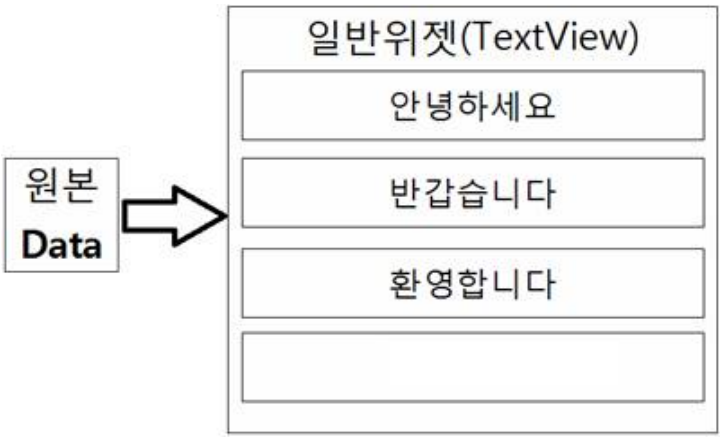
목차

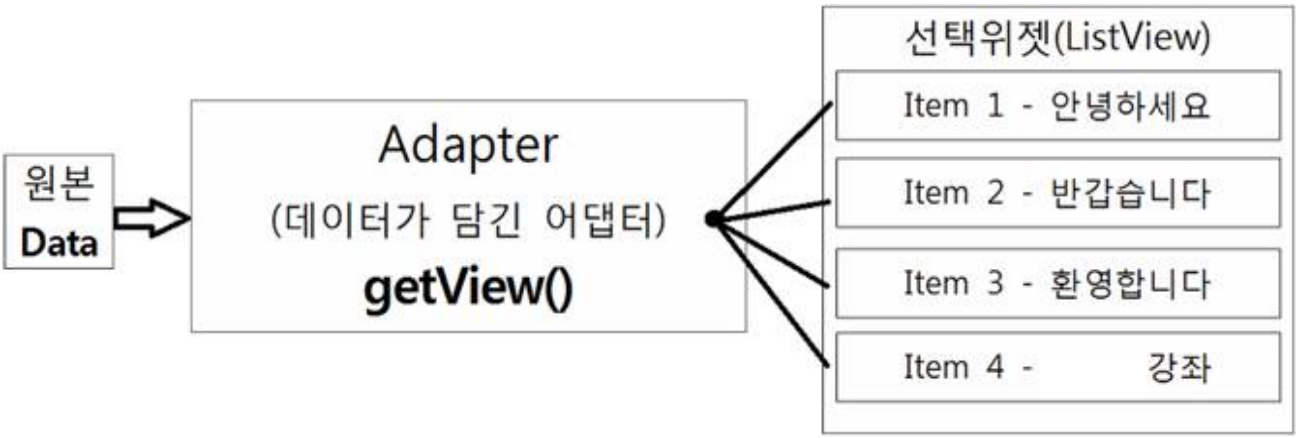
1. ListView 란?

