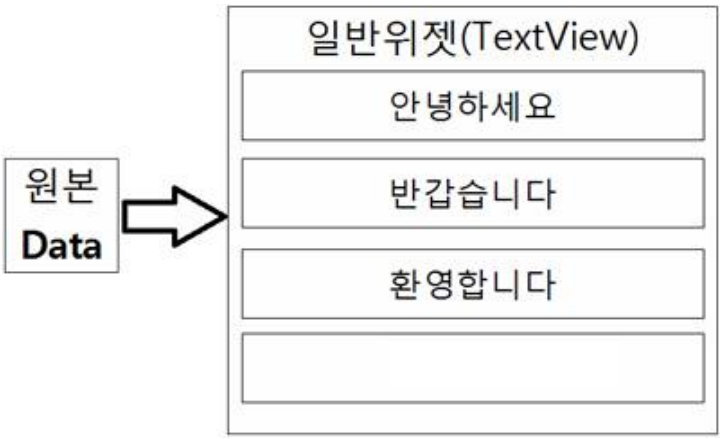
목차

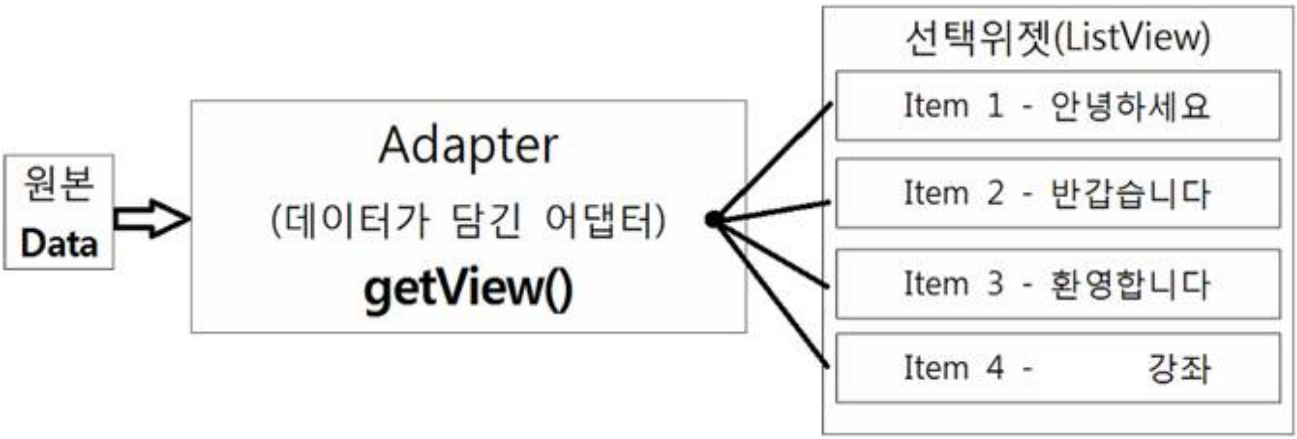
1. ListView 란?

리스트뷰는 데이터를 표시하는 뷰 그룹입니다.

리스트뷰은 직접 데이터를 설정 할 수가 없습니다. 데이터를 화면에 표시하기 위해서는 Adapter(어댑터)를 사용해야 하고, 어댑터에서 만들어주는 getView() 를 이용해 아이템을 표시합니다



- TextView 의 데이터 설정방법 – setText()으로 데이터를 직접 설정할수 있다

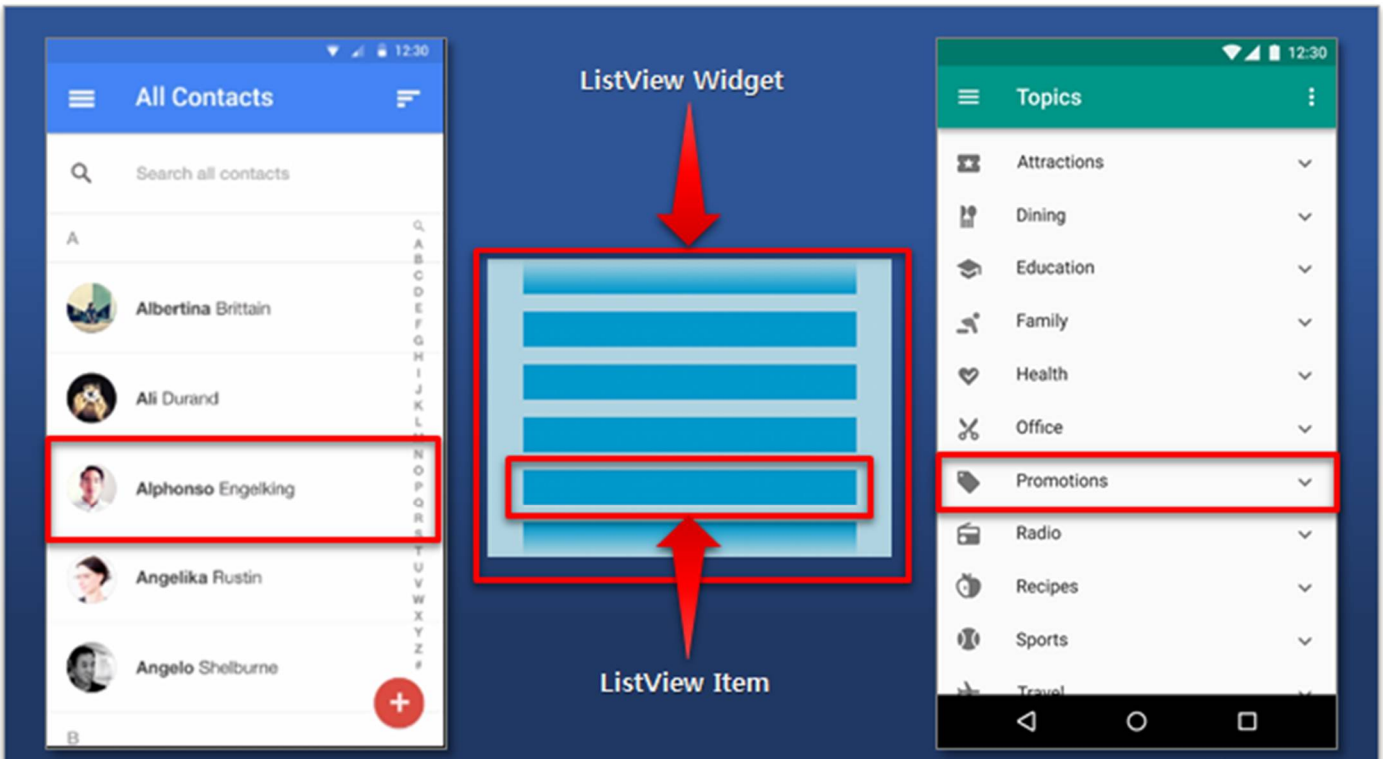


- ListView 의 데이터 설정방법 – 직접 설정이 불가능 하고 Adapter 를 이용한다

ListView 의 아이템들은 세로 방향으로 나열되며, 스크롤 기능을 사용해 ListView 의 표시 기준 위치를 이동시킬 수 있습니다.

ListView 에 표시되는 형태는 크게 2 가지로 구분할 수 있다.

- 단순 구조: 한 개의 View 만 출력하는 구조
- 복합 구조: 여러 개의 View 조합을 출력하는 구조



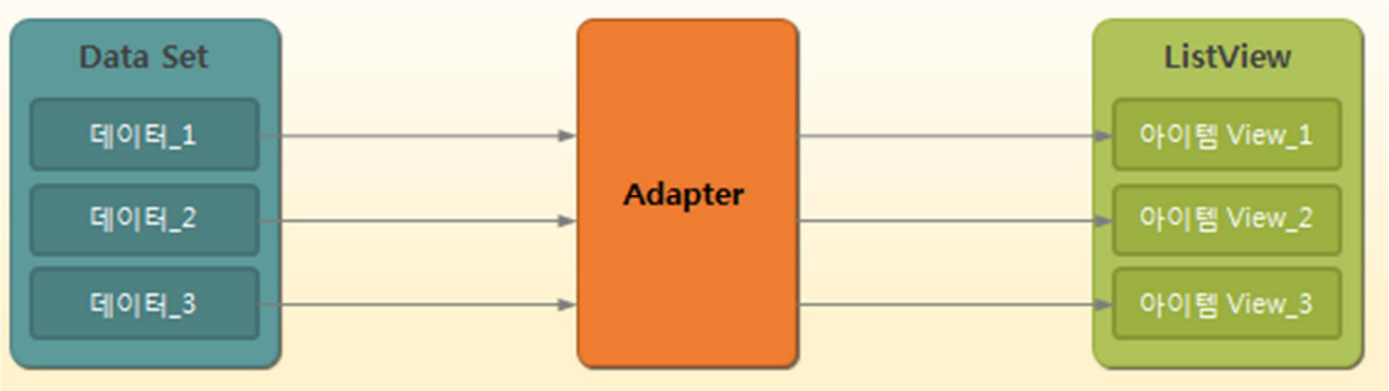
java.lang.Object
↳android.view.View
↳android.view.ViewGroup
↳android.widget.AdapterView<android.widget.ListAdapter>
↳ android.widget.AbsListView
↳ android.widget.ListView

2. Adapter 란?

✓ " 데이터와 View 를 연결하기 위해 사용하는 객체다"

"어댑터"의 의미는 "두 개의 부분을 전기적 또는 기계적으로 접속하기 위한 장치 또는 도구"를 말합니다.

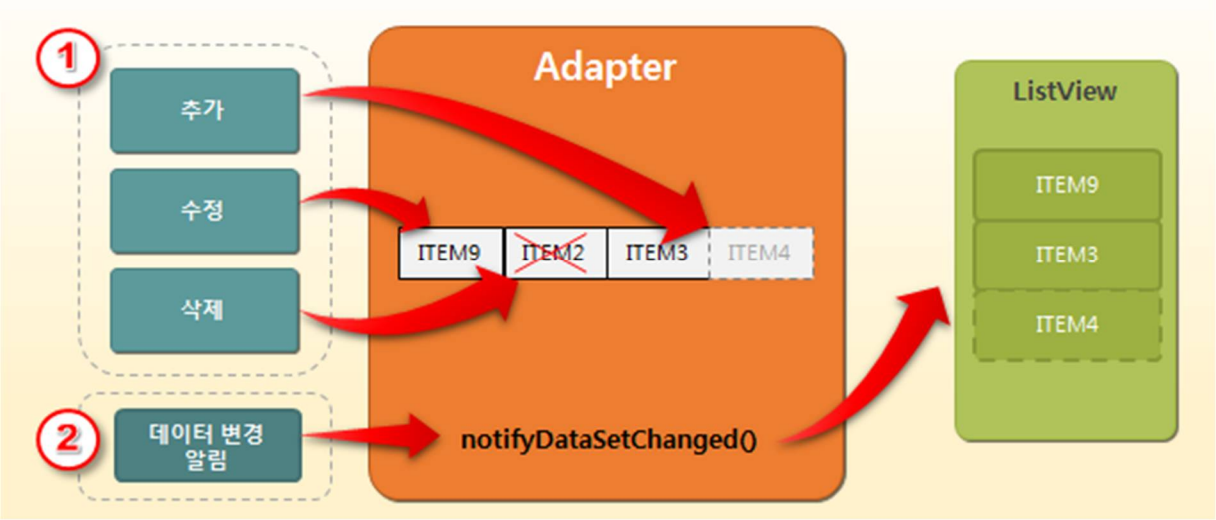
- Adapter 는 뷰(Client)와 데이터를 연결해주는 역할을 한다.
- Adapter 가 하는 역할은 사용자 데이터를 입력 받아 View 를 생성하는 것이며 Adapter 에서 생성되는 View 는 ListView 내 하나의 아이템 영역에 표시되는 것입니다.



