Linux x64	146.4 MB	jdk-7u79-linux-x64.tar.gz
Mac OS X x64	196.89 MB	jdk-7u79-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.79 MB	jdk-7u79-solaris-i586.tar.Z
Solaris x86	96.66 MB	jdk-7u79-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.67 MB	jdk-7u79-solaris-x64.tar.Z
Solaris x64	16.38 MB	jdk-7u79-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	140 MB	jdk-7u79-solaris-sparc.tar.Z
Solaris SPARC	99.4 MB	jdk-7u79-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	24 MB	jdk-7u79-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.4 MB	jdk-7u79-solaris-sparcv9.tar.gz
Windows x86	138.31 MB	jdk-7u79-windows-i586.exe
Windows x64	140.06 MB	jdk-7u79-windows-x64.exe

Java SE Development Kit 7u79

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

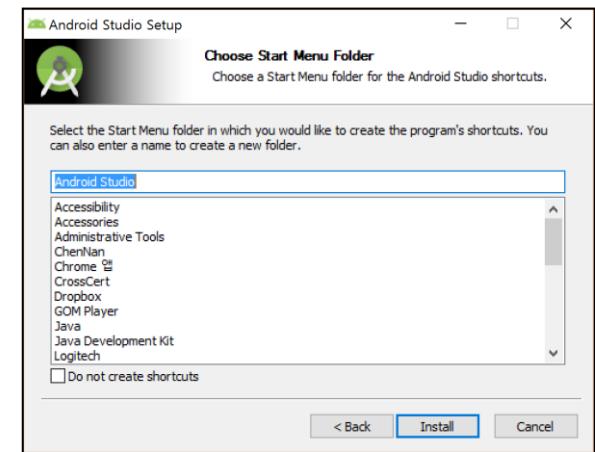
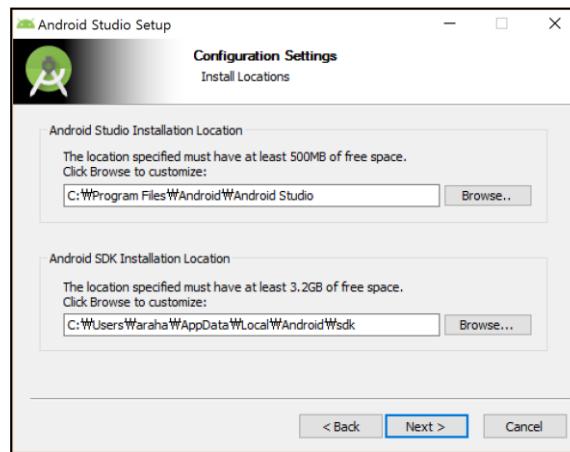
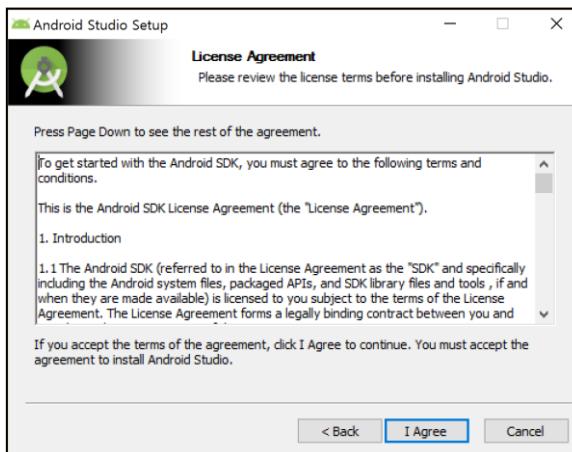
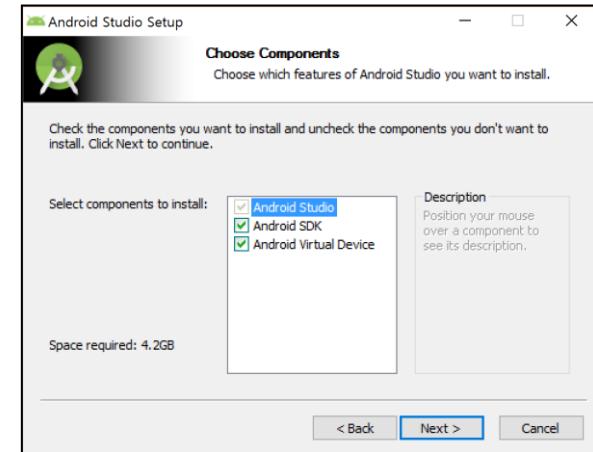
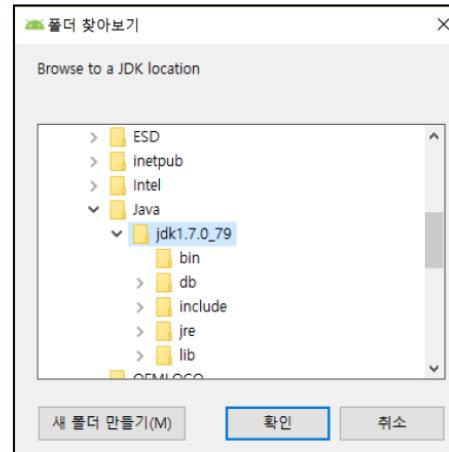
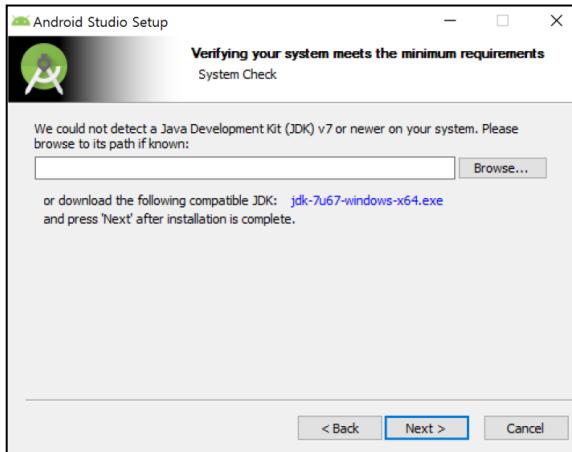
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

Product / File Description	File Size	Download
Linux x86	130.4 MB	jdk-7u79-linux-i586.rpm
Linux x86	147.6 MB	jdk-7u79-linux-i586.tar.gz
Linux x64	131.69 MB	jdk-7u79-linux-x64.rpm
Linux x64	146.4 MB	jdk-7u79-linux-x64.tar.gz
Mac OS X x64	196.89 MB	jdk-7u79-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.79 MB	jdk-7u79-solaris-i586.tar.Z
Solaris x86	96.66 MB	jdk-7u79-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.67 MB	jdk-7u79-solaris-x64.tar.Z
Solaris x64	16.38 MB	jdk-7u79-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	140 MB	jdk-7u79-solaris-sparc.tar.Z
Solaris SPARC	99.4 MB	jdk-7u79-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	24 MB	jdk-7u79-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.4 MB	jdk-7u79-solaris-sparcv9.tar.gz
Windows x86	138.31 MB	jdk-7u79-windows-i586.exe
Windows x64	140.06 MB	jdk-7u79-windows-x64.exe

Android Studio Installation

■ Android Studio download and installation

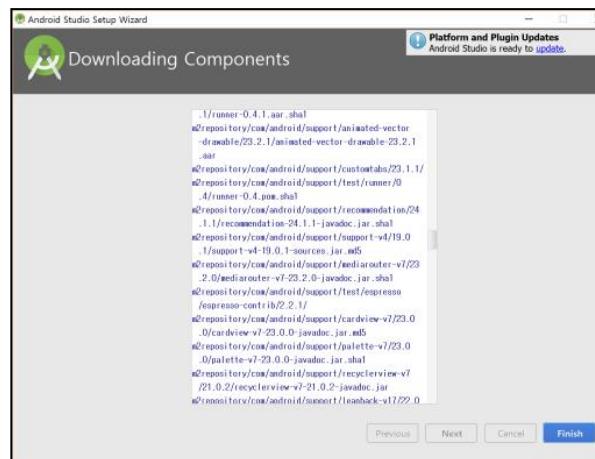
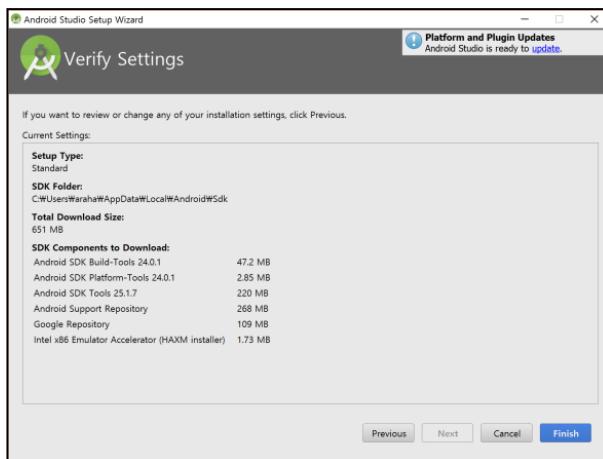
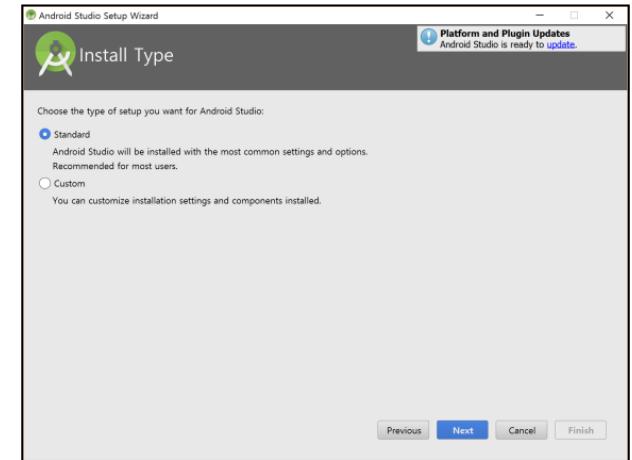
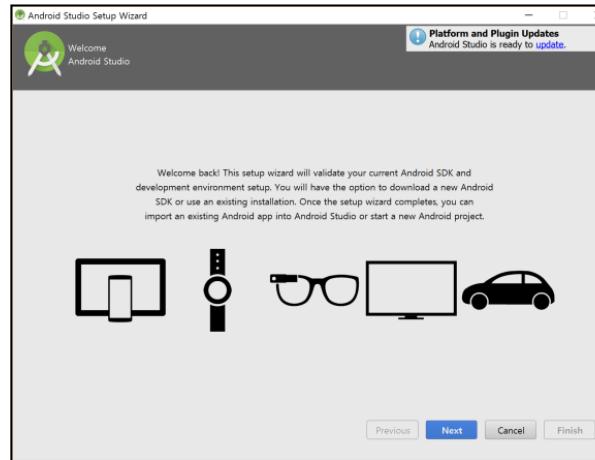
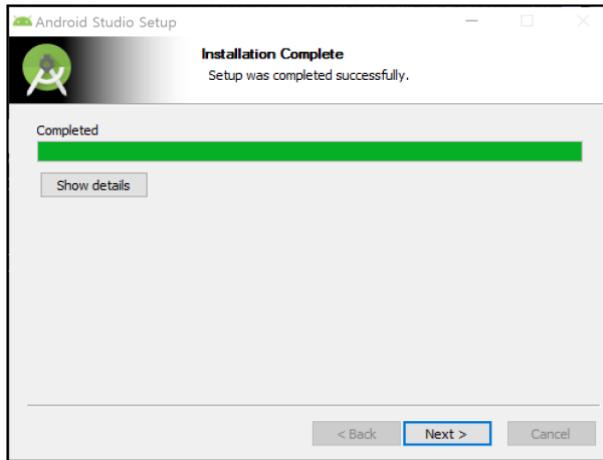
[https://developer.android.com/sdk/index.html \(Android Studio Install \)](https://developer.android.com/sdk/index.html)



