



# 공공 DB API 활용

# Contents title

## 18.1 공공 DB 오픈API 원리와 활용

### 18.1.1 공공 DB 오픈API 활용의 예

### 18.1.2 공공 DB 오픈API 원리

### 18.1.3 공공 DB 오픈API 인증키 받기

## 18.2 버스노선의 ID 조회: Bus Route ID 프로젝트(노선ID)

### 18.2.1 프로젝트 개요

### 18.2.2 프로젝트 개발

### 18.2.3 노선ID를 포함하는 XML 문서 출력

### 18.2.4 XML 문서 파싱과 노선ID 추출

## 18.3 노선버스위치 조회: Bus Positions 프로젝트(노선버스 위치)

### 18.3.1 프로젝트 개요

### 18.3.2 프로젝트 개발

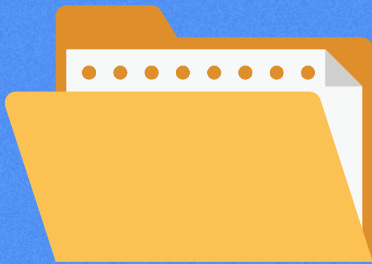
## 18.4 노선ID와 위치정보의 결합: Seoul Bus Positions(서울버스 위치)

### 18.4.1 프로젝트 개요

### 18.4.2 앱 프로젝트 개발







## 18.1 공공 DB 오픈 API 원리와 활용



## 18.1.1

## 공공 DB 오픈API 활용의 예



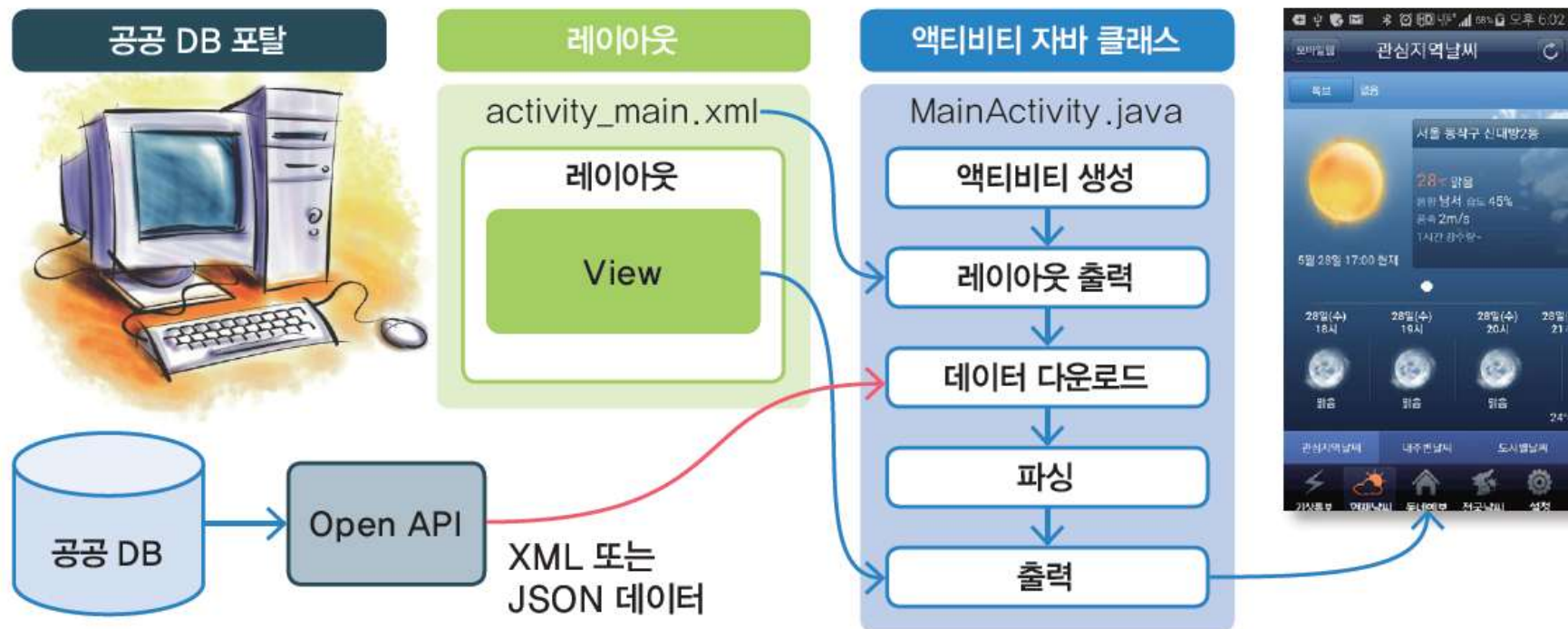
(a) 서울 버스노선번호 운행정보



(b) 기상청 날씨 정보

- 공공 DB 활용 앱

## 18.1.2 공공 DB 오픈API 원리





18.1.3

## 공공 DB 오픈 API 인증키 받기

STEP 1 회원가입과 로그인

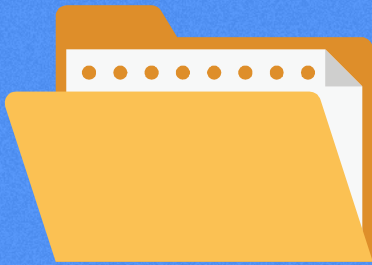
STEP 2 오픈 API 검색

STEP 3 서비스 활용신청

STEP 4 신청현황 조회

STEP 5 인증키 확인

STEP 6 OPEN API 활용방법 확인



## 18.2 버스노선의 ID 조화: Bus Route ID 프로젝트 (노선ID)



## 18.2.1 프로젝트 개요

### 프로젝트 개요: 버스노선ID 조회

Application Name: Bus Route ID

어플리케이션 라벨: 노선ID

#### 노선ID를 포함하는 XML 문서

```
<?xml version="1.0" encoding="UTF-8"
standalone="yes"?><ServiceResult><comMsgHeade
r/><msgHeader><headerCd>0</
headerCd><headerMsg>정상적으로 처리되었습니다.</
headerMsg><itemCount>0</itemCount></
msgHeader><msgBody><itemList><busRouteId>10
0100064</busRouteId><busRouteNm>406</
busRouteNm><corpNm>도선여객 02-574-0101</
corpNm><edStationNm>서울역</
edStationNm><firstBusTm>20160421042000</
firstBusTm><firstLowTm>20151125045000</
firstLowTm><lastBusTm>20160421232000</
lastBusTm><lastBusYn>0</
lastBusYn><lastLowTm>20150717224400</
lastLowTm><length>42.59</length><routeType>3</
routeType><stStationNm>개포동</
stStationNm><term>13</term></itemList></
msgBody></ServiceResult>
```