ListView 에 아이템을 출력하기 위해서는 Adapter 를 사용해야 합니다. Adapter 에 데이터를 리스트 형태로 전달하면 getView() 메서드에서 ListView 에 표시할 View 를 반환하게 된다.

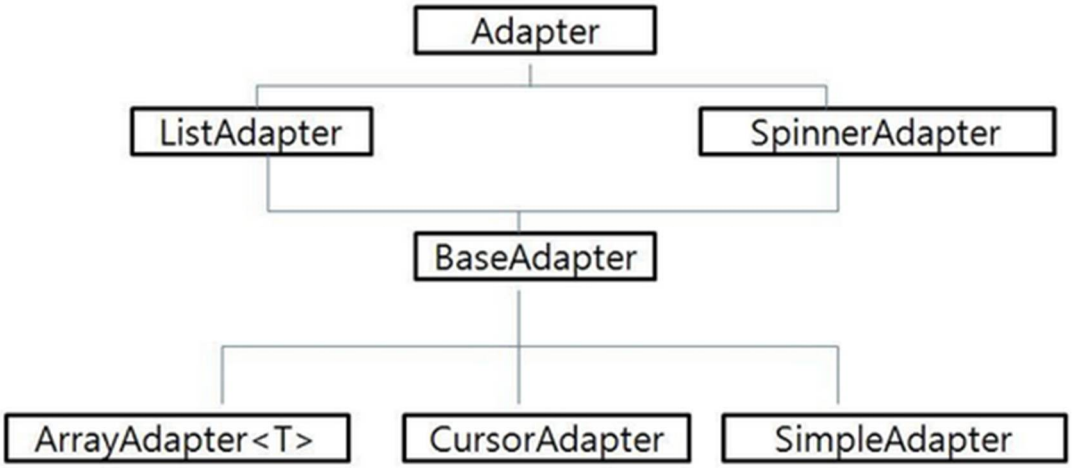
✓ notifyDataSetChanged()

ListView 의 데이터가 변경되었음을 ListView 에 알려 다시 그리도록 하면 됩니다.



2.1 Adapter 종류

Adapter 의 종류에는 ArrayAdapter, BaseAdapter, CursorAdapter, PagerAdapter, SimpleAdapter 가 있다.



- **ArrayAdapter** : array 나 java.util.List 에 저장된 data 를 위한 adapter
- BaseAdapter
- CursorAdapter : 데이터베이스에 저장된 data 를 위한 adapter

- **PagerAdaper**: FragmentPagerAdapter
- SimpleAdapter : XML 파일에 저장된 data 를 위한 adapter

2.2 Adapter 메서드

getCount()	
	데이터의 전체 갯수를 반환
getItem(int position)	
	해당 position 번째의 데이터를 반환
getItemViewType(int position)	
	뷰의 타입이 같으면 1 다르면 2 를 반환.
getView(int position, View convertView, ViewGroup parent)	
	position 번째 데이터를 가져와 inflater 를 이용하여 view 에 적용해 주는 메서드 ✓ position : 데이터 위치. ✓ convertView : 보여질 view 객체. ✓ parent : 부모 객체
add( ) addAll()	
	adapter 에 데이터 추가하는 메서드

2.3 ArrayAdapter 의 생성자

ArrayAdapter(Context context, int resource)

ArrayAdapter(Context context, int resource, T[] objects)

ArrayAdapter(Context context, int resource, List<T> objects)

ArrayAdapter(Context context, int resource, int textViewResourceId)

ArrayAdapter(Context context, int resource, int textViewResourceId, T[] objects)

ArrayAdapter(Context context, int resource, int textViewResourceId, List<T> objects)

2.4 ArrayAdapter 의 사용 예제

```
String [] list = {"111", "222"};
ArrayAdapter<String> a = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, list);
```

```
String [] list = {"111", "222"};
ArrayAdapter<String> b = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1);
b.addAll( list );
```

이렇게 하면 ArrayAdapter 에 111 과 222 라는 원본데이터가 들어가게 되고,
이 아답터를 활용해서 위젯(리스트뷰,스피너,그리드뷰)에 데이터들을 사용할 수 있다.

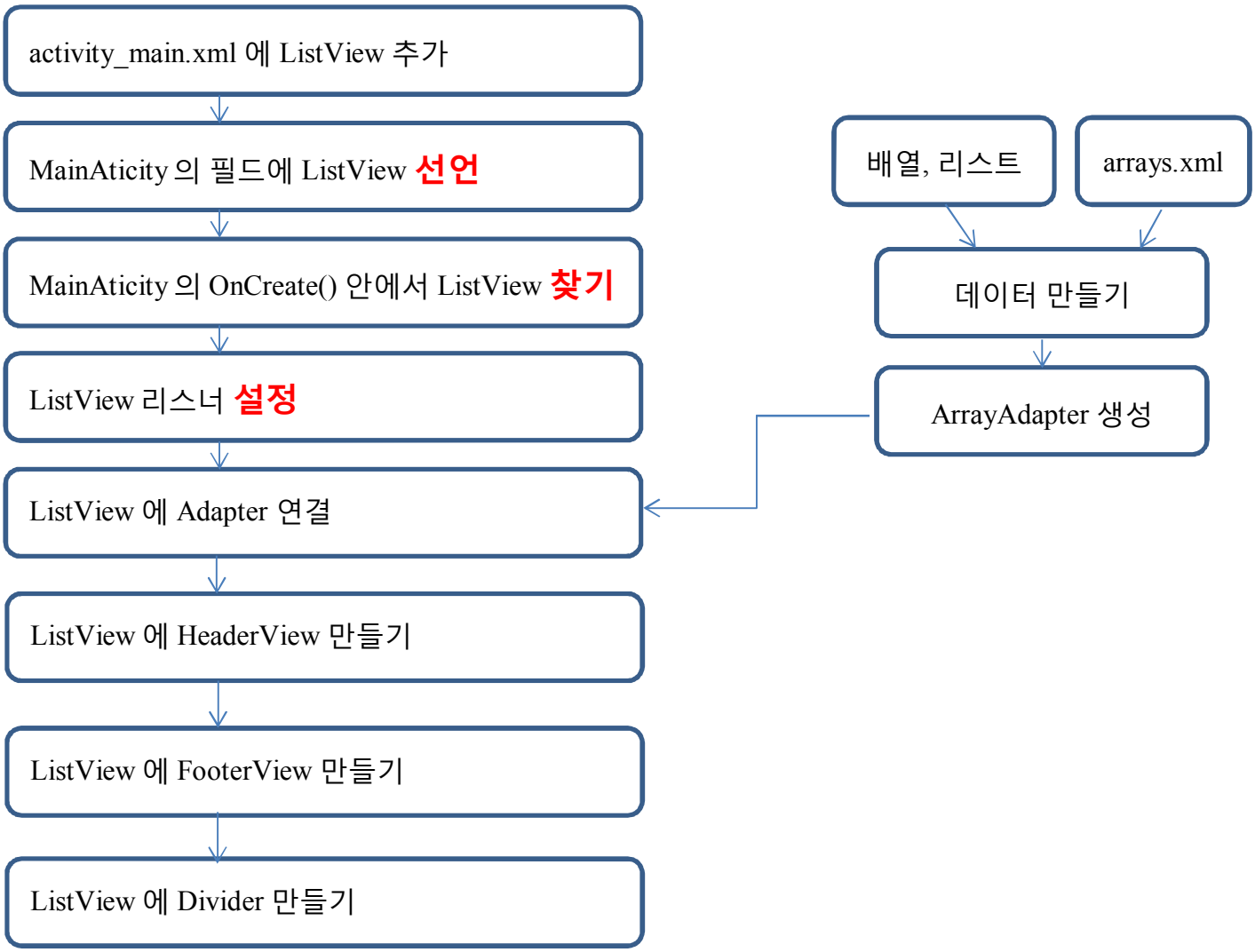
- this - 사용할 Context. 일반적으로 아답터를 포함하는 activity 의 context 가 사용됨.
- android.R.layout.simple_list_item_1 - 아이템이 표시될 view 의 resource ID.
  어떤 형식의 **TextView 리소스**를 지정하느냐에 따라 아이템의 text 형태가 바뀜.

simple_list_item_1	하나의 텍스트뷰로 구성된 레이아웃
simple_list_item_2	두개의 텍스트뷰로 구성된 레이아웃
simple_list_item_checked	오른쪽에 체크 표시가 나타남
simple_list_item_single_choice	오른쪽에 라디오 버튼이 나타남
simple_list_item_multiple_choice	오른쪽에 체크 버튼이 나타남

- list - Selection 위젯에 나열될 array 나 java.util.List 형식의 data.

ArrayAdapter 는 공급된 data 의 요소를 toString() 메소드로 String 화하여 Text 형태로 표현된다.

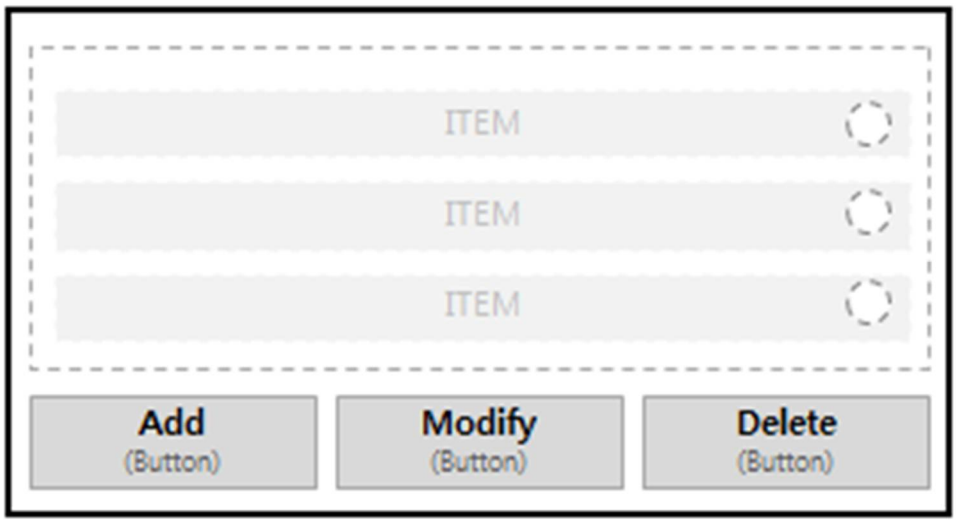
3. ListView 의 작업 순서



4. Simple ListView 만들기

ListView 에 아이템 목록을 출력하고 아이템을 추가, 수정 또는 삭제하는 방법에 대해 알아보겠습니다.

예제 화면은 다음과 같이 ListView 에 세 개의 Button 을 추가하여 구성합니다.