Android Studio Installation

■ Android Studio download and installation

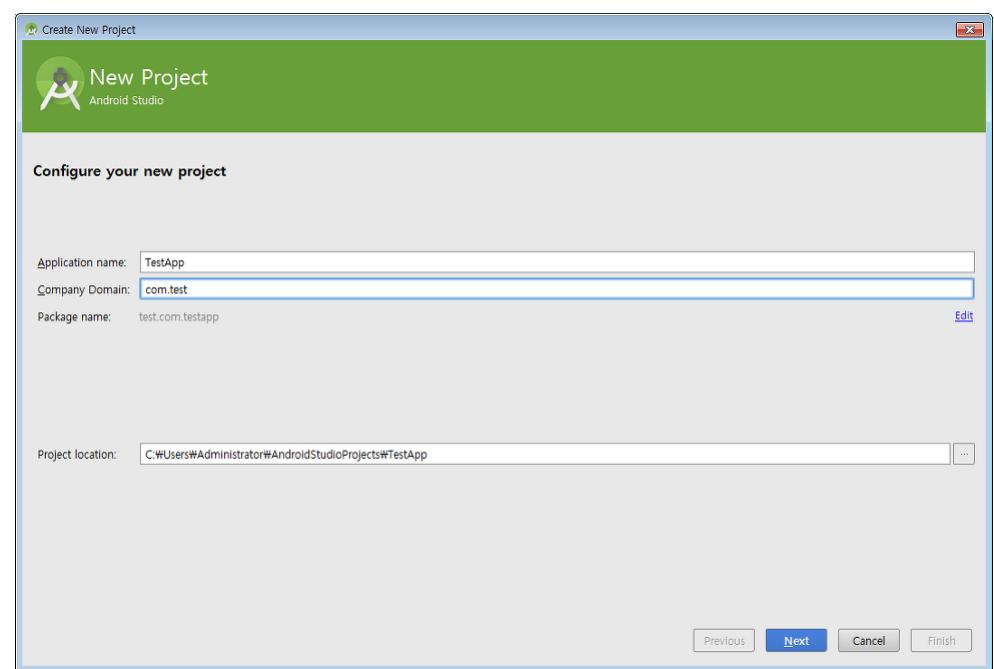
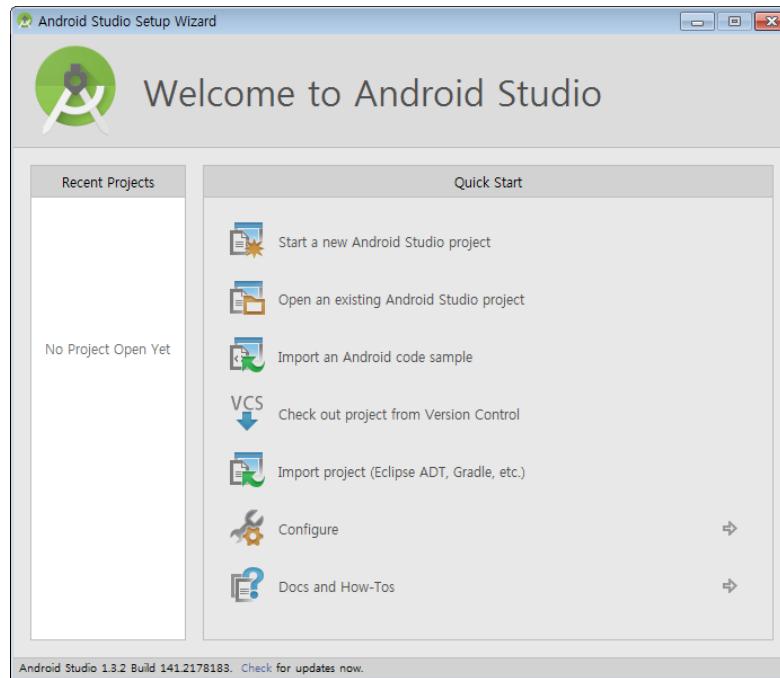
[https://developer.android.com/sdk/index.html \(Android Studio Install \)](https://developer.android.com/sdk/index.html)



Android Studio Installation

■ Execute Android Studio

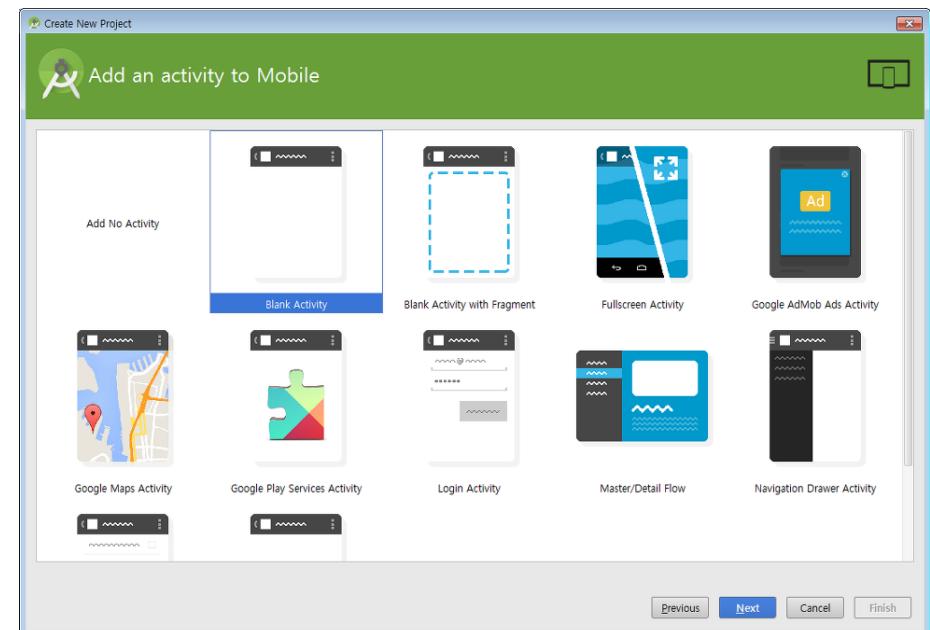
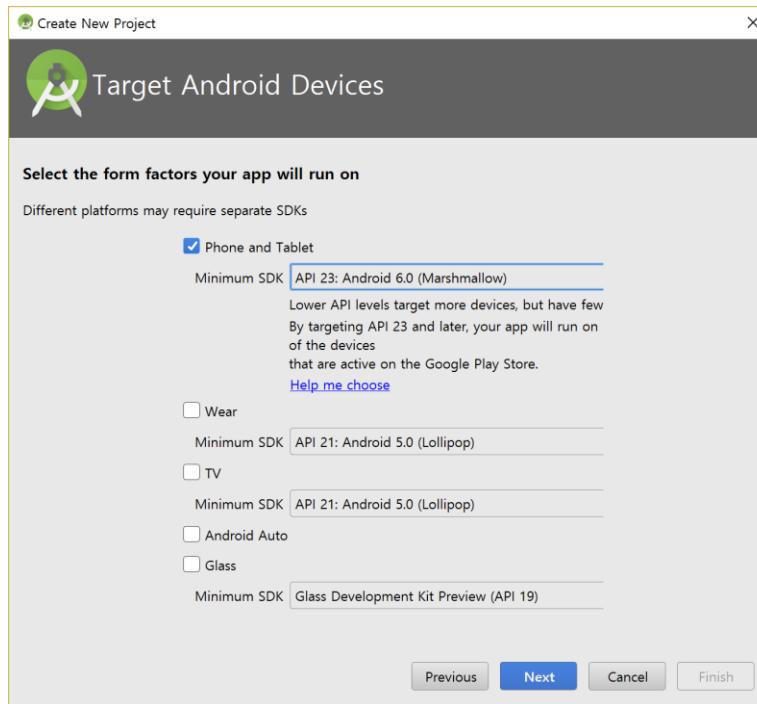
- Click “Start a new Android Studio project”.
- Type the application name and company domain



Android Studio Installation

■ Configure ADK version and activity for targeting android devices

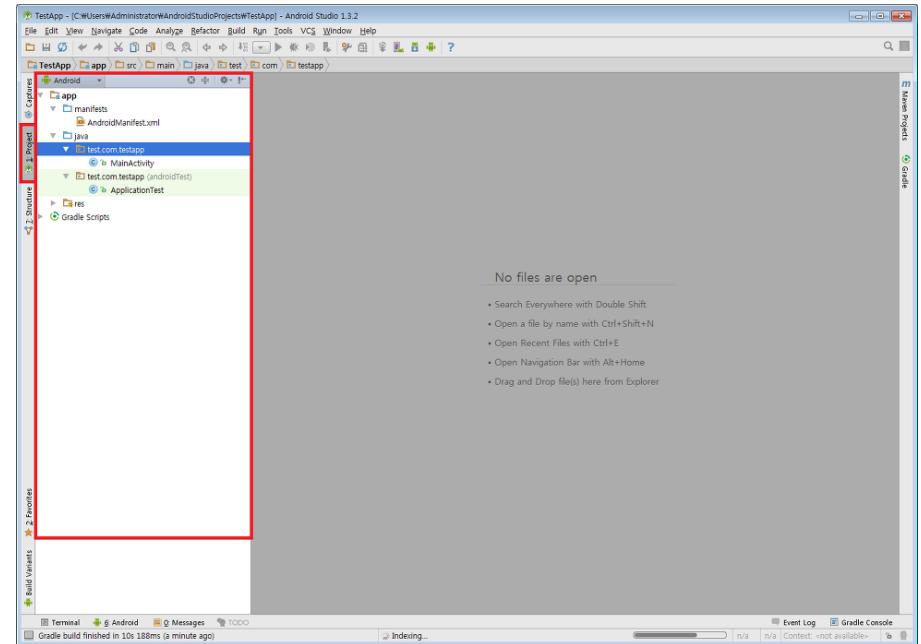
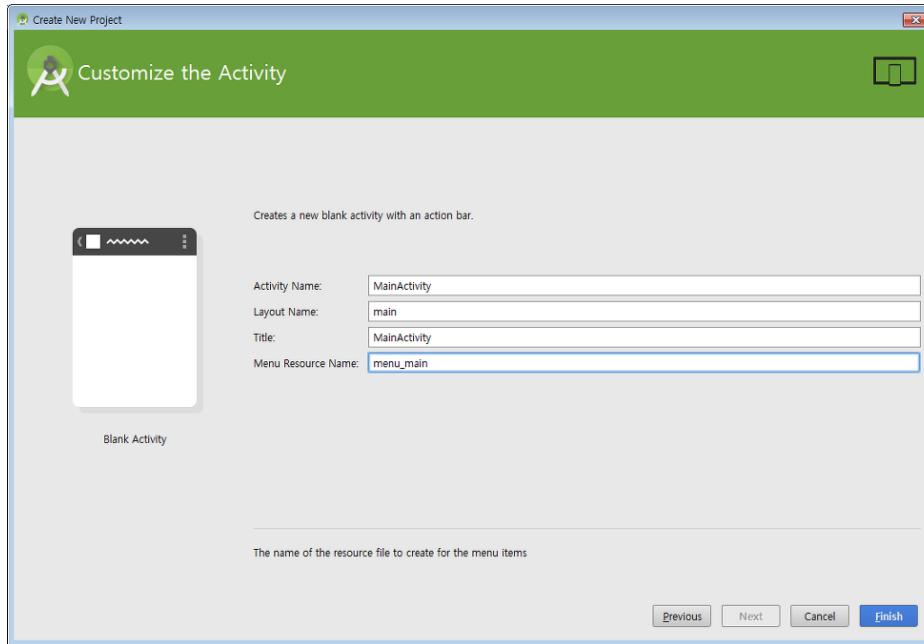
- Change Minimum SDK to API 23
- Select “Blank Activity”