#### XML 문서로부터 파싱한 노선ID

```
headerCd: 0
busRouteId: 100100064
busRouteNm: 406
```



## STEP 1 프로젝트 생성

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: <b>Bus Route ID</b> Company Domain: <b>yschang.example.com</b>
③ 제품형태	Phone and Tablet
④ 액티비티 유형	Empty Activity
⑤ 파일 옵션	디폴트 값으로 설정

## STEP 2

## 파일 편집

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	• 인터넷 사용 허용
java	com.example. yschang. busrouteid	MainActivity.java	• 노선ID 파악을 위한 공공 DB 오픈 API 호출 • XML 문서 파싱 및 출력
res	drawable		
	layout	activity_main.xml	• XML 문서 출력을 위한 텍스트뷰 배치
	mipmap	ic_launcher.png	
	values	colors.xml	
		dimens.xml	
		strings.xml	• 어플리케이션 라벨 수정
		styles.xml	

■ 수정 ■ 추가

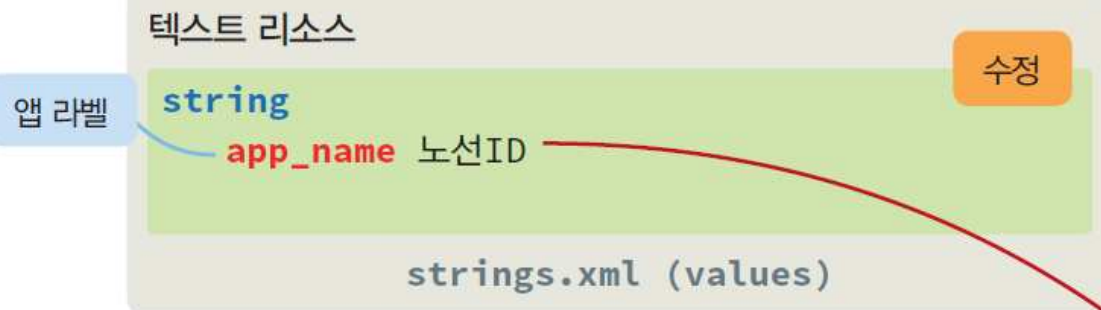
## ● 파일 간의 연관관계

**strings.xml**에는 초기치로 설정되어 있는 어플리케이션 라벨을 '노선ID'로 수정한다.

**activity\_main.xml**에는 공공 DB API(노선ID) 호출 후에 반환되는 XML 문서를 출력할 텍스트뷰를 배치한다.

**MainActivity.java**에는 공공 DB API(노선ID) 호출 후에 반환되는 XML 문서를 출력한다.

**AndroidManifest.xml**에는 인터넷 사용을 허용하도록 설정한다.





## 화면 레이아웃

RelativeLayout  
TextView

id @+id/data

activity\_main.xml (layout)

수정

텍스트뷰 인식

## 액티비티 제어

onCreate()

super.onCreate()

setContentView(R.layout.activity\_main)

tv = findViewById(R.id.data)

...

strurl = "http://....."

new DownloadWebpageTask().

execute(strurl)

MainActivity.java (java)

수정

컴파일/빌더

노션D API의 URL

URL에 해당하는 API 호출하고  
반환되는 XML 문서를 출력

## 어플리케이션 기본 정보

uses-permission

name android.permission.INTERNET

application

icon @mipmap/ic\_launcher

label @string/app\_name

theme @style/AppTheme

activity

name MainActivity

AndroidManifest.xml (manifest)

수정

인터넷 접속 허용



사용자

출력



스마트폰

## 컴파일/빌더 정보

```
build gradle(Project)
build gradle(Module app)
gradle properties
settings gradle
local properties
```

(Gradle Scripts)

### 18.2.3

## 노선ID를 포함하는 XML 문서 출력

### ● 편집

#### 1 텍스트 자원의 편집

소스 | strings.xml

```
01 <resources>
01     <string name="app_name">노선ID</string>
02 </resources>
```

app\_name의 데이터를 '노선ID'로 수정

## 2 화면 설계

소스 | activity\_main.xml → A

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
03     android:layout_width="match_parent"
04     android:layout_height="match_parent" >
05
06     <RelativeLayout
07         xmlns:tools="http://schemas.android.com/tools"
08         android:layout_width="match_parent"
09         android:layout_height="match_parent"
10         android:paddingBottom="@dimen/activity_vertical_margin"
11         android:paddingLeft="@dimen/activity_horizontal_margin"
12         android:paddingRight="@dimen/activity_horizontal_margin"
13         android:paddingTop="@dimen/activity_vertical_margin"
14         tools:context="com.example.busrouteid.MainActivity">
15
16         <TextView
17             android:id="@+id/data"
18             android:layout_width="wrap_content"
19             android:layout_height="wrap_content" />
20     </RelativeLayout>
21
22 </ScrollView>
```

노선ID 추출 결과  
출력을 위한 텍스트뷰



### 3 액티비티 제어

#### 소스 | MainActivity.java