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

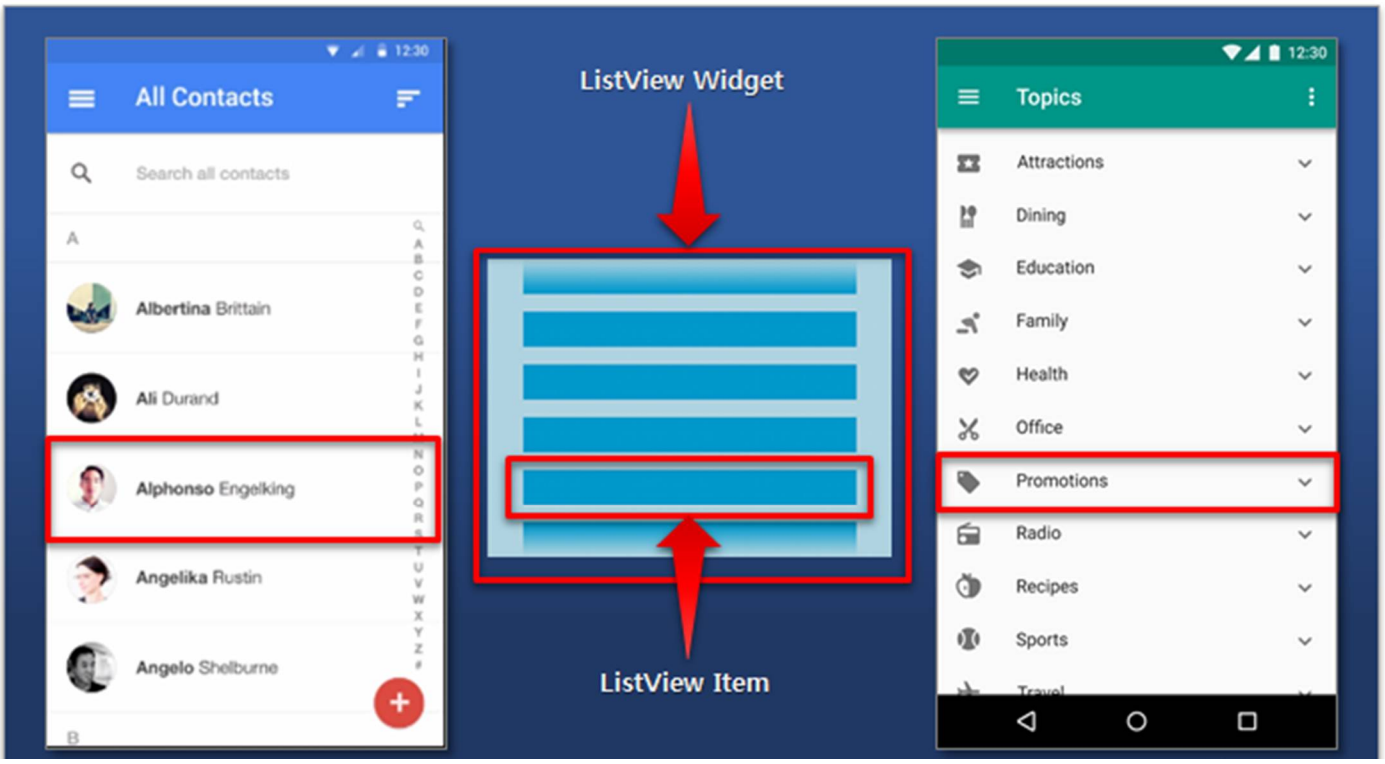


- ListView 의 데이터 설정방법 – 직접 설정이 불가능 하고 Adapter 를 이용한다

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



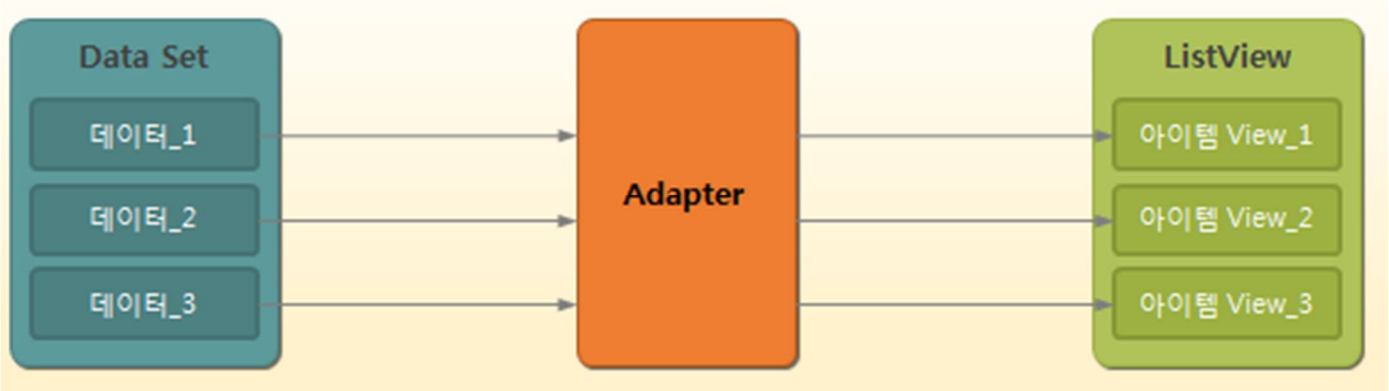
java.lang.Object
↳android.view.View
↳android.view.ViewGroup
↳android.widget.AdapterView<android.widget.ListAdapter>
↳ android.widget.AbsListView
↳ android.widget.ListView