4.1 디자인에 ListView 추가

"activity_main.xml" 파일에 ListView 를 추가합니다.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

<ListView
    android:id="@+id/listview1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:choiceMode="singleChoice" />

</LinearLayout>
```

4.2 ListView 찾기

```
public class MainActivity extends AppCompatActivity {

    // 위젯 선언
    private ListView listview = null;

    @Override
```

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

        // 위젯 찾기
        listview = findViewById(R.id.listview1);

        // 위젯 설정
        listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // get TextView's Text.
                String strText = (String) parent.getItemAtPosition(position) ;
            }
        });
    }
}
```

4.3 ListView 핸들러 설정

리스트 아이템 자체를 클릭하도록 ListView 에 Click 이벤트에 대한 리스너를 설정해주면 됩니다. onItemClick() 함수에서 ListView 아이템인 TextView 의 텍스트를 가져오려면 parent.getItemAtPosition() 함수를 사용하면 됩니다.

```
public class MainActivity extends AppCompatActivity {

    // 위젯 선언
    private ListView listview = null;

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

        // 위젯 찾기
        listview = findViewById(R.id.listview1);

        // 위젯 설정
        listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // get TextView's Text.
                String strText = (String) parent.getItemAtPosition(position) ;
            }
        });
    }
}
```

4.4 데이터 정의

이제 ListView 가 표시될 위치를 결정하였으니 ListView 에 어떤 내용을 보여줄 것인지 정의해야 합니다. ListView 에 문자열만 표시할 것이므로 string 타입의 배열을 선언합니다.

```
public class MainActivity extends AppCompatActivity {

    private String[] LIST_MENU = {"LIST1", "LIST2", "LIST3"} ;

}
```

4.5 Adapter 생성

사용자 데이터가 준비되었으니 해당 데이터를 입력 받아 View 에 연결해줄 Adapter 를 생성해야겠죠. 여기서는 ArrayAdapter 를 사용하겠습니다.

```
public class MainActivity extends AppCompatActivity {

    // 위젯 선언
    private ListView listview = null;

    private String[] LIST_MENU = {"LIST1", "LIST2", "LIST3"} ;
    private ArrayAdapter<String>  adapter = null;

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

        // 위젯 찾기
        listview = findViewById(R.id.listview1);

        // 위젯 설정
        listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // get TextView's Text.
                String strText = (String) parent.getItemAtPosition(position) ;
            }
        });

        // 어댑터 만들기
        adapter = new ArrayAdapter<String>(MainActivity.this,
android.R.layout.simple_list_item_1, LIST_MENU);
}
```

!코드 설명

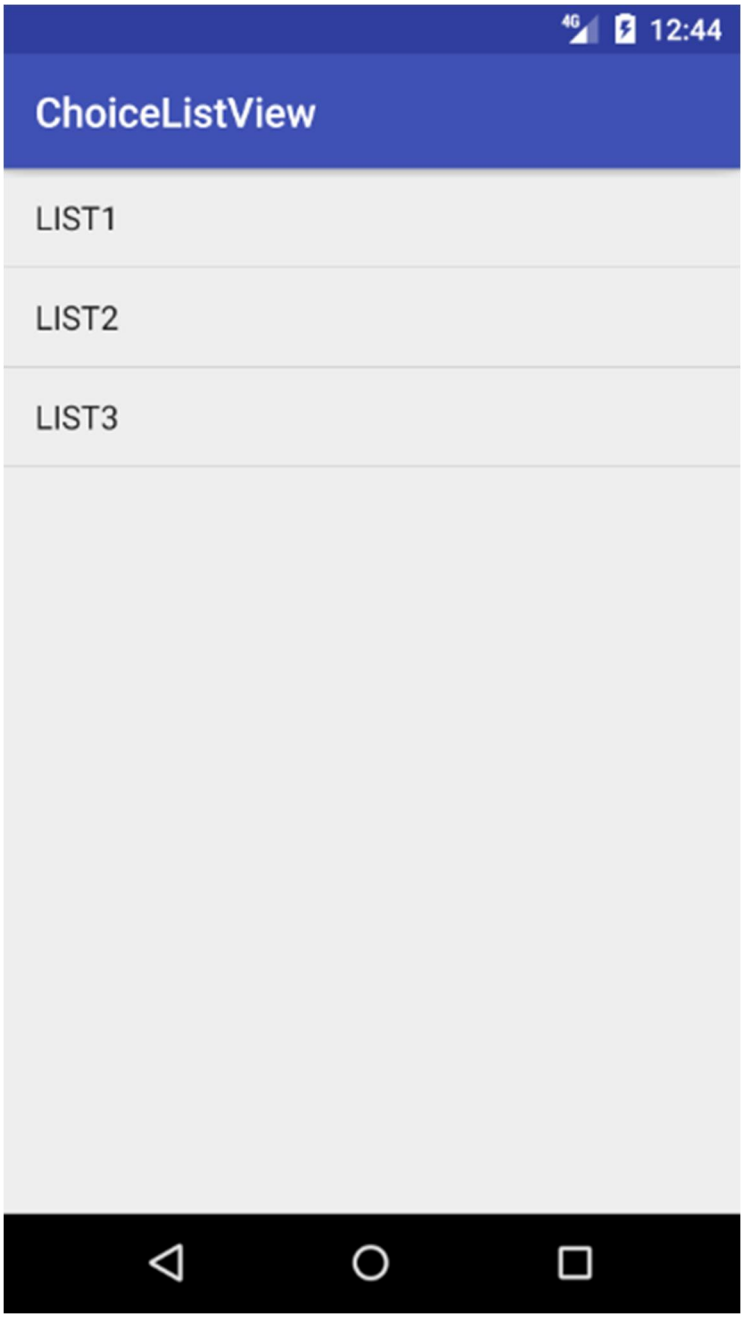
위의 코드에서 사용한 ArrayAdapter 의 생성자는 3 개의 파라미터를 가집니다.

ArrayAdapter(Context context, int resource, T[] objects)	
context	안드로이드 시스템에서 제공되는 어플리케이션 전역 환경 정보에 대한 인터페이스. (Activity 를 통해 사용 가능)
resource	View 로 매핑될 Resource Id. "android.R.layout.simple_list_item_1"은 TextView 위젯으로 구성된 ListView 아이템 리소스 Id.
objects	배열로 선언된 사용자 데이터.

4.6 ListView 와 Adapter 연결

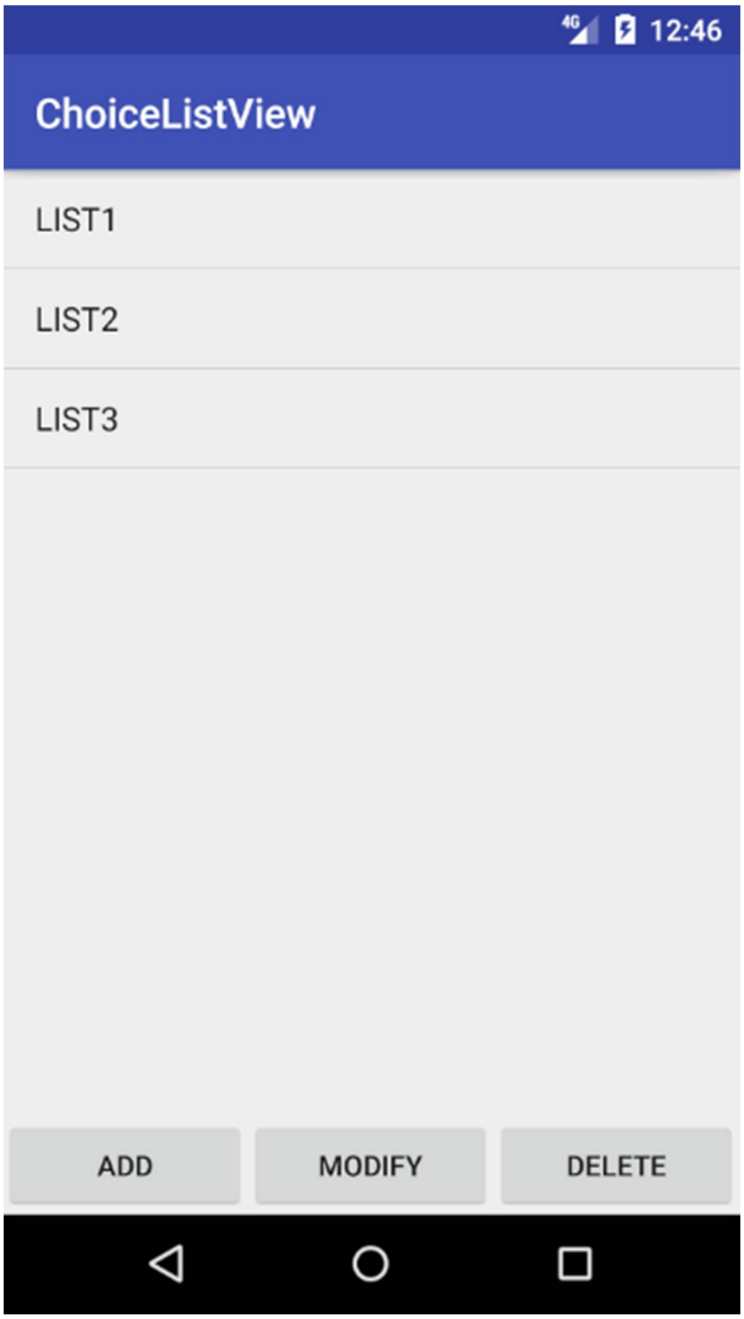
```
public class MainActivity extends AppCompatActivity {

    // 위젯 선언
    private ListView listview = null;

    private String[] LIST_MENU = {"LIST1", "LIST2", "LIST3"} ;
    private ArrayAdapter<String>  adapter = null;

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

        // 위젯 찾기
        listview = findViewById(R.id.listview1);

        // 위젯 설정
        listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // get TextView's Text.
                String strText = (String) parent.getItemAtPosition(position) ;
            }
        });

        // 어댑터 만들기
        adapter = new ArrayAdapter<String>(MainActivity.this,
android.R.layout.simple_list_item_1, LIST_MENU);

        // 어댑터와 리스트뷰 연결
        listview.setAdapter( adapter );
    }
}
```

4.7 실행 화면



4.8 디자인에 Button 추가

ListView 및 Button 이 추가된 Layout 리소스 파일을 작성합니다. 이 때 ListView 에 choiceMode 속성을 사용하였습니다. choiceMode 를 지정하면 ListView 아이템을 선택 할 수 있는 기능을 사용할 수 있습니다.



● [STEP-1] "activity_main.xml" - MainActivity 에 Button 추가.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ListView
        android:id="@+id/listview1"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:choiceMode="singleChoice" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <Button
            android:id="@+id/add"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Add" />
        <Button
            android:id="@+id/modify"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Modify" />
        <Button
            android:id="@+id/delete"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Delete" />
    </LinearLayout>

</LinearLayout>
```

MainActivity 의 onCreate() 에서 ArrayList 와 ListView 를 생성합니다. 이 때 choiceMode 를 지원하는 "android.R.layout.simple_list_item_single_

oice" 리소스를 사용한 것에 주의하세요.

"android.R.layout.simple_list_item_single_choice"를 사용하면 ListView 아이템에 TextView 와 Radio Button 을 가진 View 가 표시됩니다.

- onCreate() 에서 ListView 및 Adapter 생성.