```
24  @Override
25  protected void onCreate(Bundle savedInstanceState) {
26      super.onCreate(savedInstanceState);
27      setContentView(R.layout.activity_main);
28
29      tv = (TextView) findViewById(R.id.data);
30
31      String serviceUrl = "http://ws.bus.go.kr/api/rest/busRouteInfo/
32                          getBusRouteList";
33      String serviceKey = "DEp3%2B.....5DQ%3D%3D";
34      String strSrch = "406";
35      String strUrl = serviceUrl+"?ServiceKey="+serviceKey+
36                      "&strSrch="+strSrch;
37
38      new DownloadWebpageTask().execute(strUrl);
39
40      private class DownloadWebpageTask extends AsyncTask<String,
41                          Void, String> {
42
43          @Override
44          protected String doInBackground(String... urls) {
45              try {
46                  return (String) downloadUrl((String) urls[0]);
47              } catch (IOException e) {
48                  return "다운로드 실패";
49              }
50          }
51      }
```

노선ID 추출을 위한 공공 DB API의 URL

노선ID 추출을 위한 공공 DB API 키

노선버스의 노선번호

공공 DB API 호출을 위한 URL

1 URL에 해당하는 문서 추출을 위한 객체 생성

2 downloadUrl 메소드 호출하여 문서 추출

```

50 protected void onPostExecute(String result) {
51     tv.setText(result);
52 }

```

③ 문서 추출 결과의 출력

```

54 private String downloadUrl(String myurl) throws IOException {
55
56     HttpURLConnection conn = null;
57     try {
58         URL url = new URL(myurl);
59         conn = (HttpURLConnection) url.openConnection();
60         BufferedInputStream buf = new BufferedInputStream(
61                                     conn.getInputStream());
62         BufferedReader bufreader = new BufferedReader(
63                                     new InputStreamReader(buf, "utf-8"));
64
65         String line = null;
66         String page = "";
67         while((line = bufreader.readLine()) != null) {
68             page += line;
69         }
70
71         return page;
72     } finally {
73         conn.disconnect();
74     }
75 }

```

HTTP로 URL 접속을 위한 객체 생성

URL 객체 생성

URL 객체를 이용한 HTTP 연결

버퍼에 문서 다운로드

버퍼 내용을  
UTF-8 문서 형식으로  
변환

추출한 웹문서의 문서 내용을 반환

버퍼 내용을  
행 단위로 읽어  
문자 변수에 저장

HTTP 연결 해제

URL에 해당하는 웹문서 다운로드

# Class와 속성/메소드

## ● 클래스

클래스/인터페이스	설명
AsyncTask	<p>39행. 백그라운드로 작업을 수행하며, UI에 결과를 출력함. AsyncTask의 서브 클래스를 만들어 작업을 수행함.</p> <p>서브 클래스를 만들 때는 AsyncTask&lt;Params, Progress, Result&gt;의 3가지 매개변수를 지정해야 함: Params는 execute() 메소드의 인자값에 해당하는 데이터 타입, Progress는 doInBackground() 메소드에서 publishProgress() 메소드 호출 시의 인자값에 해당하는 데이터 타입, Result는 doInBackground() 메소드에서 반환되는 값과 onPostExecute()의 매개변수 값에 해당하는 데이터 타입임</p>
BufferedInputStream	60행. InputStream으로 받은 byte 단위의 데이터를 버퍼에 저장하는 클래스
BufferedReader	61행. Reader로부터 읽은 텍스트 데이터를 버퍼에 저장하는 클래스
IOException	45행. 입출력 관련 에러 정보 클래스
URLConnection	56행. 웹 상에서 데이터 송수신을 위해 사용되는 HTTP를 위한 URL 연결용 클래스
URL	58행. 인터넷 상의 리소스 주소를 위한 클래스



## ● 메소드

클래스	메소드	설명
AsyncTask	abstract Result <code>doInBackground</code> (Params... params)	42행. 백그라운드 쓰레드로 실행함
	final AsyncTask<Params, Progress, Result> <code>execute</code> (Params... params)	36행. 지정된 매개변수에 대해 작업을 수행함
	void <code>onPostExecute</code> (Result result)	50행. <code>doInBackground()</code> 실행 후에 결과를 받아 실행함
URL	<code>URL</code> (String spec)	58행. spec로 주어지는 리소스 위치를 해석하여 URL 객체를 생성함
	URLConnection <code>openConnection</code> ()	59행. URL에 명시된 리소스를 연결함
HttpURLConnection	abstract void <code>disconnect</code> ()	70행. Http 서버와의 연결을 닫음
URLConnection	InputStream <code>getInputStream</code> ()	60행. URL 연결에 의한 리소스를 읽기 위한 InputStream을 반환함. HttpURLConnection은 URLConnection의 서브 클래스임
BufferedInputStream	<code>BufferedInputStream</code> (InputStream)	60행. 8192 Byte 크기를 가진 BufferedInputStream을 만듦
BufferedReader	String <code>readLine</code> ()	64행. 줄 단위로 데이터를 읽음. 읽을 문자가 없는 경우에는 null 값을 반환함
InputStreamReader	<code>InputStreamReader</code> (InputStream in, Charset charset)	61행. charset 문자 유형으로 in 에 대한 InputStreamReader 객체를 생성함

## 4 환경 설정

### 소스 | AndroidManifest.java

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
03     package="com.example.yschang.busrouteid">
04
05     <uses-permission android:name="android.permission.INTERNET" />
06
07     <application
08         android:allowBackup="true"
09         android:icon="@mipmap/ic_launcher"
10         android:label="@string/app_name"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme">
13         <activity android:name=".MainActivity">
14             <intent-filter>
15                 <action android:name="android.intent.action.MAIN" />
16
17                 <category android:name="android.intent.category.LAUNCHER" />
18             </intent-filter>
19         </activity>
20     </application>
21
22 </manifest>
```

인터넷 접속 허용

### STEP 3

### 프로젝트 실행

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



## 18.2.4

## XML 문서 파싱과 노선ID 추출

TIP

XML 문서의 파싱 방법

XML 문서의 파싱에 사용되는 세 가지 방법을 비교하면 다음과 같다. 이 책에서는 초보자가 상대적으로 이해하기 쉬운 Pull Parser를 이용하였다.