2. Adapter 란?

✓ "사용자가 정의한 데이터를 View 에 출력하기 위해 사용하는 객체다"

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

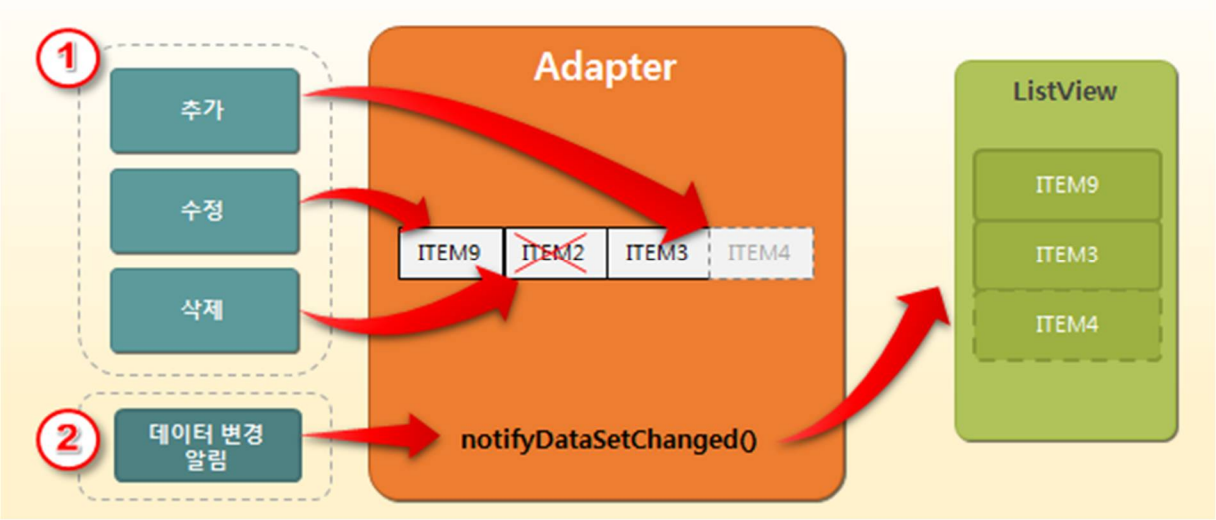
- Adapter 는 뷰(Client)와 데이터를 연결해주는 역할을 한다.
- Adapter 가 하는 역할은 사용자 데이터를 입력 받아 View 를 생성하는 것이며 Adapter 에서 생성되는 View 는 ListView 내 하나의 아이템 영역에 표시되는 것입니다.



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

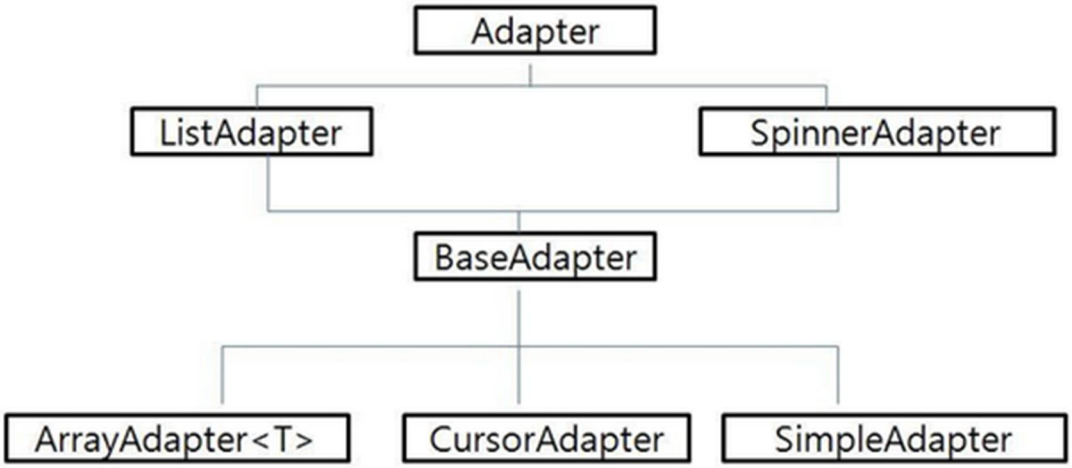
✓ notifyDataSetChanged()