Android Studio Installation

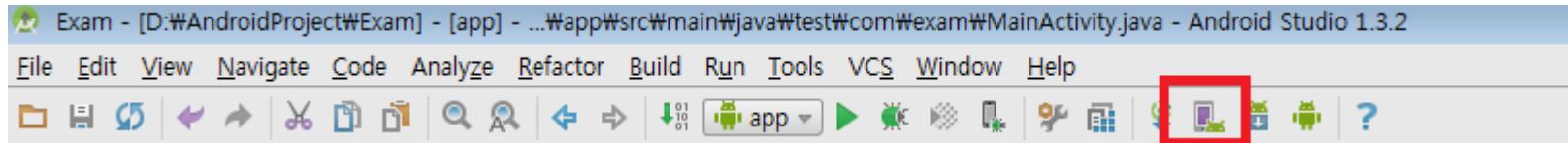
■ Customize the activity

- Input activity name and click “Finish” button.
- Right menu show the project file structure.



Android Studio Installation

■ Android Studio Virtual Devices



■ Click “AVD Manager” menu

1. The default virtual device is Nexus 5
2. [Add a new virtual device]
3. Click “Create Virtual Device”
4. Select a the phone type as VD

The image contains two side-by-side screenshots of the Android Virtual Device Manager.

Left Screenshot: Shows the main "Your Virtual Devices" screen. It has a table with columns: Type, Name, Resolution, API, Target, CPU/AB, Size on Disk, and Actions. There is one entry: "Nexus 5 API 23 x86" with resolution "1080 x 1920 xhdpi", API "23", Target "Google APIs", CPU/AB "x86", and Size on Disk "1.6G". A button at the bottom left says "+ Create Virtual Device".

Right Screenshot: Shows the "Virtual Device Configuration" dialog with the title "Select Hardware". It has a sidebar with categories: TV, Phone, Wear, and Tablet. Under "Phone", "Nexus One" is selected. The main table lists various devices with their details:

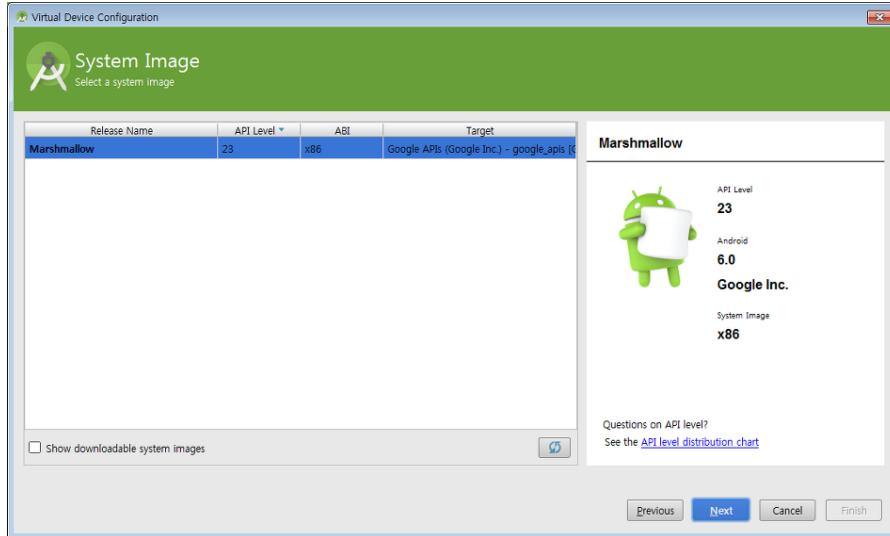
Name	Size	Resolution	Density
Nexus S	4.0"	480x800	hdpi
Nexus One	3.7"	480x800	hdpi
Nexus 6	5.96"	1440x2560	560dpi
Nexus 5	4.95"	1080x1920	xhdpi
Nexus 4	4.7"	768x1280	xhdpi
Galaxy Nexus	4.65"	720x1280	xhdpi
5.4" FWVGA	5.4"	480x854	mdpi
5.1" WVGA	5.1"	480x800	mdpi
4.7" WXGA	4.7"	720x1280	xhdpi

The "Nexus One" row is highlighted. On the right, there is a preview of the Nexus One device with dimensions 480px width and 800px height, labeled "Size: normal Ratio: long Density: hdpi". Buttons at the bottom right include "Clone Device...", "Previous", "Next", "Cancel", and "Finish".

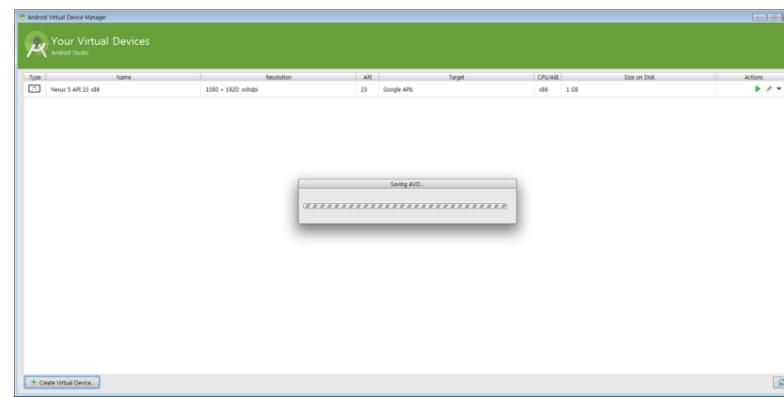
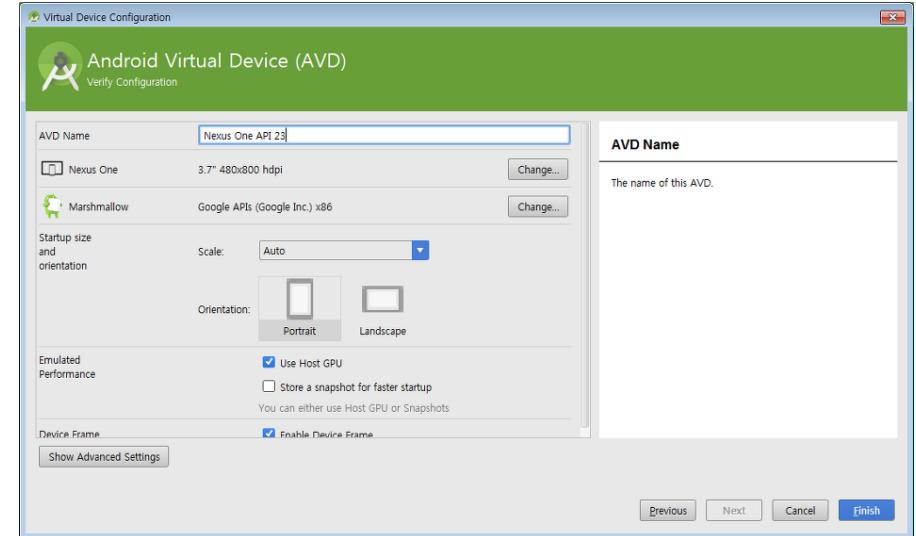
Android Studio Installation

■ Android Studio Virtual Devices

Select OS image



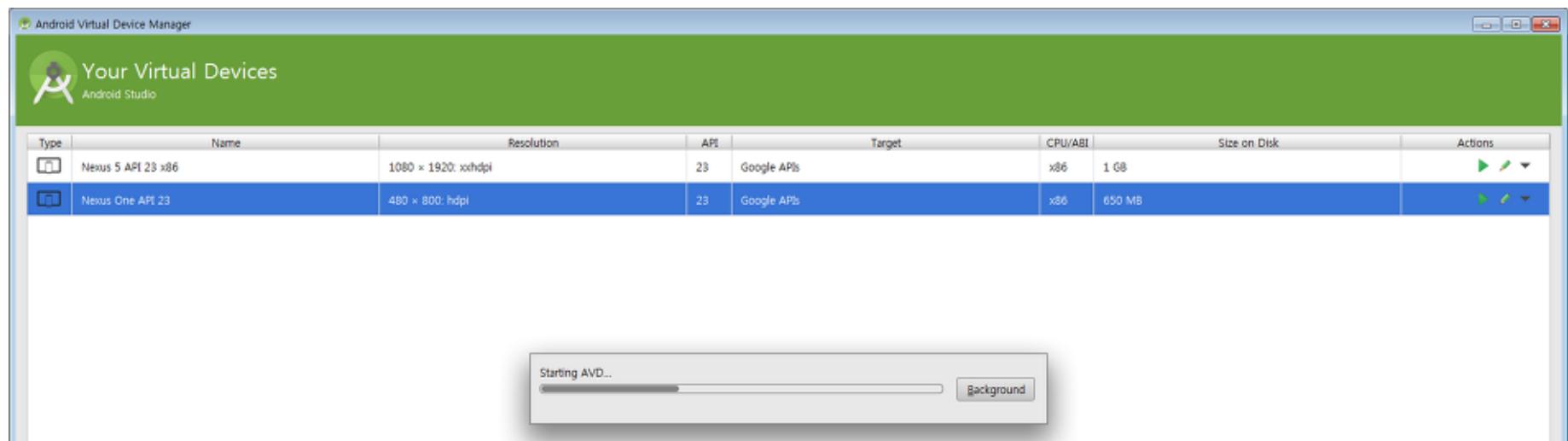
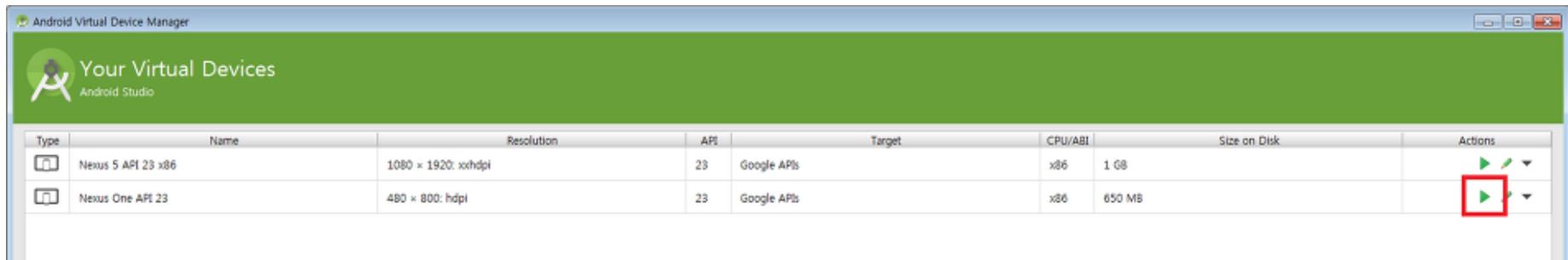
Android Virtual Device(AVD) Configuration



Android Studio Installation

■ Android Studio Virtual Devices

- Click start button to turn on the VD



Android Studio Installation

- Android Studio Virtual Devices
 - Wait for the VD system loading



Web View APP

■ Basic Default Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp" tools:context=".MainActivity">

    <TextView android:text="Hello World!" android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</RelativeLayout>
```

MainActivity.java

```
package com.example.araha.myapplication;

import ...;

public class MainActivity extends AppCompatActivity {

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

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.araha.myapplication" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:supportRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

AndroidManifest?