```
public class MainActivity extends AppCompatActivity {

 ArrayList<String> items = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
       // ... 코드 계속

        // 빈 데이터 리스트 생성.
        items = new ArrayList<String>() ;
```

```
        // ArrayAdapter 생성. 아이템 View 를 선택(single choice)가능하도록 만듦.
        ArrayAdapter adapter = new ArrayAdapter(this,
                    android.R.layout.simple_list_item_single_choice, items) ;

        // listview 생성 및 adapter 지정.
        final ListView listview = (ListView) findViewById(R.id.listview1) ;
        listview.setAdapter(adapter) ;

        // 코드 계속 ...
    }
}
```

4.8.1 "추가"에 대한 이벤트 처리.

아이템을 추가하기 위해 ListView 의 Adapter 에서 리스트를 얻어온 다음, 원하는 위치에 데이터를 추가하고 ListView 를 갱신합니다. 예제에서는 가장 마지막에 아이템을 추가합니다.

```
public class MainActivity extends AppCompatActivity {

 ArrayList<String> items = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // ... 코드 이어서

        // add button 에 대한 이벤트 처리.
        Button addButton = (Button)findViewById(R.id.add) ;
        addButton.setOnClickListener(new Button.OnClickListener() {
            public void onClick(View v) {
                int count;
                count = adapter.getCount();

                // 아이템 추가.
                items.add("LIST" + Integer.toString(count + 1));

                // listview 갱신
                adapter.notifyDataSetChanged();
            }
        }) ;
    }
}
```

4.8.2 "수정"에 대한 이벤트 처리.

아이템을 수정하기 위한 절차 또한 추가하는 것과 동일하게 작성합니다. 즉, Adapter 에 지정한 데이터 리스트에서 원하는 위치의 데이터를 수정하고 ListView 를 갱신합니다. 현재 선택된 아이템 위치를 얻어오기 위해서는 ListView 의 getCheckedItemPosition() 함수를 사용하면 됩니다.

[STEP-4] "MainActivity.java" - [STEP-3] 아래에 "수정" 버튼에 대한 핸들러 작성.

```
        Button modifyButton = (Button)findViewById(R.id.modify) ;
        modifyButton.setOnClickListener(new Button.OnClickListener() {
            public void onClick(View v) {
```

```
                int count, checked ;
                count = adapter.getCount() ;

                if (count > 0) {
                    // 현재 선택된 아이템의 position 획득.
                    checked = listview.getCheckedItemPosition();
                    if (checked > -1 && checked < count) {
                        // 아이템 수정
                        items.set(checked, Integer.toString(checked+1) + "번 수정") ;

                        // listview 갱신
                        adapter.notifyDataSetChanged();
                    }
                }
            }
        }) ;
}
```

4.8.3 "삭제"에 대한 이벤트 처리.

삭제하는 방법도 추가 및 수정 방법과 동일합니다. 리스트에서 데이터를 삭제하고 ListView 를 갱신합니다.

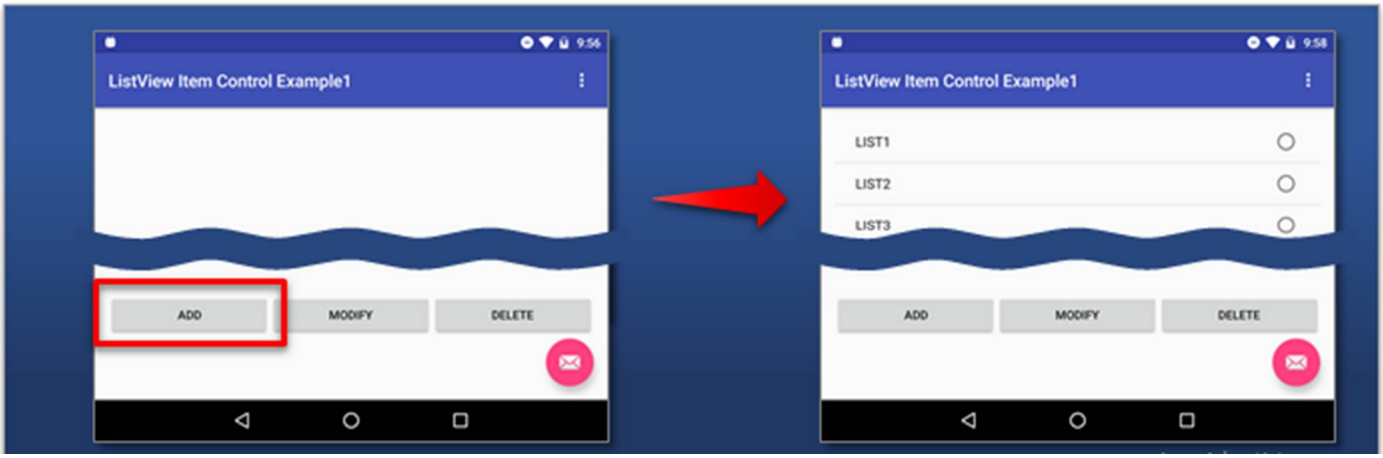
[STEP-5] "MainActivity.java" - [STEP-3] 아래에 "수정" 버튼에 대한 핸들러 작성.

```
        Button deleteButton = (Button)findViewById(R.id.delete) ;
        deleteButton.setOnClickListener(new Button.OnClickListener() {
            public void onClick(View v) {
                int count, checked ;
                count = adapter.getCount() ;

                if (count > 0) {
                    // 현재 선택된 아이템의 position 획득.
                    checked = listview.getCheckedItemPosition();

                    if (checked > -1 && checked < count) {
                        // 아이템 삭제
                        items.remove(checked) ;

                        // listview 선택 초기화.
                        listview.clearChoices();

                        // listview 갱신.
                        adapter.notifyDataSetChanged();
                    }
                }
            }
        }) ;
}
```

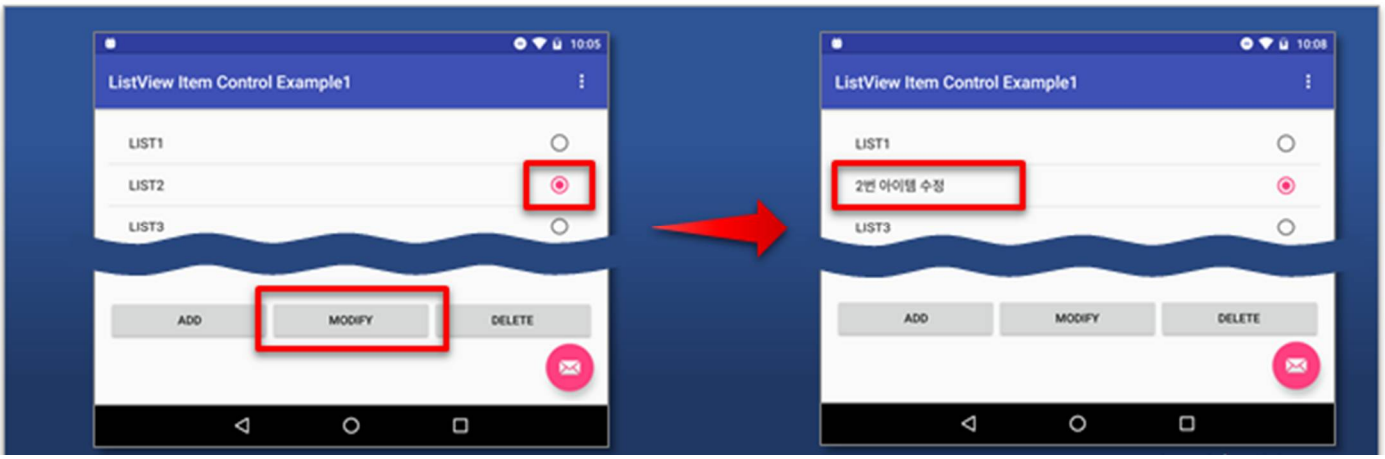
4.9 예제 실행 화면

아래는 예제를 실행한 결과 화면입니다.

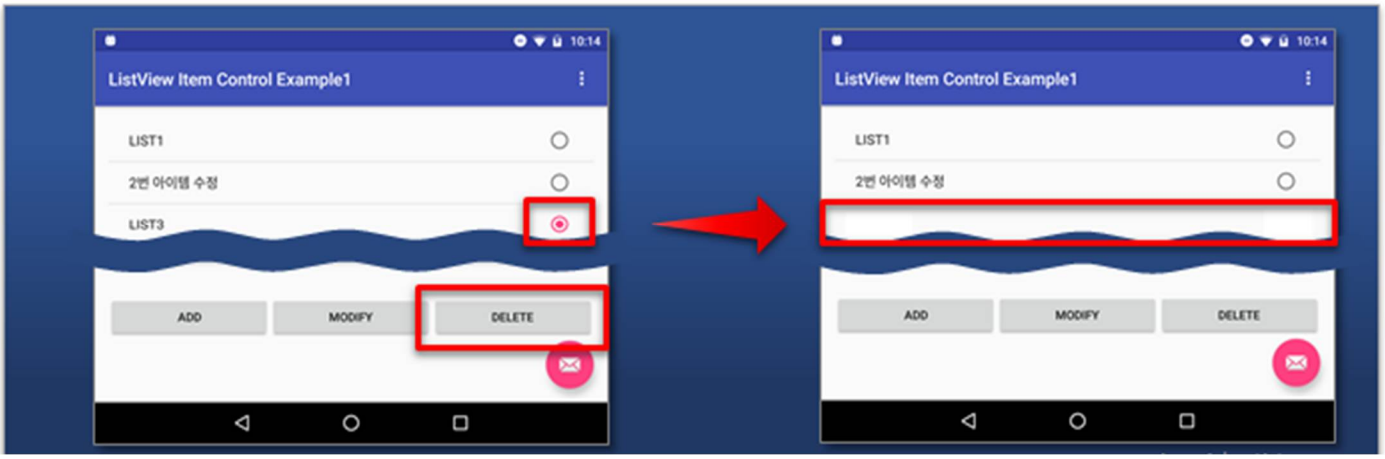
"Add" 버튼을 클릭했을 때 출력 결과입니다.



아이템을 선택하고 "Modify" 버튼을 클릭했을 때 출력 결과입니다.



아이템을 선택한 다음 "Delete" 버튼을 클릭했을 때 출력 결과입니다.



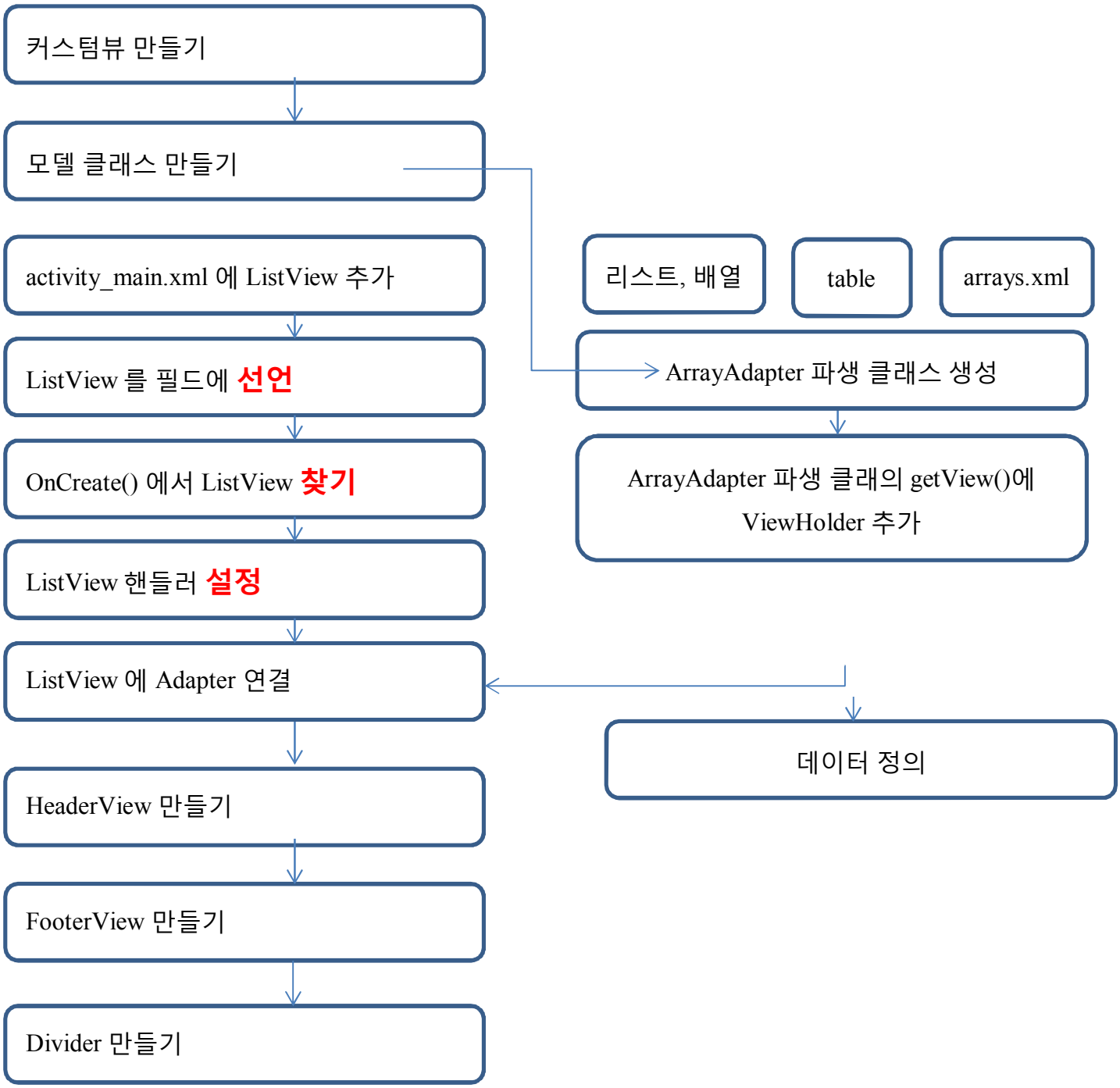
5. Custom ListView 만들기

TextView 만을 포함하는 기본 ListView 를 확장하여 TextView, ImageView, Button 등을 기본 View 로 사용하는 Custom ListView 예제를 만들어 보겠읍니다.

리스트 뷰를 사용시 개발자가 원하는 대로 사용하는 방법입니다. 리스트뷰 이미지도 넣고 싶고 타이틀과 작은 타이틀을 같이 넣고 싶을때는 어떻게 해야할까요?

리스트의 각 아이템에 보여주고 싶은 모양을 아래 그림과 같이 바꾸려고 합니다.

5.1 Custom ListView 작업 순서



5.2 스피너 데이터 정의

✓ res/values/arrays.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="spinneritems">
        <item>이름</item>
        <item>학번</item>
        <item>학과</item>
    </string-array>
</resources>
```