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:
- 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

2.3 ArrayAdapter 의 사용 예제

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

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

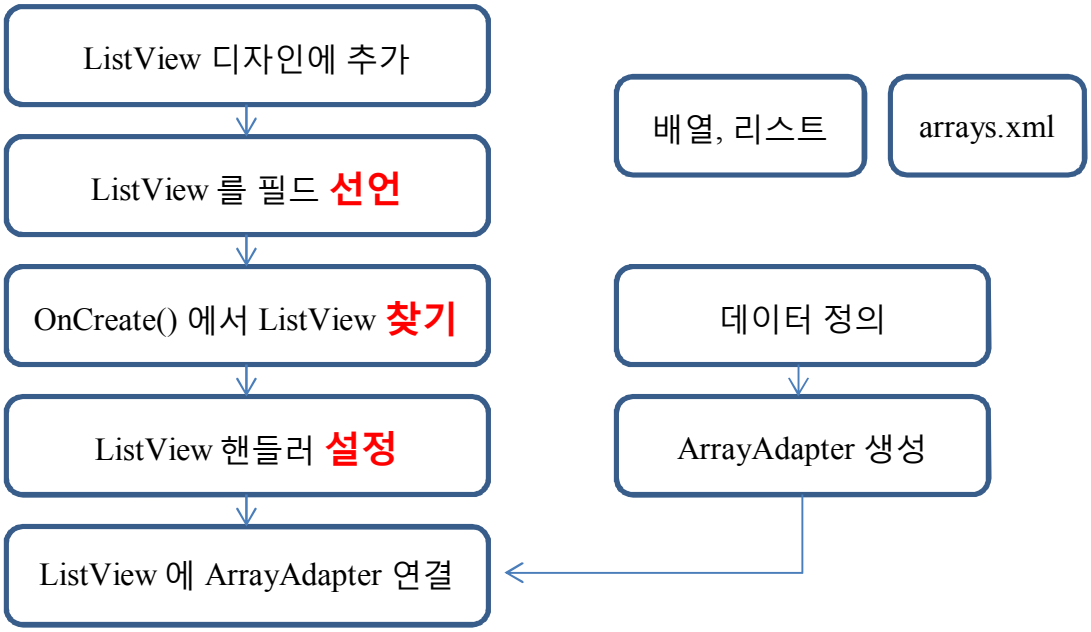
- this - 사용할 Context. 일반적으로 아답터를 포함하는 activity 의 context 가 사용됨.
- android.R.layout.simple_list_item_1 - 아이템이 표시될 view 의 resource ID.
  어떤 형식의 **TextView 리소스**를 지정하느냐에 따라 아이템의 text 형태가 바뀜.

simple_list_item_1	하나의 텍스트뷰로 구성된 레이아웃
simple_list_item_2	두개의 텍스트뷰로 구성된 레이아웃
simple_list_item_checked	오른쪽에 체크표시가 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.

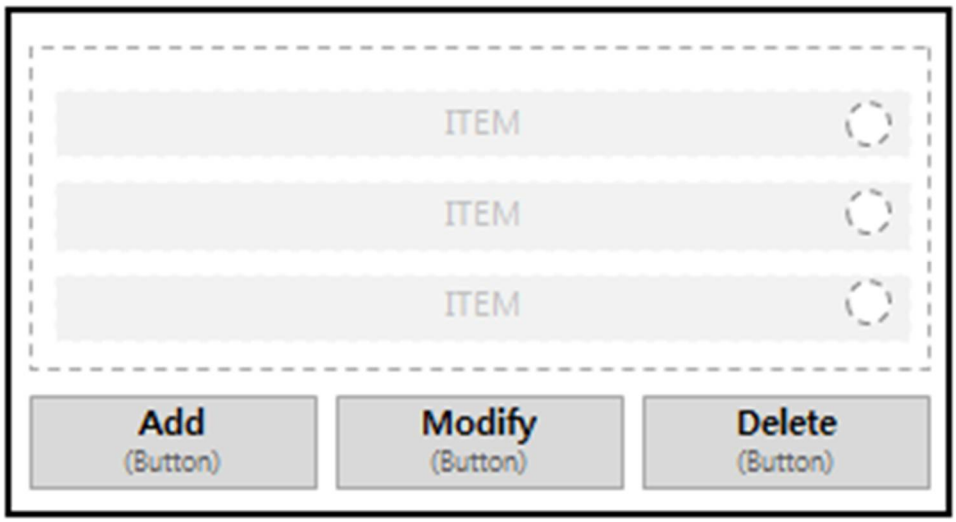
3. ListView 의 작업 순서



4. Simple ListView 예제 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

ListView 사용법에 대한 단순 예제로만 UI 화면을 구성하겠습니다.