Parser	특징	장점	단점
DOM(Document Object Model) Parser	Element를 모두 Tree 구조로 메모리에 넣어 두고 사용함	메모리에 Tree구조로 정보가 들어있기 때문에 한번 파싱해 두면 아무때나 얻고 싶은 Element에 대한 정보를 얻을 수 있음	메모리의 소모가 다른 방법보다 많음
SAX(Simple API for XML) Parser	이벤트 기반의 파서로 문서의 시작과 끝, Element의 시작과 끝, Element의 내용 등 Element Tag의 이름에 따라 각각을 처리하는 메소드를 두어 파싱함	라인 단위로 파싱하기 때문에 파싱하는데 적은 메모리 소요	파싱시 그냥 지나갔던 Element의 정보를 얻고 싶으면 다시 파싱해야함
Pull Parser	SAX와 같이 이벤트 기반의 파서이지만, SAX와 달리 문서에 대한 모든 파싱을 하지 않고도 특정 부분까지의 파싱내용을 활용할 수 있음	원하는 부분을 파싱할 수 있음	SAX의 단점을 가지며 SAX보다 약간 느림

## ● 편집

### 1 액티비티 제어

40

41

@Override

2 downloadUrl 메소드 호출하여 문서 추출

42

```
protected String doInBackground(String... urls) {
```

43

```
    try {
```

44

```
        return(String)downloadUrl((String)urls[0]);
```

45

```
    } catch(IOException e) {
```

C

46

```
        return "다운로드 실패";
```

47

```
    }
```

48

```
}
```

49

50

```
protected void onPostExecute(String result) {
```

3 문서 추출 결과의 출력

51

```
    try {
```

52

```
        XmlPullParserFactory factory =
```

XmlPull Parser를 만들기 위한  
XmlPullParserFactor 객체 생성

```
            XmlPullParserFactory.newInstance();
```

53

```
        factory.setNamespaceAware(true);
```

54

```
        XmlPullParser xpp = factory.newPullParser();
```

XmlPullParserFactor에 의해  
생성될 parser를 만들 때,  
XML name space  
지원 여부를 설정함

55

Xml Pull Parser 객체 생성

56

```
        xpp.setInput(new StringReader(result));
```

57

```
        int eventType = xpp.getEventType();
```

58

58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81

```
String headerCd = "";  
String busRouteId = "";  
String busRouteNm = "";  
  
boolean bSet_headerCd = false;  
boolean bSet_busRouteId = false;  
boolean bSet_busRouteNm = false;  
  
while(eventType != XmlPullParser.END_DOCUMENT) {  
    if(eventType == XmlPullParser.START_DOCUMENT) {  
        ;  
    } else if(eventType == XmlPullParser.START_TAG) {  
        String tag_name = xpp.getName();  
        if(tag_name.equals("headerCd"))  
            bSet_headerCd = true;  
        if(tag_name.equals("busRouteId"))  
            bSet_busRouteId = true;  
        if(tag_name.equals("busRouteNm"))  
            bSet_busRouteNm = true;  
    } else if(eventType == XmlPullParser.TEXT) {  
        if(bSet_headerCd) {  
            headerCd = xpp.getText();  
            tv.append("headerCd: " + headerCd + "\n");  
        }  
    }  
}
```

데이터  
리소스(result)에 대한  
Input Stream 설정

parser가 현재 가르키고 있는  
이벤트 타입(START\_TAG,  
END\_TAG, TEXT 등)을 반환함

현재 이벤트 타입이  
END\_DOCUMENT가  
될 때까지 처리를 반복

태그 이름  
추출

태그 이름이  
<headerCd>인  
경우

이벤트 타입이  
START\_TAG인  
경우

이벤트 타입이  
TEXT인 경우

시작 태그(< >)와 마침 태그(</ >)  
사이에 있는 데이터 추출



```

82         bSet_headerCd = false;
83     }
84     if(headerCd.equals("0")) {
85         if(bSet_busRouteId) {
86             busRouteId = xpp.getText();
87             tv.append("busRouteId: " + busRouteId + "\n");
88             bSet_busRouteId = false;
89         }
90         if(bSet_busRouteNm) {
91             busRouteNm = xpp.getText();
92             tv.append("busRouteNm: " + busRouteNm + "\n");
93             bSet_busRouteNm = false;
94         }
95     }
96 } else if(eventType == XmlPullParser.END_TAG) {
97     ;
98 }
99 eventType = xpp.next();
100 }
101 } catch(Exception e) {
102     tv.setText(e.getMessage());
103 }
104 }
105 }
106

```

headerCd 태그의 값이 "0"인 경우

이벤트 타입이 END\_TAG인 경우

parser를 다음 이벤트 타입으로 이동 후,  
인식한 타입을 반환

데이터 추출 과정에서  
에러 발생 시, 에러 내용 출력

```
107 private String downloadUrl(String myurl) throws IOException {
108
109     HttpURLConnection conn = null;
110     try {
111         URL url = new URL(myurl);
112         conn =(HttpURLConnection) url.openConnection();
113         BufferedInputStream buf = new BufferedInputStream(
114                                     conn.getInputStream());
115         BufferedReader bufreader = new BufferedReader(
116                                     new InputStreamReader(buf, "utf-8"));
117         String line = null;
118         String page = "";
119         while((line = bufreader.readLine()) != null) {
120             page += line;
121         }
122         return page;
123     } finally {
124         conn.disconnect();
125     }
126 }
127 }
```

URL에 해당하는 웹문서 다운로드

# Class와 속성/메소드

## ● 클래스

클래스/인터페이스	설명
StringReader	56행. 순차적으로 String 데이터를 character로 읽는 리더기
Throwable	가상 머신에 의해 던져지는 모든 클래스의 수퍼 클래스. 서브 클래스로는 복원 가능한 Exception 클래스와 복원 불가능한 Error 클래스가 있음. IOException은 Exception의 서브클래스이며, Exception는 Throwable의 서브클래스임
XmlPullParserFactory	52행. XmlPullParser의 구현을 위해 사용됨
XmlPullParser	54행. 파싱을 정의하는 인터페이스