5.3 커스텀뷰 만들기

리스트에서 보여지게 될 Item 에 대한 디자인으로 소스 구성은 아래와 같습니다.

아이템에 대한 화면 틀을 만들어 보겠읍니다.

- view_student.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:app="http://schemas.android.com/apk/res-auto"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:padding="10dp">

    <TextView android:id="@+id/text_name"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:textSize="20dp"
              android:text="이름"/>

    <TextView android:id="@+id/text_number"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:layout_toRightOf="@id/text_name"
              android:layout_alignParentRight="true"
              android:textSize="20dp"
              android:gravity="right"
              android:text="학번"/>

    <TextView android:id="@+id/text_department"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:layout_below="@id/text_name"
              android:text="학과" />

</RelativeLayout>
```

5.4 ModelStudent 클래스 만들기

각 아이템에 들어갈 정보를 저장할 클래스를 만들겠습니다.

모델 클래스는 Binder 를 통해 다른 Activity 로 데이터를 넘길 수 있도록 Parcelable 을 구현해야 한다.

```
public class ModelStudent {

    private String name;
    private String number;
    private String department;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getNumber() {
        return number;
    }

    public void setNumber(String number) {
        this.number = number;
    }

    public String getDepartment() {
        return department;
    }

    public void setDepartment(String department) {
        this.department = department;
    }

    @Override
    public String toString() {
        return "ModelStudent{" +
                "name='" + name + '\'' +
                ", number='" + number + '\'' +
                ", department='" + department + '\'' +
                '}';
    }

    public ModelStudent() {
    }

    public ModelStudent(String name, String number, String department) {
        this.name       = name;
        this.number     = number;
        this.department = department;
    }
```

```
// Name 으로 정렬: comparator
public static class NameCompare implements Comparator<ModelStudent> {
    // mode == 1 : 오름차순 정렬
    // mode == -1 : 내림차순 정렬
    private int mode = 1;

    public void setMode(boolean desc) {
        if( desc == true )
            this.mode = -1; // 내림차순 정렬
        else
            this.mode =  1; // 오름차순 정렬
    }

    public NameCompare() {
        mode = 1; //
    }

    public NameCompare(boolean desc) {
        this.setMode( desc );
    }

    // + 일 때 : 바꾼다.
    // 0 일 때 : 바꾸지 않는다.
    // - 일 때 : 바꾸지 않는다.
    @Override
    public int compare(ModelStudent o1, ModelStudent o2) {
        return o1.getName().compareTo( o2.getName() )*mode;
    }
}

// Number 로 정렬: comparator
public static class NumberCompare implements Comparator<ModelStudent> {
    // mode == 1 : 오름차순 정렬
    // mode == -1 : 내림차순 정렬
    private int mode = 1;

    public void setMode(boolean desc) {
        if( desc == true )
            this.mode = -1; // 내림차순 정렬
        else
            this.mode =  1; // 오름차순 정렬
    }

    public NumberCompare() {
        this.mode = 1; //
    }

    public NumberCompare(boolean desc) {
        this.setMode(desc);
    }

    // + 일 때 : 바꾼다.
    // 0 일 때 : 바꾸지 않는다.
```

```
        // - 일 때 : 바꾸지 않는다.
        @Override
        public int compare(ModelStudent o1, ModelStudent o2) {
            return o1.getNumber().compareTo( o2.getNumber() )*mode;
        }
    }

    // Department 로 정렬: comparator
    public static class DeptCompare implements Comparator<ModelStudent> {
        // mode == 1 : 오름차순 정렬
        // mode == -1 : 내림차순 정렬
        private int mode = 1;

        public void setMode(boolean desc) {
            if( desc == true )
                this.mode = -1; // 내림차순 정렬
            else
                this.mode =  1; // 오름차순 정렬
        }
        public DeptCompare() {
            this.mode = 1; //
        }

        public DeptCompare(boolean desc) {
            this.setMode( desc );
        }

        // + 일 때 : 바꾼다.
        // 0 일 때 : 바꾸지 않는다.
        // - 일 때 : 바꾸지 않는다.
        @Override
        public int compare(ModelStudent o1, ModelStudent o2) {
            return o1.getDepartment().compareTo( o2.getDepartment() )*mode;
        }
    }


    // 검색: predicate
    /* Predicate 를 사용하기 위해서는 build.gradle 에
     * compile 'commons-collections:commons-collections:3.+'
     * 를 추가해야 한다.
     */
    public static class MyPredicate implements Predicate {

        private String fieldName;
        private Object value;

        public MyPredicate(String fieldName, Object value) {
            this.fieldName = fieldName;
            this.value = value;
        }

        @Override
```

```
        public boolean evaluate(Object object) {
            if( fieldName.equals( "name" ) ) {
                return ((ModelStudent)object).getName().contains( value.toString() );
            }
            else if( fieldName.equals("number") ) {
                return ((ModelStudent)object).getNumber().contains( value.toString() );
            }
            else if( fieldName.equals("department") ){
                return ((ModelStudent)object).getDepartment().contains( value.toString() );
            }
            else {
                return false;
            }
        }
    }
}
```

5.5 AdapterStudent 클래스 만들기

AdapterStudent 는 ArrayAdapter 를 상속받아서 만들어지는 파생 클래스이다.
어댑터를 최적화 하기 위해 ViewHolder 를 적용해야 한다.

- AdapterStudent.java

```
import android.content.*;
import android.support.annotation.*;
import android.view.LayoutInflater;
import android.view.*;
import android.widget.*;
import java.util.List;

public class AdapterStudent extends ArrayAdapter<ModelStudent> {

    private Context context;

    public AdapterStudent(@NonNull Context context, int resource, @NonNull List<ModelStudent>
objects) {
        super(context, resource, objects);
        this.context = context;
    }

    /**
     * ViewHolder 란 커스텀뷰의 위젯을 닫는 클래스다
     */
    private class ViewHolder {
        TextView name       ;
        TextView number     ;
        TextView department ;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {

        View itemLayout = LayoutInflater.from( context )
                                        .inflate(R.layout.view_student, parent, false );

        ViewHolder holder = (ViewHolder) itemLayout.getTag();

        if( holder == null ) {
            holder = new ViewHolder();

            holder.name       = itemLayout.findViewById(R.id.text_name      );
            holder.number     = itemLayout.findViewById(R.id.text_number    );
            holder.department = itemLayout.findViewById(R.id.text_department);

            itemLayout.setTag( holder );
        }

        holder.name      .setText( getItem(position).getName      () );
```

```
        holder.number    .setText( getItem(position).getNumber    () );
        holder.department.setText( getItem(position).getDepartment() );

        return itemLayout;
    }
}
```

5.6 activity_main.xml 에 listView 추가

✓ res/layout/activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <EditText android:id="@+id/edit_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="이름"/>

        <EditText android:id="@+id/edit_number"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="학번"/>

        <EditText android:id="@+id/edit_department"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:hint="학과"/>

        <Button android:id="@+id/btn_add"
            android:layout_width="80dp"
            android:layout_height="wrap_content"
            android:text="추가"/>

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Spinner
            android:id="@+id/spinner"
            android:layout_width="15dp"
            android:layout_height="match_parent"
            android:layout_weight="1"/>

        <EditText android:id="@+id/edit_item"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
```

```
            android:layout_weight="1"
            android:hint="아이템"/>

    <Button android:id="@+id/btn_search"
            android:layout_width="80dp"
            android:layout_height="wrap_content"
            android:text="검색"/>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView"
        android:layout_width="50dp"
        android:layout_height="wrap_content"
        android:text="정렬"/>

    <RadioGroup
        android:id="@+id/rgpAscDesc"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:gravity="center_vertical"
        android:orientation="horizontal">

        <RadioButton
            android:id="@+id/rdoAsc"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="오름차순"/>

        <RadioButton
            android:id="@+id/rdoDesc"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="내림차순"/>
    </RadioGroup>

    <Button android:id="@+id/btn_sort"
            android:layout_width="80dp"
            android:layout_height="wrap_content"
            android:text="정렬"/>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
```

```
        <Button android:id="@+id/btn_del_check"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="선택 삭제"/>

        <Button android:id="@+id/btn_del_all"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="전체 삭제"/>

        <Button android:id="@+id/btn_init"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="초기화"/>

    </LinearLayout>

    <ListView
        android:id="@+id/list_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</LinearLayout>
```