"activity_main.xml" 파일에 ListView 를 추가합니다.

```
<?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"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.listviewexample1.MainActivity"
    tools:showIn="@layout/activity_main">

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

```
</RelativeLayout>
```

4.2 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"} ;
    // ... 코드 계속
}
```

4.3 Adapter 생성

사용자 데이터가 준비되었으니 해당 데이터를 입력 받아 View 로 만들어줄 Adapter 를 생성해야겠죠. 안드로이드 SDK 에서 기본적으로 제공하는 Adapter 는 여러 종류가 있고 사용자가 용도에 맞게 선택해서 사용할 수 있는데, 여기서는 string 배열을 입력으로 받으므로 ArrayAdapter 를 사용하겠습니다.

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

        ArrayAdapter adapter = new ArrayAdapter(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.

4.4 ListView 와 Adapter 연결

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

        ArrayAdapter adapter = new ArrayAdapter(this,
android.R.layout.simple_list_item_1, LIST_MENU) ;

        ListView listview = (ListView) findViewById(R.id.listview1) ;
        listview.setAdapter(adapter) ;
        // ... 코드 계속.
    }
```

4.5 ListView 핸들러 설정

보통 ListView 는 단순히 데이터를 리스트(목록) 형태로 보여주기 위해 사용하기도 하지만, 리스트 아이템 자체를 선택 가능(클릭)하도록 만들어 메뉴처럼 사용할 수도 있습니다. ListView 가 사용자의 아이템 터치 입력을 받을 수 있도록 만드려면 ListView 에 Click 이벤트에 대한 리스너를 설정해주면 됩니다.

```
    listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView parent, View v, int position, long id) {

            // get TextView's Text.
            String strText = (String) parent.getItemAtPosition(position) ;

            // TODO : use strText
        }
    }) ;
```

onItemClick() 함수에서 ListView 아이템인 TextView 의 텍스트를 가져오려면 위의 주석에 있는 내용처럼 parent.getItemAtPosition() 함수를 사용하면 됩니다.

4.6 ListView 및 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. choiceMode 에 대한 자세한 내용은 다음 기회에 다른 예제를 통해 자세히 설명하도록 하겠습니다.

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

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.listviewitemcontrolexample1.MainActivity"
    tools:showIn="@layout/activity_main">

    <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_choice" 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 ...
    }
}
```

4.6.1 "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();
            }
        }) ;
    }
}
```

4.6.2 "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();
                    }
                }
            }
```

```
        }) ;
}
```

4.6.3 "삭제"에 대한 이벤트 처리.

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

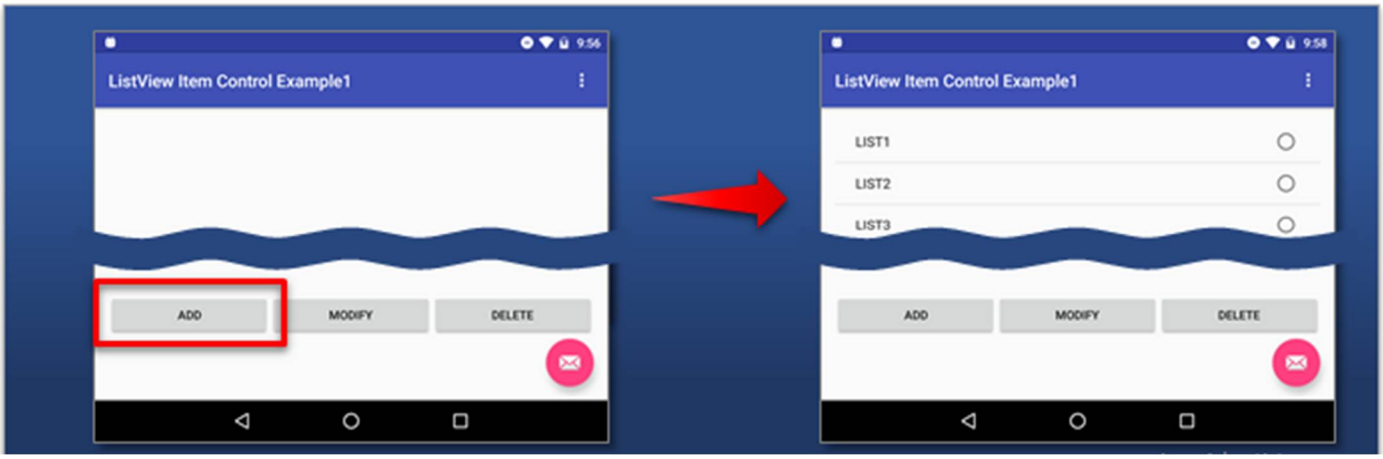
[STEP-5] "MainActivity.java" - [STEP-3] 아래에 "수정" 버튼에 대한 핸들러 작성.

```
        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();
                    }
                }
            }
        }) ;
}
```