## ● 상수

클래스	상수	설명
XmlPullParser	static final int START_DOCUMENT	68행. XML 문서의 시작
	static final int END_DOCUMENT	67행. XML 문서의 끝
	static final int START_TAG	70행. 시작 태그가 읽혀질 때 반환됨
	static final int END_TAG	97행. 마침 태그가 읽혀질 때 반환됨
	static final int TEXT	78행. 데이터가 읽혀질 때 반환됨

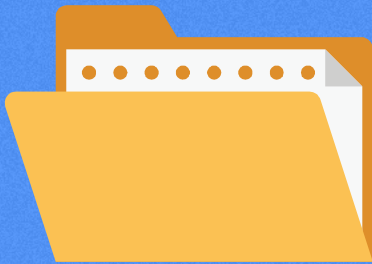


## ● 메소드

클래스	메소드	설명
Throwable	String <code>getMessage()</code>	103행. Throwable이 발생할 때 제공되는 메시지를 반환함
XmlPullParserFactory	static XmlPullParserFactory <code>newInstance()</code>	52행. XML pull parser를 만들기 위해 사용될 수 있는 PullParserFactory의 새로운 인스턴스를 만듦
	void <code>setNamespaceAware(boolean awareness)</code>	53행. parser를 만들 때, XML namespace 지원 여부를 설정함(디폴트는 false)
	XmlPullParser <code>newPullParser()</code>	54행. XML pull parser의 인스턴스를 만듦
XmlPullParser	abstract void <code>setInput(Reader in)</code>	56행. parser에 입력 소스를 설정함
	abstract int <code>getEventType()</code>	57행. 현재 이벤트 유형을 반환함(START_TAG, END_TAG, TEXT 등)
	abstract String <code>getName()</code>	71행. 태그 이름 추출
	abstract String <code>getText()</code>	80, 87, 92행. 현재 이벤트의 텍스트 콘텐츠를 반환함
	abstract int <code>next()</code>	100행. 다음 파싱 이벤트를 가져옴

**STEP 3****프로젝트 실행**

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



## 18.3 노선버스 위치 조화: Bus Position 프로젝트 (노선버스 위치)



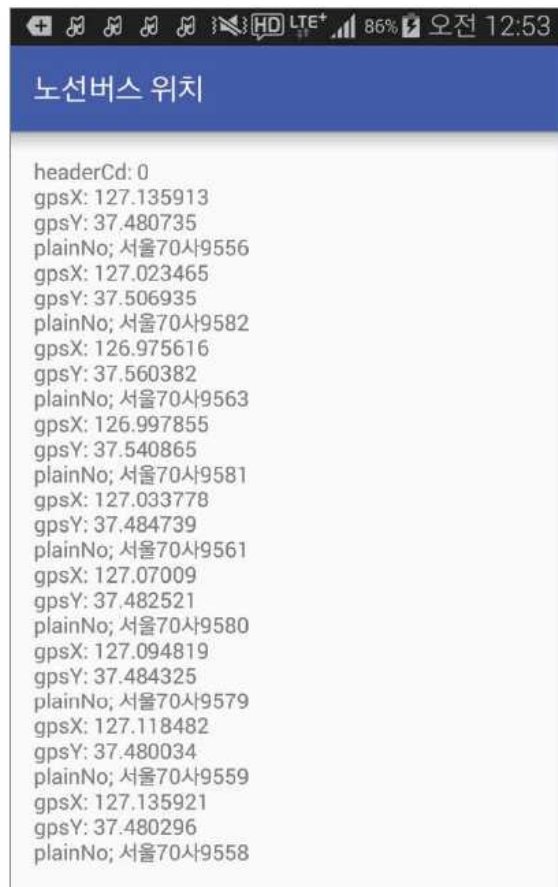


## 18.3.1 프로젝트 개요

프로젝트 개요: 노선버스의 실시간 위치 출력

Application Name: Bus Positions

어플리케이션 라벨: 노선버스위치



```
headerCd: 0
gpsX: 127.135913
gpsY: 37.480735
plainNo; 서울70사9556
gpsX: 127.023465
gpsY: 37.506935
plainNo; 서울70사9582
gpsX: 126.975616
gpsY: 37.560382
plainNo; 서울70사9563
gpsX: 126.997855
gpsY: 37.540865
plainNo; 서울70사9581
gpsX: 127.033778
gpsY: 37.484739
plainNo; 서울70사9561
gpsX: 127.07009
gpsY: 37.482521
plainNo; 서울70사9580
gpsX: 127.094819
gpsY: 37.484325
plainNo; 서울70사9579
gpsX: 127.118482
gpsY: 37.480034
plainNo; 서울70사9559
gpsX: 127.135921
gpsY: 37.480296
plainNo; 서울70사9558
```

API 실행결과:

서울 노선버스 406번에 해당하는  
노선ID를 이용한 현재 버스위치를 담고  
있는 XML 문서를 파싱한 결과

## 18.3.2 프로젝트 개발

### STEP 1 프로젝트 생성

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: <b>Bus Positions</b> Company Domain: <b>yschang.example.com</b>
③ 제품형태	<b>Phone and Tablet</b>
④ 액티비티 유형	<b>Blank Activity</b>
⑤ 파일 옵션	디폴트 값으로 설정

## STEP 2

## 파일 편집

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	• 인터넷 사용 허용
java	com.example. yschang. buspositions	MainActivity.java	• 버스위치 파악을 위한 공공 DB 오픈 API 호출 • XML 문서 파싱 및 출력
res	drawable		
	layout	activity_main.xml	• XML 문서 출력을 위한 텍스트뷰 배치
	mipmap	ic_launcher.png	
	values	colors.xml	
		dimens.xml	
		strings.xml	• 어플리케이션 라벨 수정
		styles.xml	