5.7 MainActivity 작성

- ✓ ListView 와 Adapter 연결
- ✓ ListView 핸들러 설정
- ✓ Header 추가
- ✓ Footer 추가
- ✓ Divider 추가

```
public class MainActivity extends AppCompatActivity {

    private ListView             listview1;
    private AdapterStudent       adapterListview;
    private List<ModelStudent>   data;

    // 위젯 선언
    private Spinner  spinner         = null;

    private EditText edit_name       = null;
    private EditText edit_number     = null;
    private EditText edit_department = null;
    private EditText edit_item       = null;

    private Button   btn_add         = null;
    private Button   btn_search      = null;
    private Button   btn_sort        = null;
    private Button   btn_del_all     = null;
    private Button   btn_init        = null;

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

        // 위젯 찾기
        listview1  = findViewById(R.id.list_view);

        edit_name       = findViewById(R.id.edit_name       );
        edit_number     = findViewById(R.id.edit_number     );
        edit_department = findViewById(R.id.edit_department );
        edit_item       = findViewById(R.id.edit_item       );

        btn_add         = findViewById(R.id.btn_add         );
        btn_search      = findViewById(R.id.btn_search      );
        btn_sort        = findViewById(R.id.btn_sort        );
        btn_del_all     = findViewById(R.id.btn_del_all     );
        btn_init        = findViewById(R.id.btn_init        );

        // 1. 버튼 핸들러 설정
        ButtonHanlder buttonHandler = new ButtonHanlder();
        btn_add    .setOnClickListener( buttonHandler );
        btn_search .setOnClickListener( buttonHandler );
        btn_sort   .setOnClickListener( buttonHandler );
        btn_del_all.setOnClickListener( buttonHandler );
```

```
        btn_init   .setOnClickListener( buttonHandler );

        // 2. ListView 핸들러 설정
        ListViewHandler lvHanlder = new ListViewHandler();
        listview1.setOnItemLongClickListener( lvHanlder );
        listview1.setOnItemClickListener    ( lvHanlder );

        // 데이터 만들기
        data = new ArrayList<>();
        for(int i=0; i<10; i++){
            ModelStudent student  = new ModelStudent();
            student.setName("name " + i );
            student.setNumber("number " + i );
            student.setDepartment(i+"-"+i);
            data.add(student);
        }

        // adapterListview 만들기
        adapterListview = new AdapterStudent( MainActivity.this, R.layout.view_student, data );

        // listview 와 adapterListview 연결.
        listview1.setAdapter(adapterListview);

        // 리스트뷰에 머리 아이템 추가
        View headerView = getLayoutInflater().inflate( R.layout.list_view_header, null);
        TextView headerTitle = (TextView)headerView.findViewById( R.id.header_text );
        headerTitle.setText( "리스트의 시작입니다." );
        listview1.addHeaderView( headerView, null, false );

        // 리스트뷰에 꼬리 아이템 추가
        View footerView = getLayoutInflater().inflate( R.layout.list_view_footer, null );
        TextView footerTitle = (TextView)footerView.findViewById( R.id.footer_text );
        footerTitle.setText( "로딩 중..." );
        listview1.addFooterView( footerView, null, false );

        // 리스트뷰에 구분선 추가
        listview1.setDivider( new ColorDrawable(Color.GRAY) );
        listview1.setDividerHeight( 3 );

        // 3. 스피너 설정
        // 3-1. Spinner 찾기
        // 3-2. Spinner 데이터 만들기
        // 3-3. Spinner 어댑터 생성
        // 3-4. Spinner 에 어댑터 연결
        spinner = findViewById(R.id.spinner );
        String [] spinneritems = getResources().getStringArray(R.array.spinneritems);
        ArrayAdapter<String> adapterSpinner = new ArrayAdapter<String>(MainActivity.this
                                    , android.R.layout.simple_spinner_item, spinneritems );

        spinner.setAdapter( adapterSpinner );
    }

    private class ButtonHanlder implements View.OnClickListener {
```

```
        @Override
        public void onClick(View v) {

            String searchitem  = "";
            String searchvalue = "";

            switch (v.getId() ){
                case R.id.btn_add      :
                    ModelStudent s = new ModelStudent();
                    s.setDepartment( edit_name      .getText().toString() );
                    s.setNumber    ( edit_number    .getText().toString() );
                    s.setName      ( edit_department.getText().toString() );

                    //
                    data.add( s );

                    // 데이터 변경 알림
                    adapterListview.notifyDataSetChanged();

                    // listview 의 마지막 레코드로 이동.
                    listview1.smoothScrollToPosition( data.size()-1 +
listview1.getHeaderViewsCount());

                    init();

                    break;
                case R.id.btn_del_all  :
                    data.clear(); // 전체 삭제.

                    // 데이터 변경 알림
                    adapterListview.notifyDataSetChanged();
                    break;
                case R.id.btn_init     :
                    init();
                    break;
                case R.id.btn_search   :
                    // spinner 에서 선택값 가져오기.
                    searchitem  = getFieldName( spinner.getSelectedItem().toString() );
                    searchvalue = edit_item.getText().toString() ;

                    ModelStudent.MyPredicate predicate = new
ModelStudent.MyPredicate(searchitem, searchvalue);
                    List<ModelStudent> result = (List<ModelStudent>)
CollectionUtils.select( data, predicate );
                    AdapterStudent aa = new AdapterStudent(MainActivity.this,
R.layout.view_student, result);
                    listview1.setAdapter( aa );

                    break;
                case R.id.btn_sort     :
                    // spinner 에서 선택값 가져오기.
                    searchitem  = getFieldName( spinner.getSelectedItem().toString() );
                    searchvalue = edit_item.getText().toString() ;
```

```
                // RadioGroup 에서 선택값 가져오기
                RadioGroup rgpAscDesc = findViewById(R.id.rgpAscDesc)  ;
                int ascdesc = rgpAscDesc.getCheckedRadioButtonId();
                boolean mode = false;
                switch (ascdesc){
                    case R.id.rdoAsc: mode = false; break;
                    case R.id.rdoDesc: mode = true; break;
                    default: mode = false; break;
                }

                Comparator<ModelStudent> comparator = null;
                if( searchitem.equals("name")  ) {
                    comparator = new ModelStudent.NameCompare( mode );
                }
                else if( searchitem.equals("number") ) {
                    comparator = new ModelStudent.NumberCompare( mode );
                }
                else if( searchitem.equals("department")  ) {
                    comparator = new ModelStudent.DeptCompare( mode );
                }

                Collections.sort(data, comparator);
                adapterListview.notifyDataSetChanged();

                break;
            }

        }

        private String getFieldName(String selectedItem) {

            if( selectedItem.equals("이름") ) {
                return "name";
            }
            else if( selectedItem.equals("학번") ) {
                return "number";
            }
            else if( selectedItem.equals("학과") ) {
                return "department";
            }
            else {
                return "";
            }
        }
    }

    private void init() {
        edit_name      .setText("");
        edit_number    .setText("");
        edit_department.setText("");
        edit_item      .setText("");
```

```
        listview1.setAdapter( adapterListview );
    }

    private class ListViewHandler implements ListView.OnItemClickListener
                                          , ListView.OnItemLongClickListener {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            String msg = parent.getItemAtPosition(position).toString();
            Toast.makeText( getApplicationContext(), msg, Toast.LENGTH_SHORT).show();
        }

        @Override
        public boolean onItemLongClick(AdapterView<?> parent, View view, final int position,
long id) {

            // 확인,취소 대화상자 만들기
            AlertDialog.Builder  dlg = new AlertDialog.Builder( MainActivity.this );
            dlg.setTitle("삭제 확인");
            dlg.setMessage("정말로 삭제하시겠읍니까?");
            dlg.setPositiveButton("확인", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    data.remove( position - listview1.getHeaderViewsCount() );
                    adapterListview.notifyDataSetChanged();
                }
            });
            dlg.setNegativeButton("취소",null);
            dlg.show();

            return false;
        }
    }
}
```

5.8 머리 아이템 추가

✓ /res/layout/list_view_header.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="30dp"
                android:background="#00F"
                android:padding="10dp">

    <TextView android:id="@+id/header_text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:layout_centerInParent="true"
              android:textColor="#FFF"
              android:textSize="20dp"/>

</RelativeLayout>
```

```
public class MainActivity extends AppCompactActivity {

    @Override
    protected void onCreate( Bundle savedInstanceState ) {
        ...
        // 리스트에 머리 아이템 추가
        View headerView = getLayoutInflater().inflate( R.layout.list_view_header, null);
        TextView headerTv = (TextView)headerView.findViewById( R.id.header_text );
        headerTv.setText( "리스트의 시작입니다." );
        listView.addHeaderView( headerView, null, true );
        ...
    }
}
```

5.9 꼬리 아이템 추가

✓ /res/layout/list_view_footer.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="30dp"
                android:background="#00F"
                android:padding="10dp">

    <TextView android:id="@+id/footer_text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:layout_centerInParent="true"
              android:textColor="#FFF"
              android:textSize="20dp"/>
```

```
</RelativeLayout>
```

```
public class MainActivity extends AppCompactActivity {

    @Override
    protected void onCreate( Bundle savedInstanceState ) {
        ...
        // 리스트에 꼬리 아이템 추가
        View footerView = getLayoutInflater().inflate( R.layout.list_view_footer, null );
        TextView footerTv = (TextView)footerView.findViewById( R.id.footer_text );
        footerTv.setText( "로딩 중..." );
        listView.addFooterView( footerView, null, false );
        ...
    }
}
```

5.10 구분선 추가

```
public class MainActivity extends AppCompactActivity {
    @Override
    protected void onCreate( Bundle savedInstanceState ) {
        ...
        // 리스트뷰에 구분선 추가
        listview1.setDivider( new ColorDrawable(Color.GRAY) );
        listview1.setDividerHeight( 3 );
         ...
    }
}
```

6. ListView 동적 항목 추가

위치에 따라서 데이터를 추가하기 위해서는 먼저 OnScrollListener 를 구현하여 리스트뷰에 등록해야 합니다. OnScrollListener 에서 재정의해야할 함수는 아래와 같습니다.