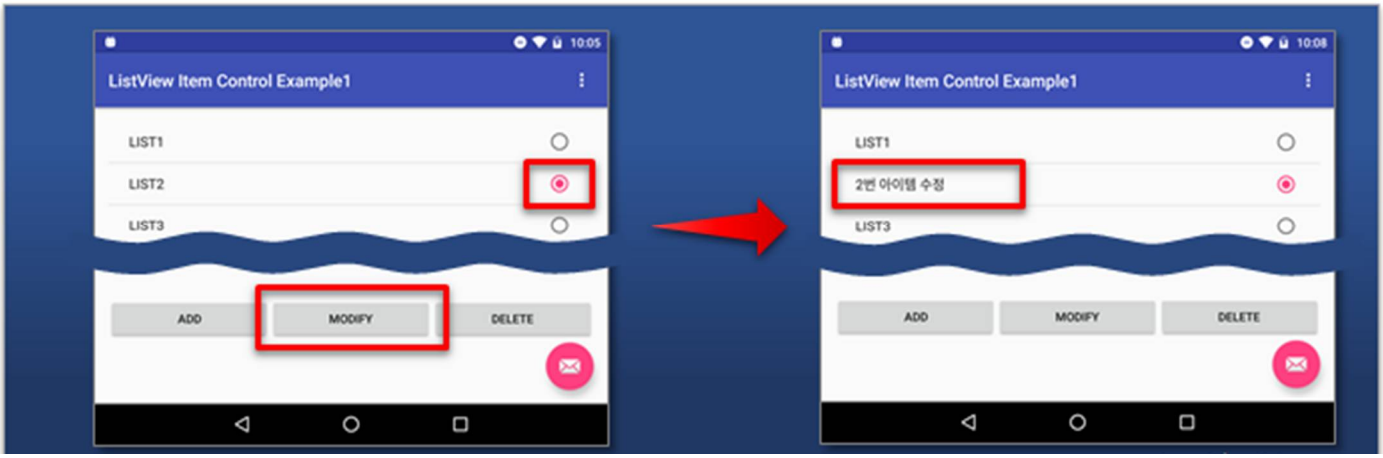
4.7 예제 실행 화면

아래는 예제를 실행한 결과 화면입니다.