■ 수정 ■ 추가



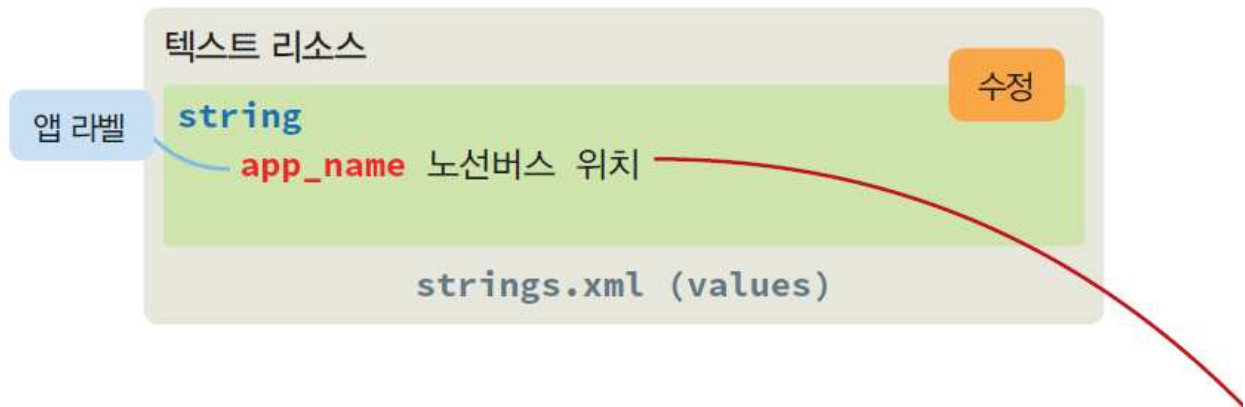
## ● 파일 간의 연관관계

**strings.xml**에는 초기치로 설정되어 있는 어플리케이션 라벨을 '노선버스 위치'로 수정한다.

**activity\_main.xml**에는 공공 DB API(노선버스 위치) 호출 후에 반환되는 XML 문서를 출력할 텍스트뷰를 배치한다.

**MainActivity.java**에는 공공 DB API(노선버스위치) 호출 후에 반환되는 XML 문서를 출력한다.

**AndroidManifest.xml**에는 인터넷 사용을 허용하도록 설정한다.



## 화면 레이아웃

`RelativeLayout`  
`TextView`

`id @+id/data`

`activity_main.xml (layout)`

수정

텍스트뷰 인식

## 액티비티 제어

`onCreate()`

`super.onCreate()`

`setContentView(R.layout.activity_main)`

`tv = findViewById(R.id.data)`

...

`strurl = "http://....."`

`new DownloadWebpageTask().`

`execute(strurl)`

`MainActivity.java (java)`

수정

컴파일/빌더

버스위치 API의 URL

URL에 해당하는 API 호출하고 반환되는  
XML 문서를 파싱하여 버스위치를 출력

## 어플리케이션 기본 정보

`uses-permission`

`name android.permission.INTERNET`

`application`

`icon @mipmap/ic_launcher`

`label @string/app_name`

`theme @style/AppTheme`

`activity`

`name MainActivity`

`AndroidManifest.xml (manifest)`

수정

인터넷 접속 허용



사용자

출력



스마트폰

## 컴파일/빌더 정보

`build.gradle(Project)`  
`build.gradle(Module app)`  
`gradle.properties`  
`settings.gradle`  
`local.properties`

(Gradle Scripts)

## ● 편집

### 1 텍스트 자원의 편집

소스 | strings.xml

```
01 <resources>
02     <string name="app_name">노선버스 위치</string>
03 </resources>
```

### 2 화면 설계

소스 | activity\_main.xml

모듈	폴더	소스 파일	편집 내용
res	layout	activity_main.xml	• 18.2절과 동일



### 3 액티비티 제어

### 소스 | MainActivity.java

```
23  @Override
24  protected void onCreate(Bundle savedInstanceState) {
25      super.onCreate(savedInstanceState);
26      setContentView(R.layout.activity_main);
27
28      tv = (TextView) findViewById(R.id.data);
29
30      String serviceUrl = "http://ws.bus.go.kr/api/rest/buspos/
                           getBusPosByRtid";
31
32      String serviceKey = "DEp3%2.....3D%3D";
33      String busRouteId = "100100063";
34      String strUrl = serviceUrl+"?ServiceKey="+serviceKey+
                           "&busRouteId="+busRouteId;
35      new DownloadWebpageTask().execute(strUrl);
36  }
37
38  private class DownloadWebpageTask extends AsyncTask<String,
                                   Void, String> {
39
40      @Override
41      protected String doInBackground(String... urls) {
42          try {
43              return (String)downloadUrl((String)urls[0]);
44          } catch (IOException e) {
45              return "다운로드 실패";
46          }
47      }
```

버스위치 추출을 위한 공공 DB API의 URL

버스위치 추출을 위한 공공 DB API 키

노선ID "402"의 버스경로ID

공공 DB API 호출을 위한 URL

URL에 해당하는 문서 추출을 위한 객체 생성

```

71     } else if(eventType == XmlPullParser.START_TAG) {
72         String tag_name = xpp.getName();
73         if(tag_name.equals("headerCd"))
74             bSet_headerCd = true;
75         if(tag_name.equals("gpsX"))
76             bSet_gpsX = true;
77         if(tag_name.equals("gpsY"))
78             bSet_gpsY = true;
79         if(tag_name.equals("plainNo"))
80             bSet_plainNo = true;
81     } else if(eventType == XmlPullParser.TEXT) {
82         if(bSet_headerCd) {
83             headerCd = xpp.getText();
84             tv.append("headerCd: " + headerCd + "\n");
85             bSet_headerCd = false;
86         }
87
88         if(headerCd.equals("0")) {
89             if(bSet_gpsX) {
90                 gpsX = xpp.getText();
91                 tv.append("gpsX: " + gpsX + "\n");
92                 bSet_gpsX = false;
93             }
94             if(bSet_gpsY) {
95                 gpsY = xpp.getText();
96                 tv.append("gpsY: " + gpsY + "\n");
97                 bSet_gpsY = false;
98             }

```

시작 태그가 <gpsX>인 경우(버스위치의 위도)