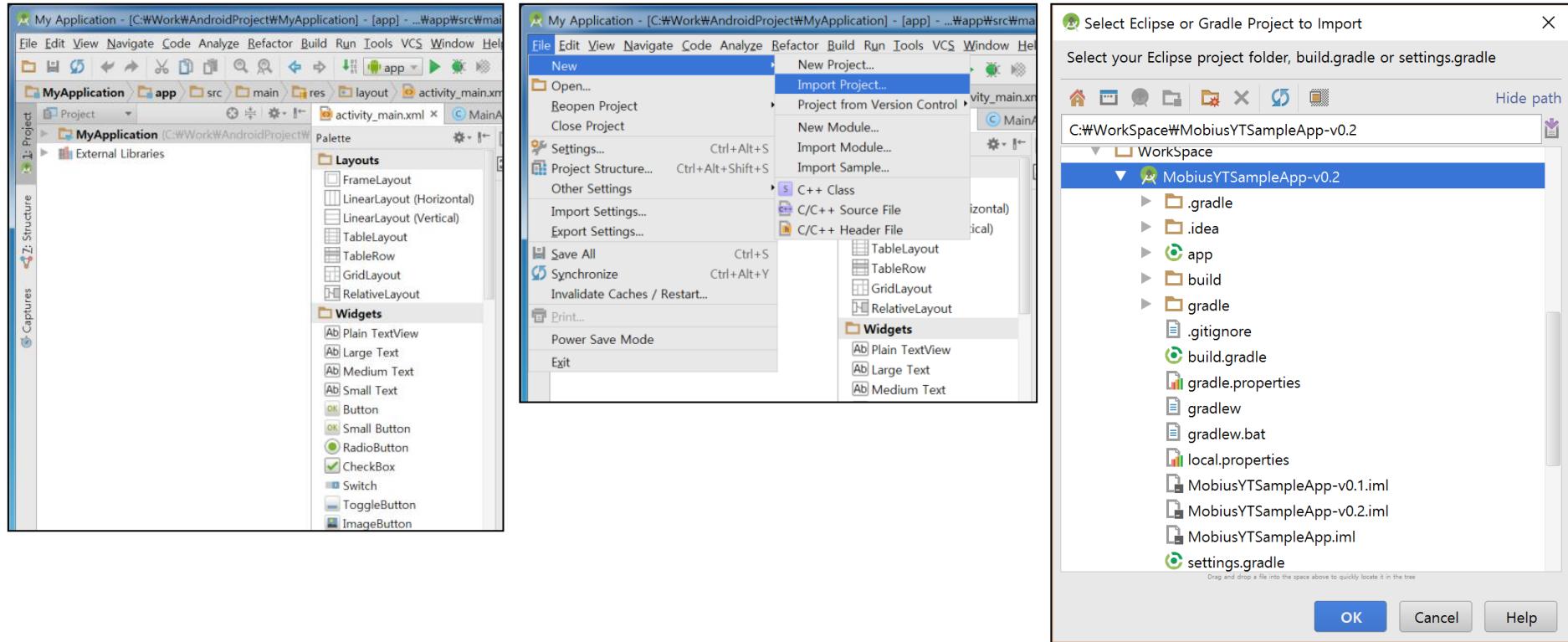
Descript and configure the android project

- App name, ID, icon, theme
- App access right
- Activity, Service, Intent

Mobius and APP connectivity

■ Import sample project

Android Project Import



* Download the sample source from IoT OCEAN (www.iotocean.org)

Mobius and APP connectivity

■ Import sample project

Android/app/manifest(AndroidManifest.xml)

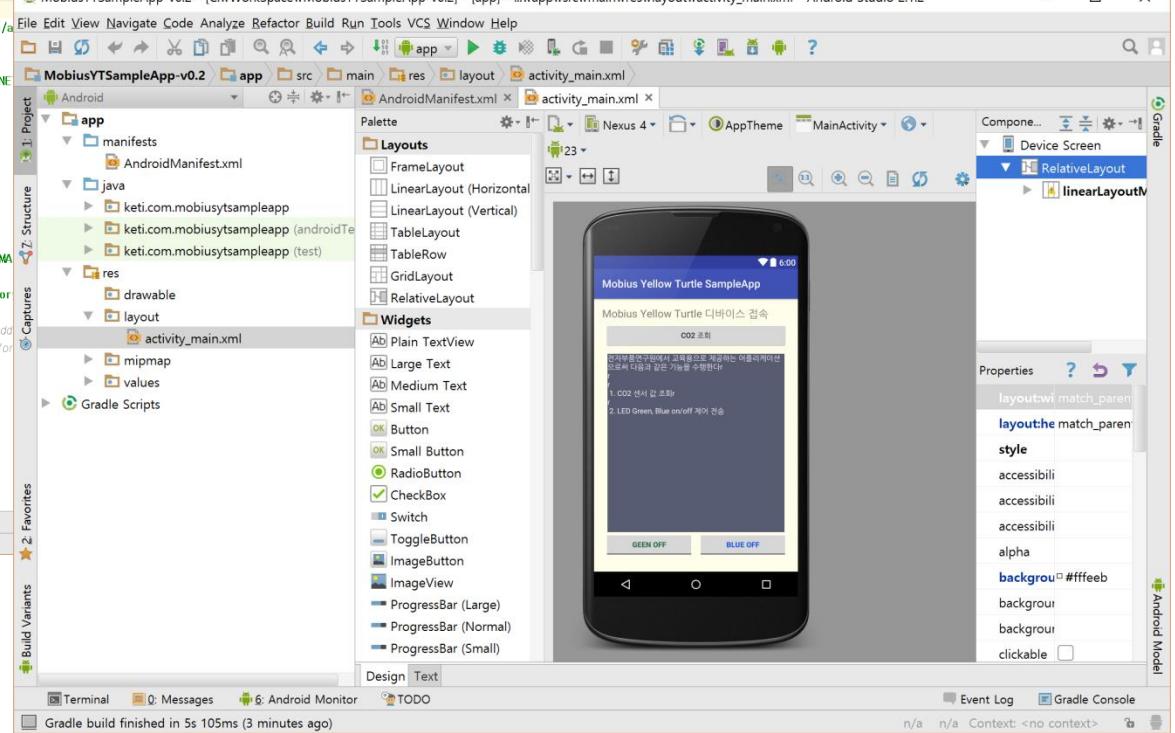
```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="keti.com.mobiusytsampleapp" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.BACKUP" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Mobius Yellow Turtle SampleApp"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity><!-- ATTENTION: This was auto-generated to add App Indexing. See https://g.co/AppIndexing/AndroidStudio for more information -->
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="9452000" />
    </application>
</manifest>

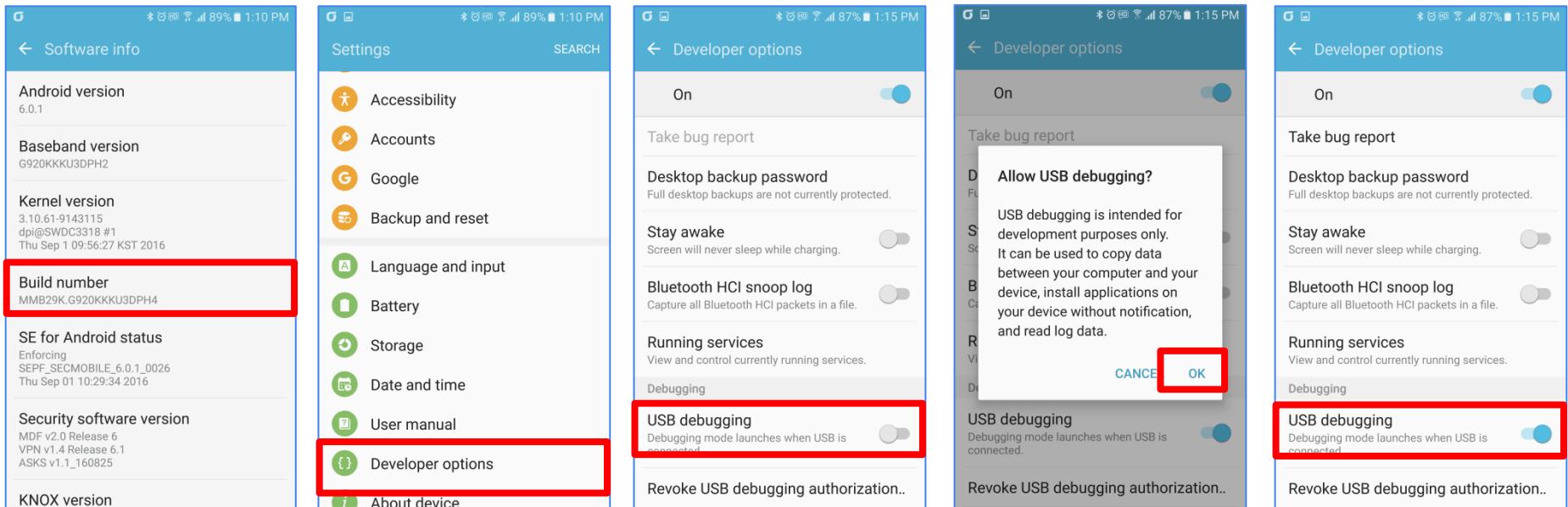
```