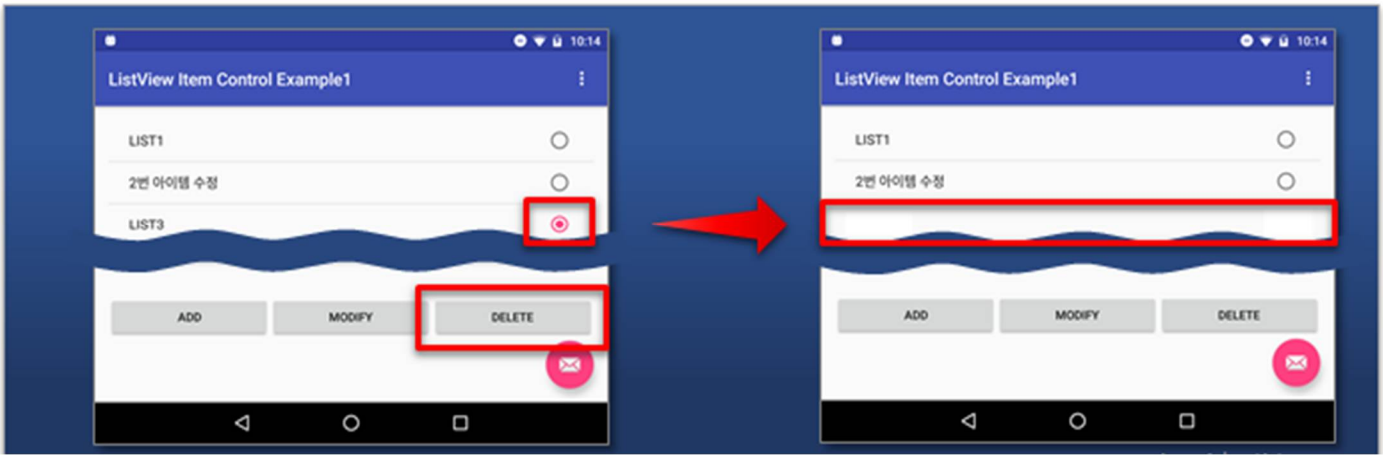
"Add" 버튼을 클릭했을 때 출력 결과입니다.



아이템을 선택하고 "Modify" 버튼을 클릭했을 때 출력 결과입니다.



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

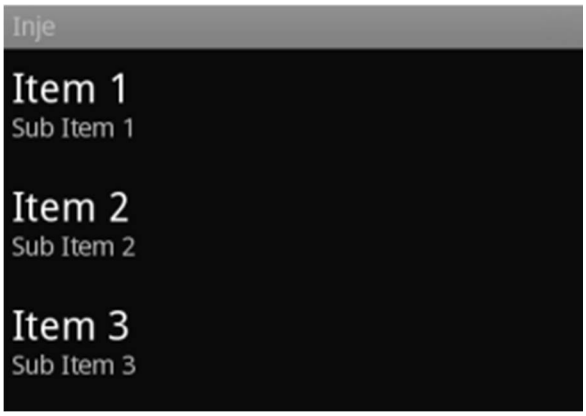


5. Custom ListView 예제 만들기

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

오늘 올릴 글 내용은 리스트 뷰를 사용시 개발자가 원하는 대로 사용하는 방법입니다.

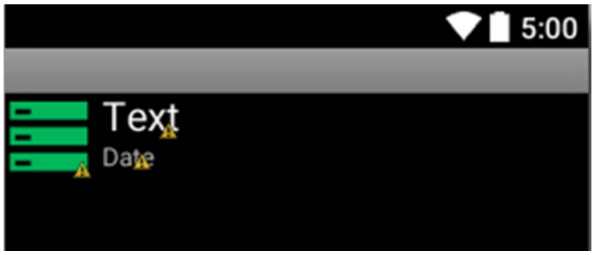
일반적으로 리스트 뷰라고 생각하면



이렇게 사용을 많이 하시죠? 그런데 리스트 사용을 이미지도 넣고 싶고 타이틀과 작은 타이틀을 같이 넣고 싶을때는 어떻게 해야할까요?

방법은 간단합니다. 우선 리스트의 각 아이템에 보여주고 싶은 모양을 설정하세요.

저는 예제로 다음과 같이 설정하겠습니다.



이미지가 하나 들어가고 그 옆에 Main Title 과 Sub Title 이 들어가도록 설정을 해보려 합니다.

5.1 ItemView 만들기

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

- listview_item.xml

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

    <ImageView
        android:id="@+id/mImage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:orientation="vertical" >

        <TextView
            android:id="@+id/mText"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Text"
            android:textAppearance="?android:attr/textAppearanceLarge" />

        <TextView
            android:id="@+id/mDate"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Date"  />

    </LinearLayout>

</LinearLayout>
```

5.2 Model 클래스 만들기

이제 아이템에 대한 틀을 만들고 나면 리스트에 연결 시켜줄 어댑터와 각 아이템에 들어갈 정보를 저장할 클래스를 만들겠습니다.

- ListData.java

```
import java.text.Collator;
import java.util.Comparator;
import android.graphics.drawable.Drawable;

public class ListData {
    /**
     * 리스트 정보를 담고 있을 객체 생성
     */
    // 아이콘
    public Drawable mIcon;