시작 태그가 <gpsY>인 경우(버스위치의 경도)

시작 태그가 <plainNo>인 경우(버스번호)

시작 태그가 <gpsY>인 경우(버스위치의 경도)

<gpsX>와 </gpsX> 사이에  
있는 데이터 추출

버스위치의 위도 출력

시작 태그가 <gpsX>인 경우(버스위치의 위도)

## 4 환경 설정

소스 | AndroidManifest.java

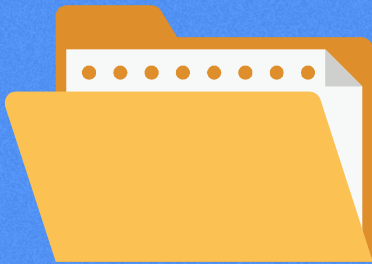
```
01 <?xml version="1.0" encoding="utf-8"?>
02 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
03     package="com.example.yschang.buspositions">
04
05     <uses-permission android:name="android.permission.INTERNET" />
06
07     <application
08         android:allowBackup="true"
09         android:icon="@mipmap/ic_launcher"
10         android:label="@string/app_name"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme">
13         <activity android:name=".MainActivity">
14             <intent-filter>
15                 <action android:name="android.intent.action.MAIN" />
16
17                 <category android:name="android.intent.category.LAUNCHER" />
18             </intent-filter>
19         </activity>
20     </application>
21
22 </manifest>
```

인터넷 접속 허용



**STEP 3****프로젝트 실행**

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



## 18.4 노선ID와 위치정보의 결 합: Seoul Bus Positions (서울버스 위치)



## 18.4.1

# 프로젝트 개요

프로젝트 개요: 노선버스의 실시간 위치 출력

Application Name: Seoul Bus Positions

어플리케이션 라벨: 서울 노선버스위치

```
서울버스 위치

===== 노선ID =====
headerCd: 0
busRouteId: 100100064
busRouteNm: 406
===== 버스 위치 =====
[1] gpsX: 127.059787
[1] gpsY: 37.476687
[1] plainNo: 서울74사4385
[2] gpsX: 126.998755
[2] gpsY: 37.48004
[2] plainNo: 서울74사4353
[3] gpsX: 126.989393
[3] gpsY: 37.494494
[3] plainNo: 서울75사1491
[4] gpsX: 126.998542
[4] gpsY: 37.510722
[4] plainNo: 서울70사9365
[5] gpsX: 126.982096
[5] gpsY: 37.56296
[5] plainNo: 서울74사3836
[6] gpsX: 126.983007
[6] gpsY: 37.573649
[6] plainNo: 서울74사4410
[7] gpsX: 126.972232
[7] gpsY: 37.559728
[7] plainNo: 서울74사4365
[8] gpsX: 126.975002
[8] gpsY: 37.559576
[8] plainNo: 서울74사4607
[9] gpsX: 126.991106
[9] gpsY: 37.529612
[9] plainNo: 서울74사3843
[10] gpsX: 126.995826
```

주어진 노선번호를 이용하여 노선ID를  
추출하고 버스위치를 파악함



## 18.4.2 프로젝트 개발

### STEP 1 프로젝트 생성

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: <b>Seoul Bus positions</b> Company Domain: <b>yschang.example.com</b>
③ 제품형태	<b>Phone and Tablet</b>
④ 액티비티 유형	<b>Blank Activity</b>
⑤ 파일 옵션	디폴트 값으로 설정

## STEP 2

## 파일 편집

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	<ul style="list-style-type: none"> <li>인터넷 사용 허용</li> </ul>
java	com.example. yschang. seoulbus positions	MainActivity.java	<ul style="list-style-type: none"> <li>노선ID추출을 위한 공공 DB 오픈 API 호출</li> <li>XML 문서 파싱 및 노선ID 출력</li> <li>버스위치 파악을 위한 공공 DB 오픈 API 호출</li> <li>XML 문서 파싱 및 버스위치 출력</li> </ul>
res	drawable		
	layout	activity_main.xml	<ul style="list-style-type: none"> <li>XML 문서 출력을 위한 텍스트뷰 배치</li> </ul>
	mipmap	ic_launcher.png	
	values	colors.xml	
		dimens.xml	
		strings.xml	<ul style="list-style-type: none"> <li>어플리케이션 라벨 수정</li> </ul>
		styles.xml	

■ 수정 ■ 추가

## ● 파일 간의 연관관계

**strings.xml**에는 초기치로 설정되어 있는 어플리케이션 라벨을 '서울버스 위치'로 수정한다.

**activity\_main.xml**에는 서울버스 위치를 출력할 텍스트뷰를 배치한다.

**MainActivity.java**에는 공공 DB에 대해 노선ID API를 먼저 호출하고, 해당 ID의 버스위치 API를 호출하여 결과를 출력한다.

**AndroidManifest.xml**에는 인터넷 사용을 허용하도록 설정한다

앱 라벨

텍스트 리소스

**string**

**app\_name** 서울버스 위치

수정

**strings.xml (values)**



## 화면 레이아웃

**RelativeLayout**  
**TextView**

**id** @+id/data

activity\_main.xml (layout)

수정

텍스트뷰 인식

## 액티비티 제어

**onCreate()**

```
super.onCreate()
setContentView(R.layout.activity_main)
tv = findViewById(R.id.data)
...
strurl = "http://....."
new DownloadWebpageTask1().
execute(strurl)
```

MainActivity.java (java)

수정

노선ID API의 URL

노선ID API를 호출하고 반환되는 XML 문서에서 노선ID를 추출하고, 버스위치 API를 호출해서 버스위치를 출력

## 어플리케이션 기본 정보

**uses-permission**

**name** android.permission.INTERNET

**application**

**icon** @mipmap/ic\_launcher

**label** @string/app\_name

**theme** @style/AppTheme

**activity**

**name** MainActivity

AndroidManifest.xml (manifest)