Mobius and APP connectivity

■ USB Debugging Mode on Android

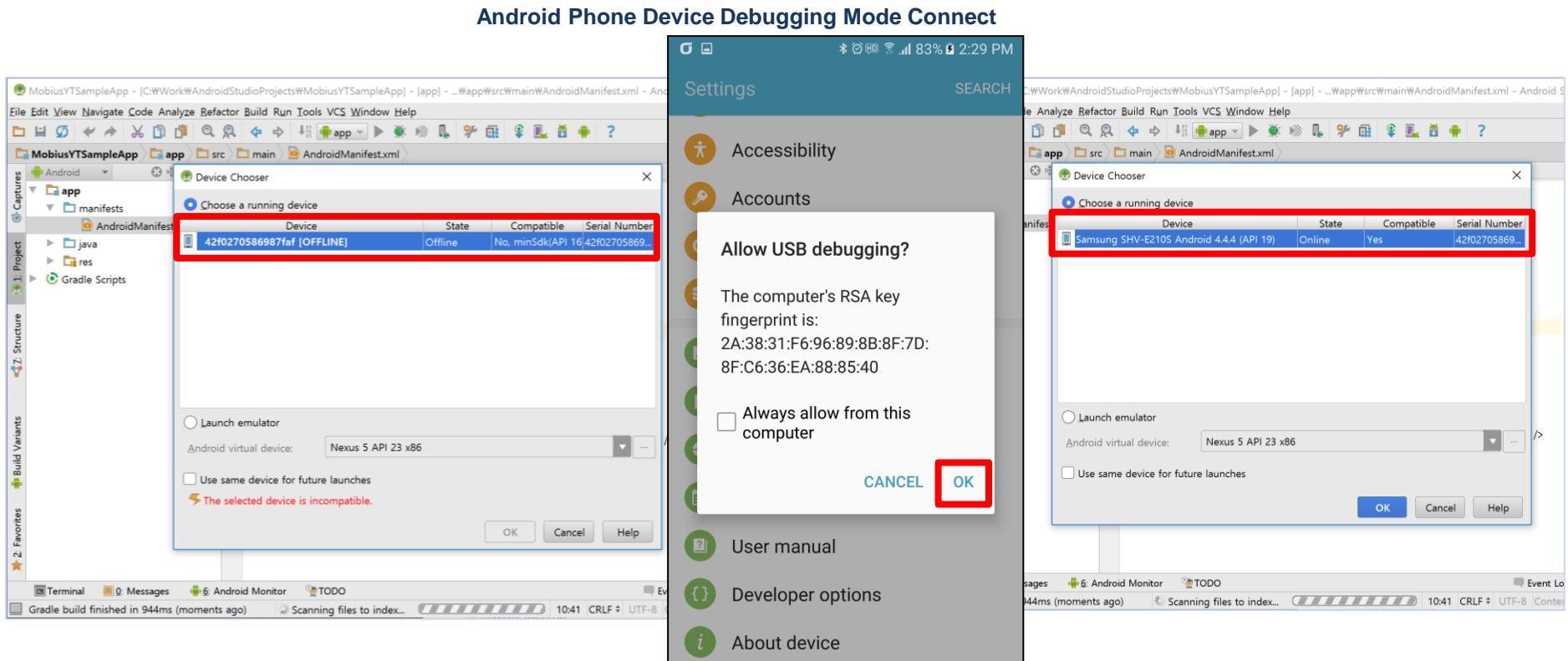
Change device to debugging mode



Click “Build number” continue

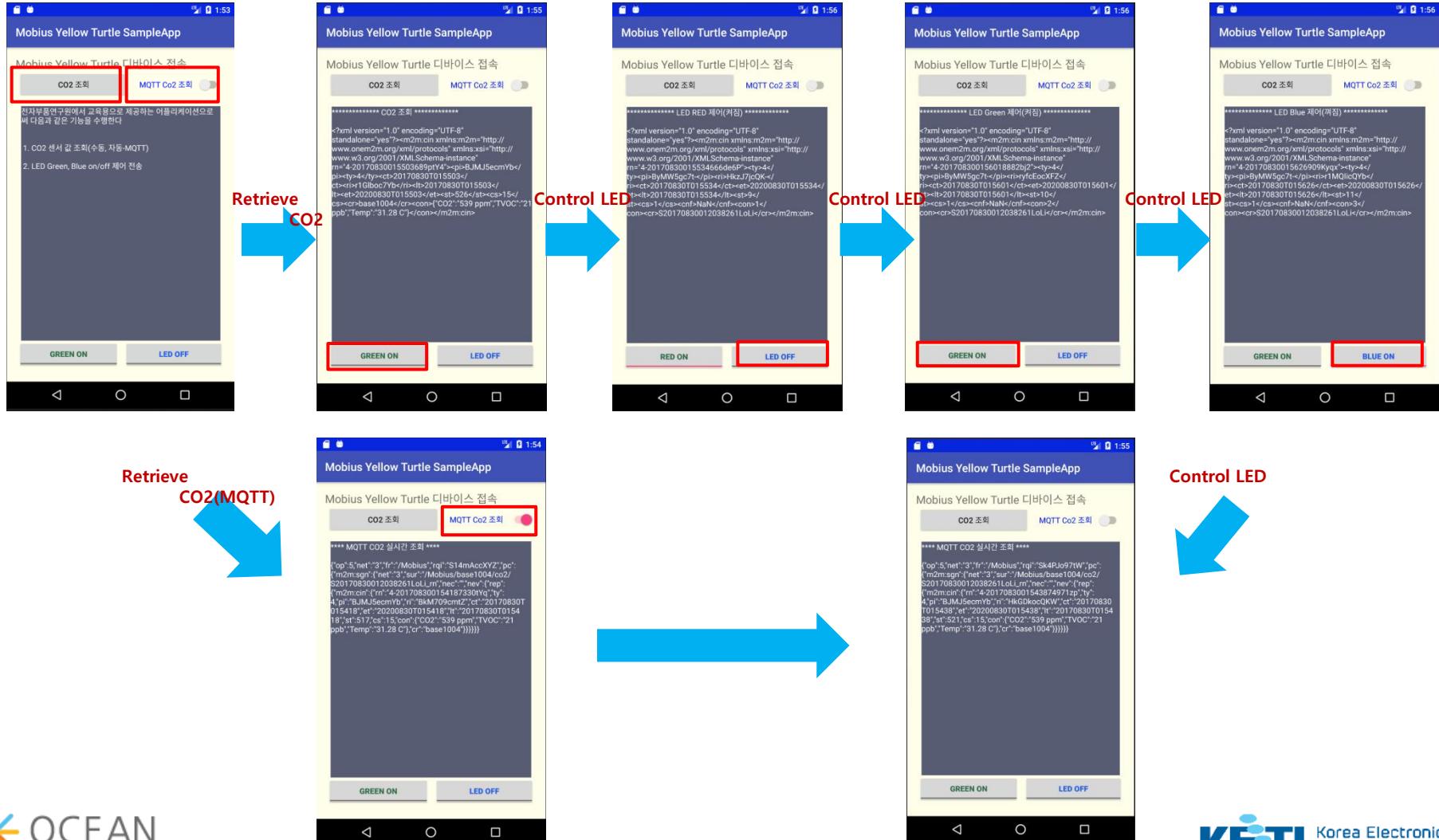
Mobius and APP connectivity

■ App Running



Mobius and APP connectivity

■ App Running



App Code Review

■ Internet Permission Setting [MQTT]

[AndroidManifest.xml]

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="keti.com.mobiusytsampleapp" >

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name="org.eclipse.paho.android.service.MqttService" />
    </application>

</manifest>
```

App Code Review

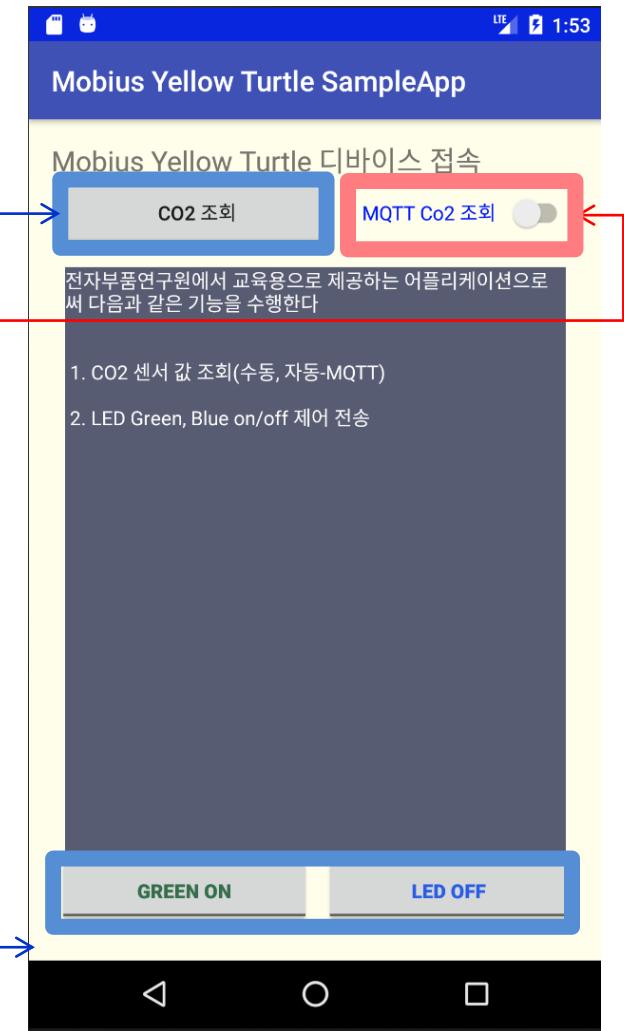
■ Code Structure Review

```
// Main
public MainActivity() { handler = new Handler(); }
/* onCreate */
protected void onCreate(Bundle savedInstanceState) {...}
/* AE Create for Androdi AE */
public void GetAEInfo() {...}
/* Switch - Get Co2 Data With MQTT */
public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {...}
/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {...}
/* MQTT Listener */
private IMqttActionListener mainIMqttActionListener = new IMqttActionListener() {...};
/* MQTT Broker Message Received */
private MqttCallback mainMqttCallback = new MqttCallback() {...};

@Override
public void onClick(View v) {...}
@Override
public void onStart() {...}
@Override
public void onStop() {...}

/* Response callback Interface */
public interface IReceived {...}

/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {...}
/* Request Control LED */
class ControlRequest extends Thread {...}
/* Reqeust AE Creation */
class aeCreateRequest extends Thread {...}
/* Retrieve AE-ID */
class aeRetrieveRequest extends Thread {...}
/* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {...}
```



App Code Review

■ Code Structure Review [AE Create or AE Retrieve …]

```
/* Request AE Creation */
class aeCreateRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    String TAG = aeCreateRequest.class.getName();
    private IReceived receiver;
    int responseCode=0;
    public ApplicationEntityObject applicationEntity;
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
    public aeCreateRequest() {
        applicationEntity = new ApplicationEntityObject();
        applicationEntity.setResourceName(ae.getappName());
    }
    @Override
    public void run() {...}
}
/* Retrieve AE-ID */
class aeRetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    private IReceived receiver;
    int responseCode=0;
    public aeRetrieveRequest() {...}
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
    @Override
    public void run() {...}
}
```