    // 제목
    public String mTitle;

    // 날짜
    public String mDate;

    /**
     * 알파벳 이름으로 정렬
     */
    public static final Comparator<ListData> ALPHA_COMPARATOR = new Comparator<ListData>() {
        private final Collator collator = Collator.getInstance();

        @Override
        public int compare(ListData mListDate_1, ListData mListDate_2) {
            return collator.compare(mListDate_1.mTitle, mListDate_2.mTitle);
        }
    };
}
```

5.3 Adapter 만들기

5.3.1 holder 만들기

- ListViewAdapter.java

```
import java.util.ArrayList;
import java.util.Collections;

import android.content.Context;
import android.graphics.drawable.Drawable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;

public class ListViewAdapter extends BaseAdapter {
    private Context mContext = null;
    public ArrayList<ListData> mListData = new ArrayList<ListData>();

    public ListViewAdapter(Context mContext) {
        super();
        this.mContext = mContext;
    }

    @Override
    public int getCount() {
        return mListData.size();
    }

    @Override
    public Object getItem(int position) {
        return mListData.get(position);
    }

    @Override
    public long getItemId(int position) {
        return position;
    }

    public void addItem(Drawable icon, String mTitle, String mDate){
        ListData addInfo = null;
        addInfo = new ListData();
        addInfo.mIcon = icon;
        addInfo.mTitle = mTitle;
        addInfo.mDate = mDate;

        mListData.add(addInfo);
    }

    public void remove(int position){
        mListData.remove(position);
```

```
        dataChange();
    }

    public void sort(){
        Collections.sort(mListData, ListData.ALPHA_COMPARATOR);
        dataChange();
    }

    public void dataChange(){
        this.notifyDataSetChanged();
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder holder;
        if (convertView == null) {
            holder = new ViewHolder();

            LayoutInflater inflater = (LayoutInflater)
mContext.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
            convertView = inflater.inflate(R.layout.listview_item, null);

            holder.mIcon = (ImageView) convertView.findViewById(R.id.mImage);
            holder.mText = (TextView) convertView.findViewById(R.id.mText);
            holder.mDate = (TextView) convertView.findViewById(R.id.mDate);

            convertView.setTag(holder);
        }else{
            holder = (ViewHolder) convertView.getTag();
        }

        ListData mData = mListData.get(position);

        if (mData.mIcon != null) {
            holder.mIcon.setVisibility(View.VISIBLE);
            holder.mIcon.setImageDrawable(mData.mIcon);
        }else{
            holder.mIcon.setVisibility(View.GONE);
        }

        holder.mText.setText(mData.mTitle);
        holder.mDate.setText(mData.mDate);

        return convertView;
    }

      private class ViewHolder {
          public ImageView mIcon;

          public TextView mText;

          public TextView mDate;
      }
```

```
}
```

5.4 디자인에 listView 추가

- activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >

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

</LinearLayout>
```

5.5 ListView 와 Adapter 연결

5.6 ListView 핸들러 설정

- MainActivity.java

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ListView;
import android.widget.Toast;

public class MainActivity extends Activity {
    private ListView mListView = null;
    private ListViewAdapter mAdapter = null;

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

        mListView = (ListView) findViewById(R.id.mList);
```

```
        mAdapter = new ListViewAdapter(this);
        mListView.setAdapter(mAdapter);

        mAdapter.addItem(getResources().getDrawable(R.drawable.ok), "확인이 완료되었습니다",
"2014-02-18");
        mAdapter.addItem(getResources().getDrawable(R.drawable.idonknow), "난 몰라요 ㅠㅠ",
"2014-02-01");
        mAdapter.addItem(getResources().getDrawable(R.drawable.supersu), "슈퍼유저~~", "2014-
02-04");
        mAdapter.addItem(null, "이미지가 null이면...", "2014-02-15");

        mListView.setOnItemClickListener(new OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> parent, View v, int position, long id){
                ListData mData = mAdapter.mListData.get(position);
                Toast.makeText(MainActivity.this, mData.mTitle, Toast.LENGTH_SHORT).show();
            }
        });
    }

}
```

5.7 Header 추가

5.8 Footer 추가

5.9 Divider 추가