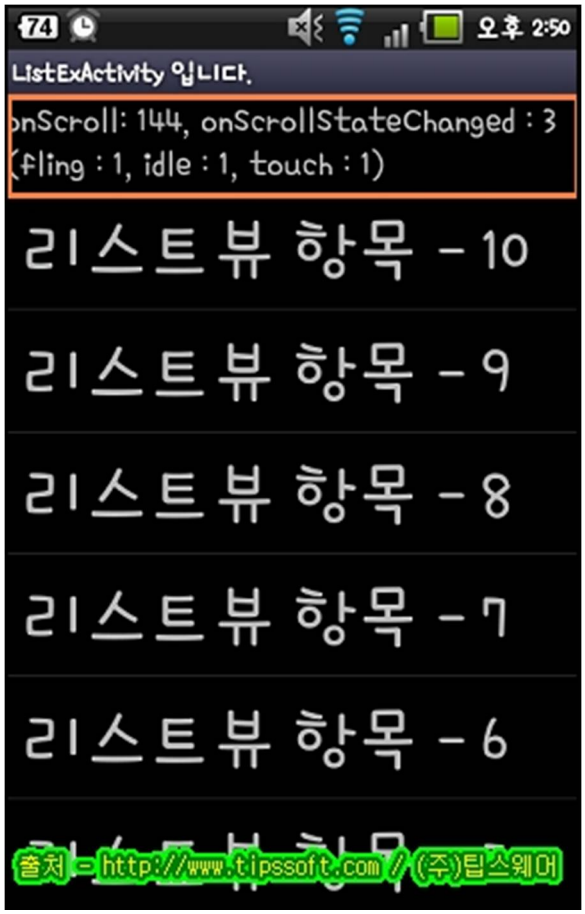
```
public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int totalItemCount)
public void onScrollStateChanged(AbsListView view, int scrollState)
```

onScroll 는 스크롤이 발생하는 동안 지속적으로 호출되는 메서드로 현재 보여지는 리스트뷰에서 상단에 보여지는 항목의 index 와 현재 리스트뷰에서 보여지는 항목의 수, 그리고 리스트뷰의 총 항목의 수를 알려줍니다.

onScrollStateChanged 는 현재 리스트뷰의 상태를 알려줍니다. scrollState 으로 넘어오는 상태값은 다음과 같은 3 가지입니다.

SCROLL_STATE_FLING (2) : 터치 후 손을 뗀 상태에서 아직 스크롤 되고 있는 상태입니다.
SCROLL_STATE_IDLE (0) : 스크롤이 종료되어 어떠한 애니메이션도 발생하지 않는 상태입니다.
SCROLL_STATE_TOUCH_SCROLL (1) : 스크린에 터치를 한 상태에서 스크롤하는 상태입니다.

onScroll 메서드의 경우 행위에 대한 콜백 메서드이고 onScrollStateChanged 메서드는 상태에 대한 콜백 메서드이기 때문에 재정의할 루틴의 성격에 따라서 어떤 메서드에서 코드를 구성할 것인지를 잘 선택해야합니다. onScrollStateChanged 메서드의 경우 한번 터치하여 스크롤 할 경우 최대 3 번 호출되지만 onScroll 메서드는 행위에 대한 콜백 메서드라는 특성상 100 번 이상 호출될 수도 있습니다. 아래의 그림은 두 메서드의 호출 빈도를 알아보기 위해 예제를 구성하여 한번의 스크롤로 각 메서드가 몇 번 호출됐는지를 나타내는 화면입니다.



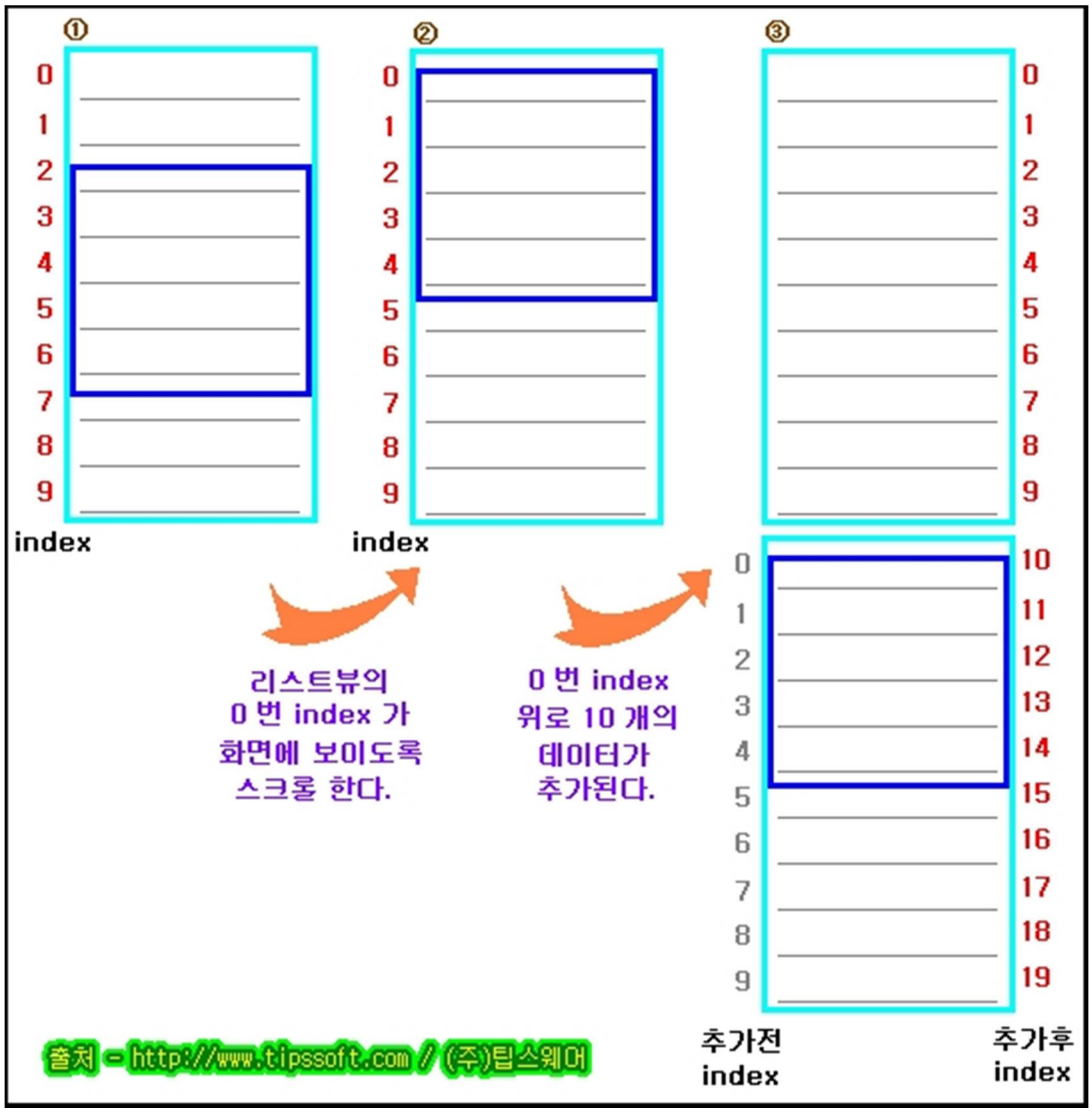
onScroll 메서드에 사용자 재정의 루틴을 추가할 경우 높은 빈도로 해당 루틴이 수행되기 때문에 어플리케이션에 부하를 줄 수 있다는 단점이 있지만 사용자가 원하는 상황을 빨리 판단하여 매끄러운 처리를 할 수 있다는 장점이 있습니다.

반대로 onScrollStateChanged 메서드에 사용자 재정의 루틴을 추가하면 메서드의 호출 빈도가 상대적으로 낮기때문에 어플리케이션이 가볍게 동작할 수 있지만 상대적으로 매끄럽지 못하게 사용자 처리가 될 수 있다는 단점이 있습니다.

그러므로 OnScrollListener 를 사용할 때에는 두 메서드의 장단점을 잘 고려하여 사용해야합니다.

6.1 동적 항목 추가 원리

리스트뷰의 항목을 동적으로 추가하려면 항목이 몇개씩 추가할 것인지와 항목을 추가할 기준 위치가 리스트뷰의 상단과 하단중 어느 지점인지를 정해야합니다. 이 강좌에서는 아래 그림처럼 리스트뷰의 최상단으로 스크롤됐을 때 (즉, 리스트뷰의 0 번 인덱스 항목이 보여졌을 때) 해당 항목 위로 10 개씩 데이터 항목이 추가되도록 예제를 구성할 것입니다.



6.2 코드 구성 하기

간단하게 동적으로 항목을 추가하는 것에 대하여 OnScrollListener 인터페이스의 onScroll 메서드를 이용하는 방법과 onScrollStateChanged 메서드를 이용하는 방법을 각각 알아보도록 하겠습니다.

6.2.1 onScroll 메서드를 사용하는 방법

onScroll 메서드는 호출 빈도가 매우 높기 때문에 스크롤이 보이지 않는다면 항목이 동적으로 추가되고 있다는 것을 느끼지 못할 정도로 매끄럽게 처리됩니다.

```
import android.app.Activity;
import android.os.Bundle;

import android.widget.AdapterView;
import android.widget.AbsListView;
import android.widget.AbsListView.OnScrollListener;
import android.widget.ListView;
import android.view.View;
```

```
import java.util.ArrayList;
import android.widget.ArrayAdapter;

public class ListExActivity extends Activity implements OnScrollListener
{
    // 동적으로 추가하는 데이터 단위
    public final static int INSERT_COUNT = 10;
    private ArrayAdapter<String> m_adapter = null;
    private ListView m_list = null;


    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        String str;
        ArrayList<String> str_list = new ArrayList<String>();
        m_adapter = new ArrayAdapter(this, R.layout.list_item, str_list);

        // 리스트뷰에 10 개의 데이터를 추가한다.
        for(int i = 0; i < INSERT_COUNT; i++) {
            str = "리스트뷰 항목 - " + (i + 1);
            m_adapter.insert(str, 0);
        }

        m_list = (ListView) findViewById(R.id.scroll_list);
        m_list.setAdapter(m_adapter);
        // 리스트뷰에 OnScrollListener 를 설정한다.
        m_list.setOnScrollListener(this);
        // 리스트뷰의 하단이 보여지도록 설정한다.
        m_list.setSelection(m_list.getCount() - 1);
    }

    @Override
    public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalItemCount)
    {
        // 리스트뷰가 구성이 완료되어 보이는 경우
        if(view.isShown()){
            // 리스트뷰의 0 번 인덱스 항목이 리스트뷰의 상단에 보이고 있는 경우
            if(firstVisibleItem == 0) {
                // 항목을 추가한다.
                String str;
                for(int i = 0; i < INSERT_COUNT; i++) {
                    str = "리스트뷰 항목 - " + (totalItemCount + i + 1);
                    m_adapter.insert(str, 0);
                }
                // 0 번 인덱스 항목 위로 INSERT_COUNT 개수의 항목이 추가되었으므로
                // 기존의 0 번 인덱스 항목은 INSERT_COUNT 번 인덱스가 되었다.
                // 기존 0 번 항목이 보여져서 항목이 추가될때 해당 항목의 모든 영역이
                // 보이지않았을 수도 있으므로 이미 모든 영역이 노출됐던 INSERT_COUNT + 1
```

```
                // 항목을 보이도록 설정하여 스크롤을 부드럽게 보이도록 한다.
                view.setSelection(INSERT_COUNT + 1);
            }
        }
    }

    @Override
    public void onScrollStateChanged(AbsListView view, int scrollState)
    {
    }
}
```