```
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<m2m:ae xmlns:m2m="http://www.onem2m.org/xml/protocols"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  rn="ncubeapp">
  <pi>/Mobius</pi>
  <ty>2</ty>
  <ct>20160928T161212</ct>
  <ri>/mobius-xt/ncubeapp</ri>
  <lt>20160928T161212</lt>
  <api>0.2.481.2.0001.001.0001114</api>
  <aei>xxxxxx</aei>
  <rr>true</rr>
</m2m:ae>
```

Note: In oneM2M standard Android Application is mapping with Application Entity(AE). It should use the AE-ID as request “origin” to do other resource operation(Create, Retrieve, Update, Delete).

App Code Review

■ Code Structure Review [CO2 Retrieve, LED Control]

```
/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(RetrieveRequest.class.getName());
    private IReceived receiver;
    private String ContainerName = "co2";

    public RetrieveRequest(String containerName) { this.ContainerName = containerName; }
    public RetrieveRequest() {}
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

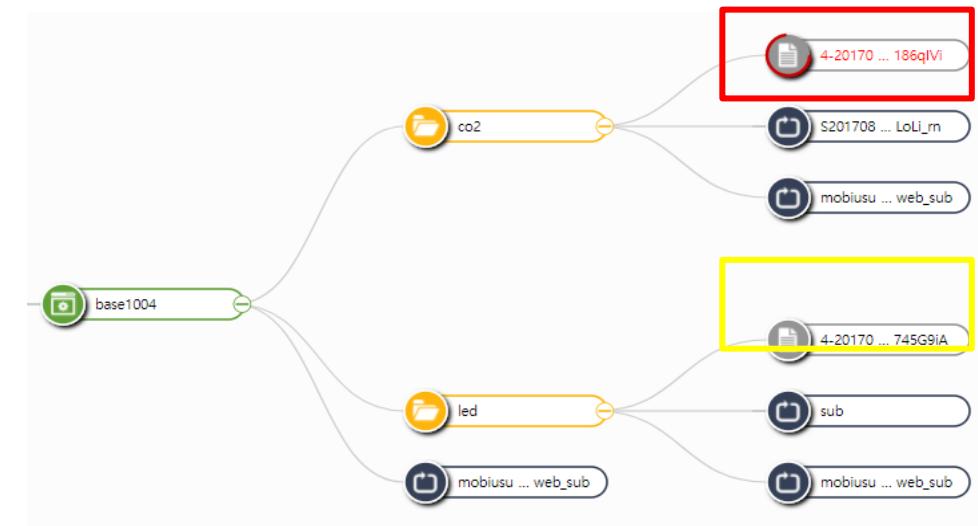
    @Override
    public void run() {...}
}
```

```
/* Request Control LED */
class ControlRequest extends Thread {
    private final Logger LOG = Logger.getLogger(ControlRequest.class.getName());
    private IReceived receiver;
    private String container_name = "led";

    public ContentInstanceObject contentinstance;
    public ControlRequest(String comm) {
        contentinstance = new ContentInstanceObject();
        contentinstance.setContent(comm);
    }
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() {...}
}
```

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:cin
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4-
20160925121654339wkyn"><pi>/Mobius/base1004/led</pi><ty>4</ty><ct>201609
25T121654</ct><ri>/Mobius/base1004/led/4-
20160925121654339wkyn</ri><lt>20160925T121654</lt><st>18</st><mni>900719
9254740991</mni><cs>1</cs><cnf>text</cnf><con>4</con></m2m:cin>
```



```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:cin
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4-
20160925130205542urps"><pi>/Mobius/base1004/co2</pi><ty>4</ty><ct>20160925
T130205</ct><ri>/Mobius/base1004/co2/4-
20160925130205542urps</ri><lt>20160925T130205</lt><st>3756</st><mni>9007199
254740991</mni><cs>3</cs><cnf>719</cnf></m2m:cin>
```

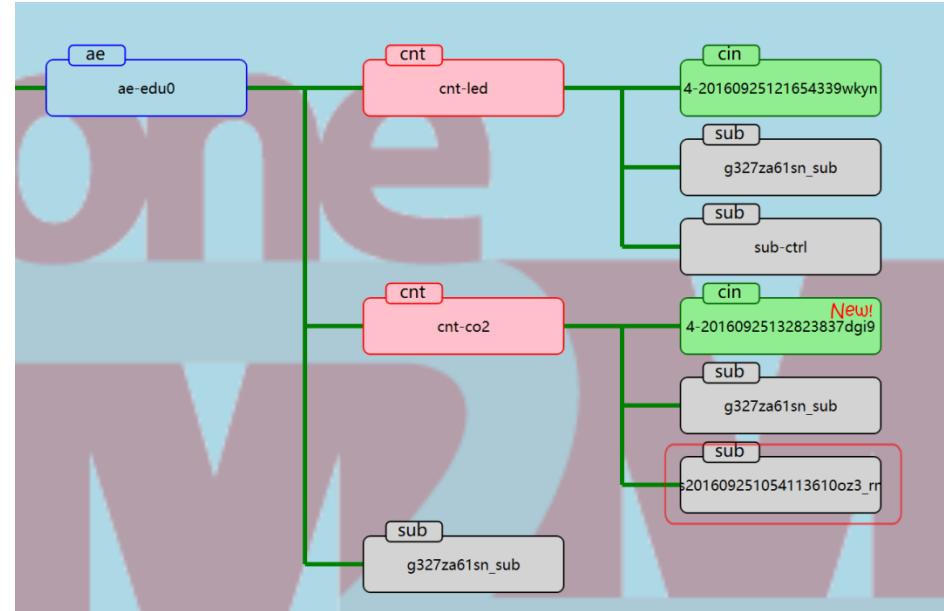
App Code Review

■ Code Structure Review [Subscription create for MQTT]

```
/* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {
    private final Logger LOG = Logger.getLogger(SubscribeResource.class.getName());
    private IReceived receiver;
    private String container_name = "co2"; //change to control container name

    public ContentSubscribeObject subscribeInstance;
    public SubscribeResource() {
        subscribeInstance = new ContentSubscribeObject();
        subscribeInstance.setUrl(csebase.getHost());
        subscribeInstance.setResourceName(ae.getAEid()+"_rn");
        subscribeInstance.setPath(ae.getAEid()+"_sub");
        subscribeInstance.setOrigin_id(ae.getAEid());
    }
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }

    @Override
    public void run() { ... }
}
```



```
<?xml version="1.0" encoding="utf-16" standalone="yes"?><m2m:sub
xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
rn="s201609251054113610oz3_rn"><pi>/mobius-yt/ae-edu0/cnt-
co2</pi><ty>23</ty><ct>20160925T133651</ct><ri>/mobius-yt/ae-edu0/cnt-
co2/s201609251054113610oz3_rn</ri><lt>20160925T133651</lt><st>0</st><enc
><net>3</net></enc><nu>mqt://192.168.25.47/S201609251054113610oz3_sub</nu><nct>2</nct><cr>S201609251054113610oz3</cr></m2m:sub>
```

App Code Review

■ Code Structure Review [Button Click Event]

```

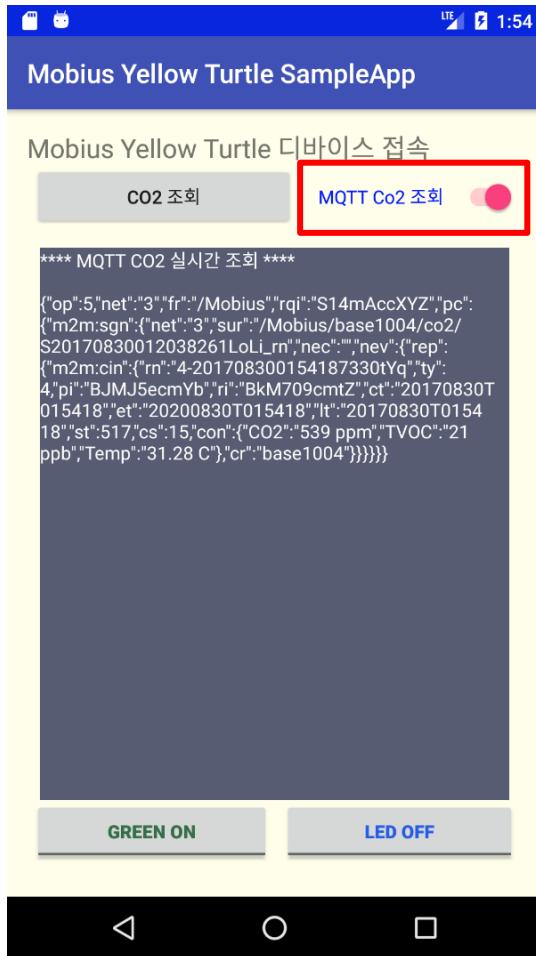
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.btnRetrieve: {
            RetrieveRequest req = new RetrieveRequest();
            textViewData.setText("");
            req.setReceiver(new IReceived() {
                @Override
                public void getResponseBody(final String msg) {
                    handler.post(new Runnable() {
                        @Override
                        public void run() {
                            textViewData.setText("***** CO2 조회 ***");
                        }
                    });
                }
            });
            req.start();
            break;
        }
        ...
    }
}

case R.id.btnControl_Green: {
    if (((ToggleButton) v).isChecked()) {
        ControlRequest req = new ControlRequest("1");
        req.setReceiver(new IReceived() {
            @Override
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        textViewData.setText("***** LED RED 제어(켜짐)..");
                    }
                });
            }
        });
        req.start();
    } else {
        ControlRequest req = new ControlRequest("2");
        req.setReceiver(new IReceived() {
            @Override
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        textViewData.setText("***** LED Green 제어(켜짐)..");
                    }
                });
            }
        });
        req.start();
    }
}
break;
}

```

App Code Review

■ Code Structure Review [Switch onCheckChanged Event]



```

public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
    if (isChecked) {
        Log.d(TAG, "MQTT Create");
        MQTT_Create(true);
    } else {
        Log.d(TAG, "MQTT Close");
        MQTT_Create(false);
    }
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subscribeResource = new SubscribeResource();
        subscribeResource.setReceiver(new IReceived() {
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    public void run() { ... }
                });
            }
        });
        subscribeResource.start();
    }
}

/* MQTT Subscribe */
mqttClient = new MqttAndroidClient(this.getApplicationContext(), "tcp://" + csebase.getHost() + ":" +
csebase.getMQTTPort(), MqttClient.generateClientId());
mqttClient.setCallback(mainMqttCallback);
try {
    IMqttToken token = mqttClient.connect();
    token.setActionCallback(mainIMqttActionListener);
} catch (MqttException e) {
    e.printStackTrace();
}
} else {
    /* MQTT unSubscribe or Client Close */
    mqttClient.setCallback(null);
    mqttClient.close();
    mqttClient = null;
}
}

```

App Code Review

■ Retrieve by POSTMAN [CO2 Sensing Data]

The screenshot shows the POSTMAN interface with the following details:

- Method:** GET (highlighted with a red box)
- URL:** {{mp_url}}/{{cb}}/base1004/co2/latest
- Headers:** (3)
 - Accept: application/xml
 - X-M2M-RI: 12345
 - X-M2M-Origin: SOrigin
- Body:** (7) (highlighted with a red box)
- Status:** 200 OK Time: 74 ms
- Response:** XML (Pretty, Raw, Preview, XML, JSON)
 - Pretty-printed XML response:

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4
-20170830021402583w5D2">
3 <pi>83MJ5ecmYb</pi>
4 <ty>4</ty>
5 <ct>20170830T021402</ct>
6 <ri>ryf7uysmY-</ri>
7 <lt>20170830T021402</lt>
8 <et>20200830T021402</et>
9 <st>754</st>
10 <cs>15</cs>
11 <cr>base1004</cr>
12 <con>{"CO2":"588 ppm","TVOC":"28 ppb","Temp":"32.01 C"}</con>
13 </m2m:cin>
```

App Code Review

■ Retrieve by Java Code [CO2 Sensing Data]

```

private String ServiceAENAME = "base1004";
...
public void GetAEInfo() {
    csebase.setInfo("203.253.128.161", "7579", "Mobius", "1883"); //Sample IP
...
}
/* Retrieve Co2 Sensor */
class RetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(RetrieveRequest.class.getName());
    private IReceived receiver;
    private String ContainerName = "co2";

    public RetrieveRequest(String containerName) {
        this.ContainerName = containerName;
    }
    public RetrieveRequest() {}
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
}

@Override
public void run() {
    try {
        String sb = csebase.getServiceUrl() + "/" + ServiceAENAME + "/" + ContainerName + "/" + "latest";
        URL mUrl = new URL(sb);

        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        conn.setDoOutput(false);

        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-Origin", ae.getAEid() );
        conn.setRequestProperty("nmtype", "long");
        conn.connect();

        String strResp = "";
        BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));

        String strLine= "";
        while ((strLine = in.readLine()) != null) {
            strResp += strLine;
        }

        if ( strResp != "" ) {
            receiver.getResponseBody(strResp);
        }
        conn.disconnect();
    } catch (Exception exp) {
        LOG.log(Level.WARNING, exp.getMessage());
    }
}
}

```



<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4 -20170830021402583w5D2">
<pi>B3MJ5ecmYb</pi>
<ty>4</ty>
<cct>20170830T021402</cct>
<ri>ryf7uysmY-</ri>
<lt>20170830T021402</lt>
<et>20200830T021402</et>
<st>754</st>
<cs>15</cs>
<cr>base1004</cr>
<con>{"CO2":"588 ppm", "TVOC":"28 ppb", "Temp":"32.01 C"}</con>
</m2m:cin>

App Code Review

■ Control by POSTMAN [LED Control]

▶ cin 생성

POST `{{mp_url}}/{{cb}}/base1004/led`

Headers (4)

Key	Value	Description
Accept	application/json	
X-M2M-RI	12345	
X-M2M-Origin	5000000000012157039160	
Content-Type	application/vnd.onem2m-rest+json; ty=4	

▶ cin 생성

POST `{{mp_url}}/{{cb}}/base1004/led`

Body (1)

raw

```
{
  "m2m:cin": {
    "con": "1"
  }
}
```

Status: 201 Created Time: 151 ms

Pretty

```
{
  "m2m:cin": {
    "rn": "4-20170830022116201K8tt",
    "ty": 4,
    "pi": "ByMw5gc7t-",
    "ri": "SJMNXBj7tZ",
    "ct": "20170830T022116",
    "et": "20200830T022116",
    "lt": "20170830T022116",
    "st": 13,
    "rc": 1,
    "con": "1",
    "cr": "5000000000012157039160"
  }
}
```

1: RED LED ON

2: Green LED ON

3: Blue LED ON

0: ALL LED OFF

App Code Review

■ Control by Java Code [LED Control]

```
private String ServiceAENAME = "base1004";
```

```
...
```

```
public void GetAEInfo() {
```

```
    csebase.setInfo("203.253.128.161", "7579", "Mobius", "1883");
```

```
...
```

```
...
```

```
} <?xml version="1.0" encoding="UTF-8"?>
<m2m:cin
  xmlns:m2m="http://www.onem2m.org/xml/protocols"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <cnf>text</cnf>
  <con>1</con>
</m2m:cin>|
```

```
class ControlRequest extends Thread {
    private final Logger LOG = Logger.getLogger(ControlRequest.class.getName());
    private IReceived receiver;
    private String container_name = "led";

    public ContentInstanceObject contentinstance;
    public ControlRequest(String comm) {
        contentinstance = new ContentInstanceObject();
        contentinstance.setContent(comm);
    }
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
```

```
@Override
public void run() {
try {
    String sb = csebase.getServiceUrl() + "/" + ServiceAENAME + "/" + container_name;

    URL mUrl = new URL(sb);

    HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
    conn.setRequestMethod("POST");
    conn.setDoInput(true);
    conn.setDoOutput(true);
    conn.setUseCaches(false);
    conn.setInstanceFollowRedirects(false);

    conn.setRequestProperty("Accept", "application/xml");
    conn.setRequestProperty("Content-Type", "application/vnd.onem2m-res+xml;ty=4");
    conn.setRequestProperty("locale", "ko");
    conn.setRequestProperty("X-M2M-RI", "12345");
    conn.setRequestProperty("X-M2M-Origin", ae.getAEid() );

    String reqContent = contentinstance.makeXML();
    conn.setRequestProperty("Content-Length", String.valueOf(reqContent.length()));

    DataOutputStream dos = new DataOutputStream(conn.getOutputStream());
    dos.write(reqContent.getBytes());
    dos.flush();
    dos.close();

    BufferedReader in = new BufferedReader(new
InputStreamReader(conn.getInputStream()));

    String resp = "";
    String strLine="";
    while ((strLine = in.readLine()) != null) {
        resp += strLine;
    }
    if (resp != "") {
        receiver.getResponseBody(resp);
    }
    conn.disconnect();

} catch (Exception exp) {
    LOG.log(Level.SEVERE, exp.getMessage());
}
}
```




```
{
  "m2m:cin": {
    "rn": "4-20170830022116201K8tt",
    "ty": 4,
    "pi": "ByMW5gc7t-",
    "ri": "SJMNXbj7tZ",
    "ct": "20170830T022116",
    "et": "20200830T022116",
    "lt": "20170830T022116",
    "st": 13,
    "cs": 1,
    "con": "1",
    "cr": "S000000000012157039160"
  }
}
```

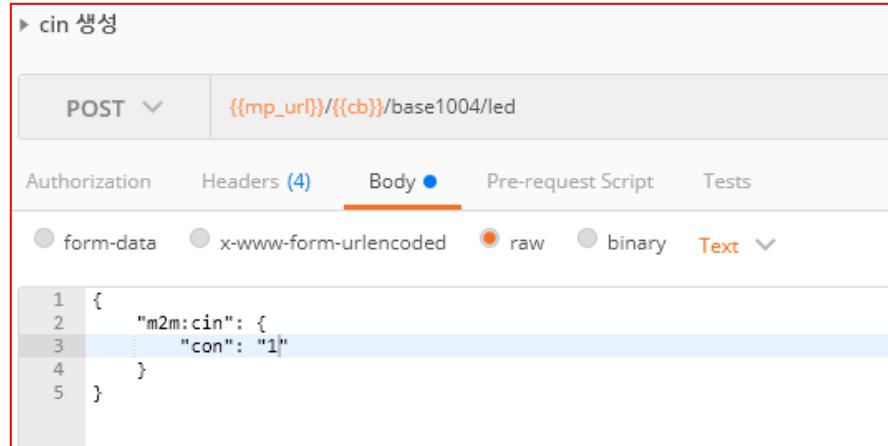




KETI Korea Electronics
Technology Institute

App Code Review

■ Content Instance Creation Java Code



```
public class ContentInstanceObject {
    private String content = "";
    public void setContent(String contentValues) {
        this.content = contentValues;
    }

    public String makeXML() {
        String xml = "";

        xml += "<?xml version="W"1.0W" encoding="W"UTF-8W"?>";
        xml += "<m2m:cin ";
        xml += "xmlns:m2m=W"http://www.onem2m.org/xml/protocolsW" ";
        xml += "xmlns:xsi=W"http://www.w3.org/2001/XMLSchema-instanceW">";
        xml += "<con>" + content + "</con>";
        xml += "</m2m:cin>";

        return xml;
    }
}
```

JSON

XML

App Code Review

■ AE Create by POSTMAN

M-IF50. AE 리소스 생성

POST {{mp_url}}/{{cb}}

Headers (6)

Accept	application/xml
locale	ko
X-M2M-RI	12345
X-M2M-Origin	S
Content-Type	application/vnd.onem2m-res+xml; ty=2
nmtype	short

POST {{mp_url}}/{{cb}}

Body

Text

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <m2m:ae xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp">
3   <api>0.2.481.2.0001.001.0001112</api>
4   <rr>true</rr>
5 </m2m:ae>
    
```

Status: 201 Created

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <m2m:ae xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp">
3   <ty>2</ty>
4   <pi>/mobius-yt</pi>
5   <ri>/mobius-yt/ncubeapp</ri>
6   <ct>20160928T193850</ct>
7   <lt>20160928T193850</lt>
8   <api>0.2.481.2.0001.001.0001112</api>
9   <rr>true</rr>
10  <aei>ncubeapp</aei>
11 </m2m:ae>
    
```

Status: 409 Conflict

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <m2m:rsp xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3   <cap>resource is already exist</cap>
4 </m2m:rsp>
    
```

App Code Review

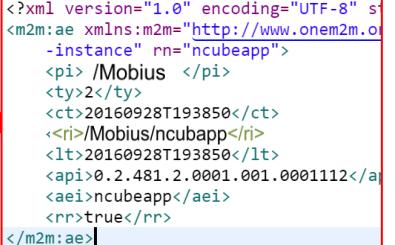
■ AE Create by Java Code

```
private String ServiceAEName = "base1004";
...
public void GetAEInfo() {
    csebase.setInfo("203.253.128.161","7579","Mobius","1883");
    ae.setAppName("ncubeapp");
...
}
```

```
... <?xml version="1.0" encoding="UTF-8"?>
<m2m:ae
  xmlns:m2m="http://www.onem2m.org/xml/protocols"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  rn="ncubeapp">
  <api>0.2.481.2.0001.001.0001114</api>
  <rr>true</rr>
</m2m:ae>
```

```
class aeCreateRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    String TAG = aeCreateRequest.class.getName();
    private IReceived receiver;
    int responseCode=0;
    public ApplicationEntityObject applicationEntity;
    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
    public aeCreateRequest(){
        applicationEntity = new ApplicationEntityObject();
        applicationEntity.setResourceName(ae.getappName());
    }
}
```

```
public void run() {
    try {
        String sb = csebase.getServiceUrl();
        URL mUrl = new URL(sb);
        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("POST");
        conn.setDoInput(true);
        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setInstanceFollowRedirects(false);
        conn.setRequestProperty("Content-Type", "application/vnd.onem2m-res+xml;ty=2");
        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("locale", "ko");
        conn.setRequestProperty("X-M2M-Origin", "S");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-NM", ae.getappName() );
        String reqXml = applicationEntity.makeXML();
        conn.setRequestProperty("Content-Length", String.valueOf(reqXml.length()));
        DataOutputStream dos = new DataOutputStream(conn.getOutputStream());
        dos.write(reqXml.getBytes());
        dos.flush();
        dos.close();
        responseCode = conn.getResponseCode();
        BufferedReader in = null;
        String aei = "";
        if (responseCode == 201) {
            // Get AEID from Response Data
            in = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            String resp = "";
            String strLine;
            while ((strLine = in.readLine()) != null) {
                resp += strLine;
            }
            ParseElementXml pxml = new ParseElementXml();
            aei = pxml.getElementXml(resp, "aei");
            ae.setAEid( aei );
            Log.d(TAG, "Create Get AEID[" + aei + "]");
            in.close();
        }
        if (responseCode != 0) {
            receiver.getResponseBody( Integer.toString(responseCode) );
            conn.disconnect();
        } catch (Exception exp) {
            LOG.log(Level.SEVERE, exp.getMessage());
        }
    }
}
```