수정

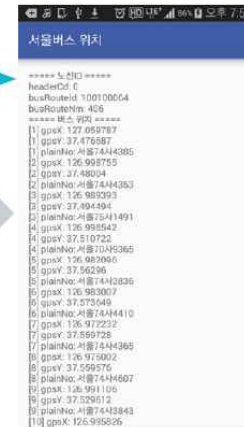
인터넷 접속 허용



사용자

출력

컴파일/빌더



스마트폰

## 컴파일/빌더 정보

```
build.gradle(Project)
build.gradle(Module app)
gradle.properties
settings.gradle
local.properties
```

(Gradle Scripts)

## ● 편집

### 1 텍스트 자원의 편집

소스 | strings.xml

```
01 <resources>
02     <string name="app_name">서울버스 위치</string>
03 </resources>
```

### 2 화면 설계

모듈	폴더	소스 파일	편집 내용
res	layout	activity_main.xml	• 18.2절과 동일

### 3 액티비티 제어

### 소스 | MainActivity.java

```
29     tv =(TextView) findViewById(R.id.data);
30
31     String serviceUrl = "http://ws.bus.go.kr/api/rest/busRouteInfo/
                               getBusRouteList";
32     String serviceKey = "DEp3%2.....3D%3D";
33     String strSrch = "406";
34     String strUrl = serviceUrl+"?ServiceKey="+serviceKey+"&strSrch=
                               "+strSrch;
35
36     DownloadWebpageTask1 task1 = new DownloadWebpageTask1();
37     task1.execute(strUrl);
38 }
39
40 private class DownloadWebpageTask1 extends AsyncTask<String, Void,
                                   String> {
41
42     @Override
43     protected String doInBackground(String... urls) {
44         try {
45             return (String)downloadUrl((String)urls[0]);
46         } catch (IOException e) {
47             return "다운로드 실패";
48         }
49     }
50
51     protected void onPostExecute(String result) {
52
53     
```

노선ID 추출을 위한 공공 DB API의 URL

노선ID 추출을 위한 공공 DB API 키

노선버스의 노선번호

공공 DB API 호출을 위한 URL

1 URL에 해당하는 문서 추출을 위한 객체 생성

2 문서 추출 객체 실행

3 downloadUrl 메소드 호출하여 문서 추출

A

4 문서 추출 결과의 출력

```
105     } catch(Exception e) {  
106         tv.setText(e.getMessage());  
107     }  
108
```

```
109     String serviceUrl = "http://ws.bus.go.kr/api/rest/buspos/  
                           getBusPosByRtid";
```

버스위치 추출을 위한 공공 DB API의 URL

```
110     String serviceKey = "DEp3%2.....3D%3D";
```

버스위치 추출을 위한 공공 DB API 키

```
111     String strUrl = serviceUrl+"?ServiceKey="+serviceKey+  
                           "&busRouteId="+busRouteId;
```

공공 DB API 호출을 위한 URL(버스경로ID 포함)

```
112     DownloadWebpageTask2 task2 = new DownloadWebpageTask2();
```

```
113     task2.execute(strUrl);
```

① URL에 해당하는  
문서 추출을 위한  
객체 생성

```
114 }  
115
```

② 문서 추출 객체 실행

```
116  
117     private String downloadUrl(String myurl) throws IOException {  
118
```



```
158     xpp.setInput(new StringReader(result));
159     int eventType = xpp.getEventType();
160
161     int count = 0;
162     while(eventType != XmlPullParser.END_DOCUMENT) {
163         if(eventType == XmlPullParser.START_DOCUMENT) {
164             ;
165         } else if(eventType == XmlPullParser.START_TAG) {
166             String tag_name = xpp.getName();
167             if(tag_name.equals("headerCd"))
168                 bSet_headerCd = true;
169             if(tag_name.equals("gpsX"))
170                 bSet_gpsX = true;
171             if(tag_name.equals("gpsY"))
172                 bSet_gpsY = true;
173             if(tag_name.equals("plainNo"))
174                 bSet_plainNo = true;
175         } else if(eventType == XmlPullParser.TEXT) {
176             if(bSet_headerCd) {
177                 headerCd = xpp.getText();
178                 // tv.append("headerCd: " + headerCd + "\n");
179                 bSet_headerCd = false;
180             }
```

시작 태그가 <gpsX>인 경우  
(버스위치의 위도)

시작 태그가 <gpsY>인 경우  
(버스위치의 경도)

시작 태그가 <plainNo>인 경우  
(버스번호)

```

186         gpsX = xpp.getText();
187         tv.append("[ " + count + " ] gpsX: " + gpsX + "\n");
188         bSet_gpsX = false;
189     }
190     if(bSet_gpsY) {
191         gpsY = xpp.getText();
192         tv.append("[ " + count + " ] gpsY: " + gpsY + "\n");
193         bSet_gpsY = false;
194     }
195     if(bSet_plainNo) {
196         plainNo = xpp.getText();
197         tv.append("[ " + count + " ] plainNo: " + plainNo +
198             "\n");
199         bSet_plainNo = false;
200     }
201     } else if(eventType == XmlPullParser.END_TAG) {
202         ;
203     }
204     eventType = xpp.next();
205 }
206 } catch(Exception e) {
207     tv.setText(e.getMessage());
208 }
209
210 }
211 }
212 }

```

시작 태그가 <gpsX>인 경우, 버스위치의 위도 추출

버스위치 위도 출력

시작 태그가 <gpsY>인 경우, 버스위치의 경도 추출

버스위치 경도 출력

시작 태그가 <plainNo>인 경우, 버스번호 추출

버스번호 출력

## 4 환경 설정

### 소스 | AndroidManifest.java

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
03     package="com.example.yschang.seoulbuspositions">
04
05     <uses-permission android:name="android.permission.INTERNET" />
06
07     <application
08         android:allowBackup="true"
09         android:icon="@mipmap/ic_launcher"
10         android:label="@string/app_name"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme">
13         <activity android:name=".MainActivity">
14             <intent-filter>
15                 <action android:name="android.intent.action.MAIN" />
```

인터넷 접속 허용



**STEP 3****프로젝트 실행**

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



**Thank you**