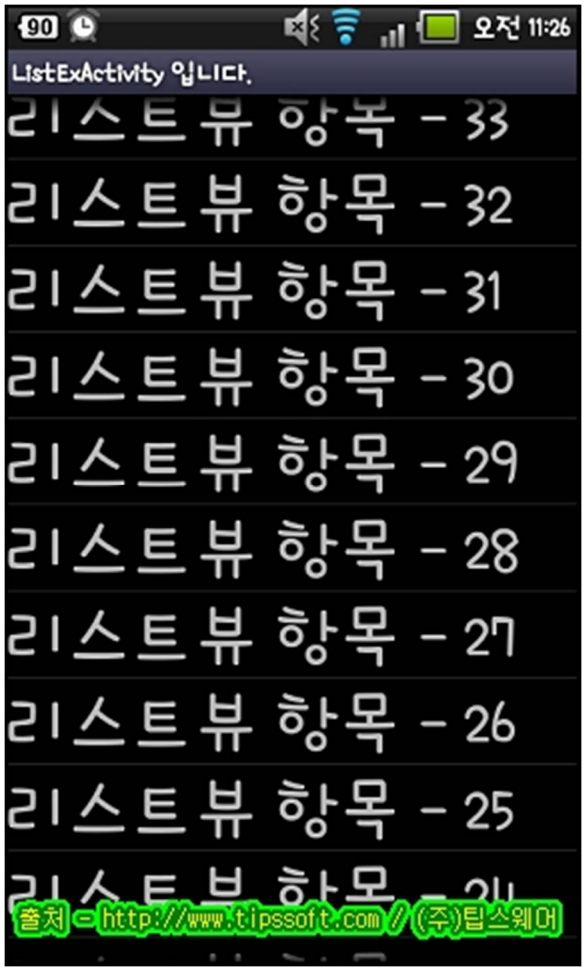
6.2.2 onScrollStateChanged 메서드를 사용하는 방법

onScrollStateChanged 메서드는 호출 빈도가 onScroll 메서드에 비해 적고 사용가능한 상태값중 SCROLL_STATE_IDLE (스크롤 중지 상태) 에서만 항목 추가가 발생하므로 스크롤시 항목이 추가될 때마다 끊김 현상이 나타납니다.

```
import android.app.Activity;
import android.os.Bundle;

import android.widget.AdapterView;
import android.widget.AbsListView;
import android.widget.AbsListView.OnScrollListener;
import android.widget.ListView;
import android.view.View;
import java.util.ArrayList;
import android.widget.ArrayAdapter;

public class ListExActivity extends Activity implements OnScrollListener
{
    // 동적으로 추가하는 데이터 단위
    public final static int INSERT_COUNT = 10;
    private ArrayAdapter<String> m_adapter = null;
    private ListView m_list = null;

    private int m_list_count = 0;


    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        String str;
        ArrayList<String> str_list = new ArrayList<String>();
        m_adapter = new ArrayAdapter(this, R.layout.list_item, str_list);

        // 리스트뷰에 10 개의 데이터를 추가한다.
```

```
        for(int i = 0; i < INSERT_COUNT; i++) {
            str = "리스트뷰 항목 - " + (i + 1);
            m_adapter.insert(str, 0);
        }
        m_list_count += INSERT_COUNT;

        m_list = (ListView) findViewById(R.id.scroll_list);
        m_list.setAdapter(m_adapter);
        // 리스트뷰에 OnScrollListener 를 설정한다.
        m_list.setOnScrollListener(this);
        // 리스트뷰의 하단이 보여지도록 설정한다.
        m_list.setSelection(m_list.getCount() - 1);
    }

    @Override
    public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalItemCount)
    {
    }

    @Override
    public void onScrollStateChanged(AbsListView view, int scrollState)
    {
        // 리스트뷰가 구성이 완료되어 보이는 경우
        if(view.isShown()){
            if(scrollState == SCROLL_STATE_IDLE) {
                // 리스트뷰의 0 번 인덱스 항목이 리스트뷰의 상단에 보이고 있는 경우
                if(view.getFirstVisiblePosition() == 0) {
                    // 항목을 추가한다.
                    String str;
                    for(int i = 0; i < INSERT_COUNT; i++) {
                        str = "리스트뷰 항목 - " + (m_list_count + i + 1);
                        m_adapter.insert(str, 0);
                    }
                    m_list_count += INSERT_COUNT;

                    // 0 번 인덱스 항목 위로 INSERT_COUNT 개수의 항목이 추가되었으므로
                    // 기존의 0 번 인덱스 항목은 INSERT_COUNT 번 인덱스가 되었다.
                    // 호출 빈도가 매우 적은 onScrollStateChanged 에서는 기존 0 번 항목이
보여져서
                    // 항목이 추가될때 해당 항목의 모든 영역이 보였을 가능성이 크므로
                    // 해당 항목을 보이도록 설정한다.
                    view.setSelection(INSERT_COUNT);
                }
            }
        }
    }
}
```

4. 실행 화면



6.3 ListView 스크롤 위치 저장 및 복원

ListView 를 일정 시간 간격으로 재로딩할 경우 스크롤 위치를 저장하고 복원할 필요가 있다.

현재 위치값을 알아 내려면, 아래 함수를 호출하여 저장.

ListView.getLastVisiblePosition();
ListView.getFirstVisiblePosition();

ListView 를 다시 열었을때 onCreate 에서 아래 함수 실행.

ListView.setSelection(저장된 값);

ListView.scrollTo 함수를 쓰게 되면 scroll thumb 은 이동을 하나 ListView 에 list item 이 제대로 표시가 안됨.

가장 상위가 0 부터 시작한다.

6.4 Activity 에서 사용하는 예제

```
private void getListScrollInfo() {
        mPosition = mList.getFirstVisiblePosition();

        View v = mList.getChildAt(0);
```

```
        if(v != null)
            mScrollY = (int) v.getTop();
        else
            mScrollY = 0;

        Log.i("mPosition", "" + mPosition + "," + mScrollY);
}

mList.setAdapter(adapter);

mList.setSelectionFromTop(mPosition, mScrollY);
```

6.5 getView 에서 bindView 와 newView 의 호출 과정

getView 는 newView 와 bindView 를 모두 호출하는 함수에요.

```
public View getView(int position, View convertView, ViewGroup parent) {

    if (!mDataValid) {
        throw new IllegalStateException("this should only be called when the cursor is valid");
    }

    if (!mCursor.moveToPosition(position)) {
        throw new IllegalStateException("couldn't move cursor to position " + position);
    }

    View v;

    if (convertView == null) {
        v = newView(mContext, mCursor, parent);
    } else {
        v = convertView;
    }

    bindView(v, mContext, mCursor);

    return v;
}
```

ListView 아이템을 보여줄때 계속해서 getView 를 호출하게 되죠. 스크롤 할 때도 계속 호출..

근데 아이템의 View 는 안 바뀌는데 계속 가져올 필요는 없으니 View 가 처음 보여질 때 즉 null 인 경우 newView 를 통해 view 를 생성합니다. 그리고 나서 bindView 를 호출하죠.

이후에 스크롤해서 화면을 이동했다가 다시 해당 아이템이 화면에 보여질 때는 이전에 newView 를 통해 생성한 view 가 있으니 bindView 만 호출하고요.

6.6 getView 중복 호출 문제

원인 : xml 파일에서 크기를 wrap_content 로 지정해준 것이었다.

해결 : 상위 뷰만 뿐만 아니라 ListView 자신의 가로,세로도 꼭 fill_parent 로 해주어야 한다.

Reference

[1] 해결법 1 : http://www.androidpub.com/41674

[2] getView 호출에 대한 좋은 설명을 해주신 개발자분 : http://myelf.egloos.com/2723978

[3] getView 를 구현할 때 성능을 고려한 좋은 설계 방안 : http://blog.naver.com/PostView.nhn?blogId=dythmall&logNo=30095935698&parentCategoryNo=8&viewDate=&currentPage=1&listtype=0

http://atin.tistory.com/379

7. Reference

http://www.codeproject.com/Articles/11385/Aspect-Oriented-Programming-in-NET

http://recipes4dev.tistory.com/42

http://recipes4dev.tistory.com/48

http://itmir.tistory.com/477

http://hrhdev.tistory.com/45

http://www.ezzylearning.com/tutorial/customizing-android-listview-items-with-custom-arrayadapter

http://recipes4dev.tistory.com/42

http://recipes4dev.tistory.com/57

Fragment 에서 ListView 사용하기

지금까지의 ListView 기본 사용법은 Activity 에 ListView 를 추가할 때 사용하는 방법입니다. 만약 동일한 코드를 사용하여 Fragment 에 ListView 를 추가하면 어떤 결과가 나올까요? 직접 코드를 작성하여 빌드해보면 아시겠지만 "에러가 발생"합니다.

```
    error: no suitable constructor found for
ArrayAdapter(MainActivityFragment,int,String[])
        ...
    error: cannot find symbol method findViewById(int)
```

첫 번째 에러가 발생하는 원인은 ArrayAdapter 생성자의 첫번째 파라미터로 전달되는 Context 에 대한 접근 문제 때문입니다. Activity 는 android.content.Context 로부터 상속받은 클래스이므로 Context 의 기능을 그대로 이용할 수 있기 때문에 onCreate() 함수에서 ArrayAdapter 생성자의 첫번째 파라미터로 this 를 전달할 수 있습니다. 하지만 Fragment 는 java.lang.Object 로부터 직접 상속받은 클래스입니다. 즉, Context 와는 전혀 관계가 없는 상태라서 this 를 사용할 수 없는거죠.

그럼 Fragment 에서는 ListView 를 사용할 수 없는 걸까요? 아닙니다. 당연히 해결할 수 있는 방법이 존재합니다. 그것도 아주 간단하게 말이죠. 바로 Fragment 에서 자신이 속한 Activity 의 참조를 얻어온 다음 ArrayAdapter 의 생성자로 전달하는 것입니다. 이미 Fragment 에는 Activity 의 참조를 획득할 수 있는 getActivity()라는 함수가 있습니다.

두 번째 에러는 findViewById() 함수 호출 때문입니다. findViewById() 함수는 Fragment 의 멤버 함수가 아닙니다. View 의 멤버 함수이죠. 즉, Fragment 의 멤버 함수가 아닌 findViewById() 함수를 Fragment 에서 호출하려고 하니 에러가 발생하는 것입니다.

해결 방법은 LayoutInflater 를 사용하여 Resource Layout 을 View 로 변환한 다음 해당 View 를 사용하여 findViewById()를 호출하는 것입니다.

Fragment 에 ListView 를 추가하는 코드는 다음처럼 작성하시면 됩니다.

```
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        // Inflate the layout for this fragment
        View view = inflater.inflate(R.layout.fragment1, null) ;
        ArrayAdapter Adapter = new ArrayAdapter(getActivity(),
android.R.layout.simple_list_item_1, LIST_MENU) ;

        ListView listview = (ListView) view.findViewById(R.id.listview1) ;
        listview.setAdapter(adapter) ;
    }
```