App Code Review

■ AE Retrieve POSTMAN [AE-ID]

The screenshot shows the POSTMAN application interface with the following details:

- Request Method:** GET
- URL:** {{mp_url}}/{{ct}}/ncubeapp
- Authorization:** No Auth
- Status:** 200 OK
- Time:** 48 ms
- Response Body (Pretty JSON):**

```
1 [
2   "m2m:ae": {
3     "pi": "rkpPvqMKZ",
4     "ty": 2,
5     "ct": "20170830T012038",
6     "ri": "S20170830012038261LoLi",
7     "rn": "ncubeapp",
8     "lt": "20170830T012038",
9     "et": "20200830T012038",
10    "api": "0.2.481.2.0001.001.0001114",
11    "aei": "S20170830012038261LoLi",
12    "rr": true
13  }
14 }
```

App Code Review

■ AE Retrieve Java Code

```

private String ServiceAEName = "base1004";
...
public void GetAEInfo() {
    csebase.setInfo("203.253.128.161", "7579", "Mobius", "1883");
    ae.setAppName("ncubeapp");
...
}
/* Retrieve AE-ID */
class aeRetrieveRequest extends Thread {
    private final Logger LOG = Logger.getLogger(aeCreateRequest.class.getName());
    private IReceived receiver;
    int responseCode=0;

    public aeRetrieveRequest() {
    }
    public void setReceiver(IReceived hanlder) {
        this.receiver = hanlder;
    }
}

@Override
public void run() {
    try {
        String sb = csebase.getServiceUrl() + "/" + ae.getappName();
        URL mUrl = new URL(sb);

        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("GET");
        conn.setDoInput(true);
        conn.setDoOutput(false);

        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-Origin", "Sandoroid");
        conn.setRequestProperty("nmtype", "short");
        conn.connect();

        responseCode = conn.getResponseCode();

        BufferedReader in = null;
        String aei = "";
        if (responseCode == 200) {
            // Get AEID from Response Data
            in = new BufferedReader(new InputStreamReader(conn.getInputStream()));

            String resp = "";
            String strLine;
            while ((strLine = in.readLine()) != null) {
                resp += strLine;
            }

            ParseElementXml pxml = new ParseElementXml();
            aei = pxml.getElementXml(resp, "aei");
            ae.setAEid( aei );
            Log.d(TAG, "Retrieve Get AEID[" + aei + "]");
            in.close();
        }
        if (responseCode != 0) {
            receiver.getResponseBody( Integer.toString(responseCode) );
        }
        conn.disconnect();
    } catch (Exception exp) {
        LOG.log(Level.SEVERE, exp.getMessage());
    }
}

```



```

{
    "m2m:ae": {
        "pi": "rkPvqMKZ",
        "ty": 2,
        "ct": "20170830T012038",
        "ri": "S20170830012038261LoLi",
        "rn": "ncubeapp",
        "lt": "20170830T012038",
        "et": "20200830T012038",
        "api": "0.2.481.2.0001.001.0001114",
        "aei": "S20170830012038261LoLi",
        "rr": true
    }
}

```

App Code Review

■ Subscription Creation by POSTMAN

The screenshot shows two instances of the POSTMAN application interface, illustrating the process of creating a subscription.

Request Headers:

```

POST {{mp_url}}/{{cb}}/base1004/co2
Headers (4)
Key          Value
Accept       application/xml
X-M2M-RI     12345
X-M2M-Origin SOrigin
Content-Type application/vnd.onem2m-rest+xml; ty=23
  
```

Request Body (Text):

```

<?xml version="1.0" encoding="UTF-8"?>
<m2m:sub xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp_rn">
  <enc>
    <net>3</net>
  </enc>
  <nu>mqtt://203.253.128.161/ncubeapp_rn</nu>
</m2m:sub>
  
```

Response Status:

Status: 201 Created Time: 330 ms

Response Body (Pretty):

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:sub xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="ncubeapp_rn">
  <ty>23</ty>
  <pi>BJMj5ecmYb</pi>
  <ri>SJzkm_sQtW</ri>
  <ct>20170830T025103</ct>
  <et>20200830T025103</et>
  <ln>20170830T025103</ln>
  <nu>mqtt://203.253.128.161/ncubeapp_rn</nu>
  <enc>
    <net>3</net>
  </enc>
  <bn>
    <num>0</num>
    <dur>10</dur>
  </bn>
  <nct>2</nct>
  <cr>SOrigin</cr>
</m2m:sub>
  
```

App Code Review

■ Subscription Creation by Java Code

```

private String ServiceAENAME = "base1004";
public void GetAEInfo() {
    csebase.setInfo("203.253.128.161","7579","Mobius","1883");
    ae.setAppName("ncubeapp");
    ...
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subscribeResource = new SubscribeResource();
        subscribeResource.setReceiver(new IReceived() {
            public void getResponseBody(final String msg) {
                handler.post(new Runnable() {
                    public void run() {
                        ...
                    }
                });
            }
        });
        subscribeResource.start();
        ...
    }
}

/* Subscribe Co2 Content Resource */
class SubscribeResource extends Thread {
    private IReceived receiver;
    private String container_name = "cnt-co2"; //change to control container name

    public ContentSubscribeObject subscribeInstance;
    public SubscribeResource() {
        subscribeInstance = new ContentSubscribeObject();
        subscribeInstance.setUrl(csebase.getHost());
        subscribeInstance.setResourceName(ae.getAEid()+"_rn");
        subscribeInstance.setPath(ae.getAEid()+"_sub");
        subscribeInstance.setOrigin_id(ae.getAEid());
    }

    public void setReceiver(IReceived hanlder) { this.receiver = hanlder; }
}

```

```

public void run() {
    try {
        String sb = csebase.getServiceUrl() + "/" + ServiceAENAME + "/" + container_name;
        URL mUrl = new URL(sb);

        HttpURLConnection conn = (HttpURLConnection) mUrl.openConnection();
        conn.setRequestMethod("POST");
        conn.setDoInput(true);
        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setInstanceFollowRedirects(false);

        conn.setRequestProperty("Accept", "application/xml");
        conn.setRequestProperty("Content-Type", "application/vnd.onem2m-res+xml; ty=23");
        conn.setRequestProperty("locale", "ko");
        conn.setRequestProperty("X-M2M-RI", "12345");
        conn.setRequestProperty("X-M2M-Origin", ae.getAEid());

        String reqmqttContent = subscribeInstance.makeXML();
        conn.setRequestProperty("Content-Length", String.valueOf(reqmqttContent.length()));

        DataOutputStream dos = new DataOutputStream(conn.getOutputStream());
        dos.write(reqmqttContent.getBytes());
        dos.flush();
        dos.close();

        BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream()));

        String resp = "";
        String strLine="";
        while ((strLine = in.readLine()) != null) {
            resp += strLine;
        }

        if (resp != "") {
            receiver.getResponseBody(resp);
        }
        conn.disconnect();
    } catch (Exception exp) {
        LOG.log(Level.SEVERE, exp.getMessage());
    }
}

```

App Code Review

■ MQTT Client Subscription, Listener, Callback, Response Java Code

```

private String ServiceAENAME = "base1004";
public void GetAEInfo() {
    csebase.setInfo("203.253.128.161", "7579", "Mobius", "1883");
    ae.setAppName("ncubeapp");
    ...
    MQTT_Req_Topic = "/oneM2M/req/:mobius-yt/" + ae.getAEid() + "_sub" + "/xml";
    MQTT_Resp_Topic = "/oneM2M/resp/:mobius-yt/" + ae.getAEid() + "_sub" + "/xml";
}

/* MQTT Subscription */
public void MQTT_Create(boolean mtqqStart) {
    if (mtqqStart && mqttClient == null) {
        /* Subscription Resource Create to Yellow Turtle */
        SubscribeResource subscribeResource = new SubscribeResource();
        subscribeResource.setReceiver(new IReceived() {
            ...
            subscribeResource.start();

        /* MQTT Subscribe */
        mqttClient = new MqttAndroidClient(this.getApplicationContext(), "tcp://" +
csebase.getHost() + ":" + csebase.getMQTTPort(), MqttClient.generateClientId());
        mqttClient.setCallback(mainMqttCallback);
        try {
            IMqttToken token = mqttClient.connect();
            token.setActionCallback(mainIMqttActionListener);
        } catch (MqttException e) {
            e.printStackTrace();
        }
    } else {
        /* MQTT unSubscribe or Client Close */
        mqttClient.setCallback(null);
        mqttClient.close();
        mqttClient = null;
    }
}

```

```

/* MQTT Listener */
private IMqttActionListener mainIMqttActionListener = new IMqttActionListener() {
    @Override
    public void onSuccess(IMqttToken asyncActionToken) {
        String payload = "";
        int mqttQos = 1; /* 0: NO QoS, 1: No Check , 2: Each Check */

        MqttMessage message = new MqttMessage(payload.getBytes());
        try {
            mqttClient.subscribe(MQTT_Req_Topic, mqttQos);
        } catch (MqttException e) {
            e.printStackTrace();
        }
    }
    ...
};

/* MQTT Broker Message Received */
private MqttCallback mainMqttCallback = new MqttCallback() {
    @Override
    public void connectionLost(Throwable cause) { Log.d(TAG, "connectionLost"); }
    @Override
    public void messageArrived(String topic, MqttMessage message) throws Exception {
        Log.d(TAG, "messageArrived");
        textViewData.setText("");
        textViewData.setText("**** MQTT CO2 실시간 조회 ****\n\n" +
message.toString().replaceAll("\t|\n", ""));
        ArrayList<String> mqttMessage = new ArrayList<String>();
        mqttMessage = MqttClientRequestParser.notificationParse(message.toString());
        String responseMessage = MqttClientRequest.notificationResponse(mqttMessage);

        /* Make xml for MQTT Response Message */
        MqttMessage resmessage = new MqttMessage(responseMessage.getBytes());

        try {
            mqttClient.publish(MQTT_Resp_Topic, resmessage);
        } catch (MqttException e) { e.printStackTrace(); }
    }
    @Override
    public void deliveryComplete(IMqttDeliveryToken token) {
        Log.d(TAG, "deliveryComplete");
    }
};

```

App Code Review

■ Android Studio Generate APK

- Build -> Generate Signed APK...
- If have Key store, Select “Choose existing”
- If do not have Key store, Select “Create new”
- Save location and input file name, Password, Alias, Certification information.
- Select key file
- Locate the APK path.

