RecyclerView 2

2019-09-06

이승진

**학습목표**

RecyclerView 객체를 사용하여 객체 목록을 화면에 표시하기

RecyclerView 항목 클릭 기능 구현

RecyclerView 항목 삭제 기능 구현

**목차**

[1. RecyclerView2Activity 화면 만들기 2](#_Toc24366017)

[1) 개요 2](#_Toc24366018)

[2) 새 액티비티 생성 2](#_Toc24366019)

[3) activity\_recycler\_view2.xml 수정 3](#_Toc24366020)

[4) RecyclerView 라이브러리 등록 확인하기 6](#_Toc24366021)

[5) item2.xml 파일 생성 8](#_Toc24366022)

[6) Item2.java 생성 12](#_Toc24366023)

[7) RecyclerView2Adapter.java 생성 14](#_Toc24366024)

[8) RecyclerView2Activity.java 수정 16](#_Toc24366025)

[2. MainActivity 수정 17](#_Toc24366026)

[1) activity\_main.xml 수정 17](#_Toc24366027)

[2) MainActivity.java 수정 18](#_Toc24366028)

[3) 실행 19](#_Toc24366029)

[3. RecyclerView 항목 클릭 기능 구현 20](#_Toc24366030)

[1) RecyclerView2Adapter.java 수정 20](#_Toc24366031)

[2) 실행 22](#_Toc24366032)

[4. 삭제 기능 구현 23](#_Toc24366033)

[1) item2.xml 23](#_Toc24366034)

[2) item2a.xml 생성 24](#_Toc24366035)

[3) Item2.java 수정 29](#_Toc24366036)

[4) RecyclerView2Adapter.java 수정 30](#_Toc24366037)

[5) 삭제 메뉴 구현 32](#_Toc24366038)

[6) RecyclerView2Activity.java 수정 34](#_Toc24366039)

[7) 실행 35](#_Toc24366040)

[5. 메뉴 상태 변경 36](#_Toc24366041)

[1) RecyclerView2Adapter.java 수정 36](#_Toc24366042)

[2) RecyclerView2Activity.java 수정 38](#_Toc24366043)

[3) 실행 40](#_Toc24366044)

[6. 삭제 확인 대화상자 41](#_Toc24366045)

[1) strings.xml 수정 41](#_Toc24366046)

[2) RecyclerView2Activity.java 수정 41](#_Toc24366047)

[3) 실행 43](#_Toc24366048)

[7. 부록 44](#_Toc24366049)

# RecyclerView2Activity 화면 만들기

## 개요

지난 시간에는 RecyclerView를 이용하여 문자열(String) 목록을 화면에 표시하였다.

즉 데이터 항목은 문자열 한 개이었다.

이번 시간에는 데이터 항목에 문자열과 시각을 표시하자.

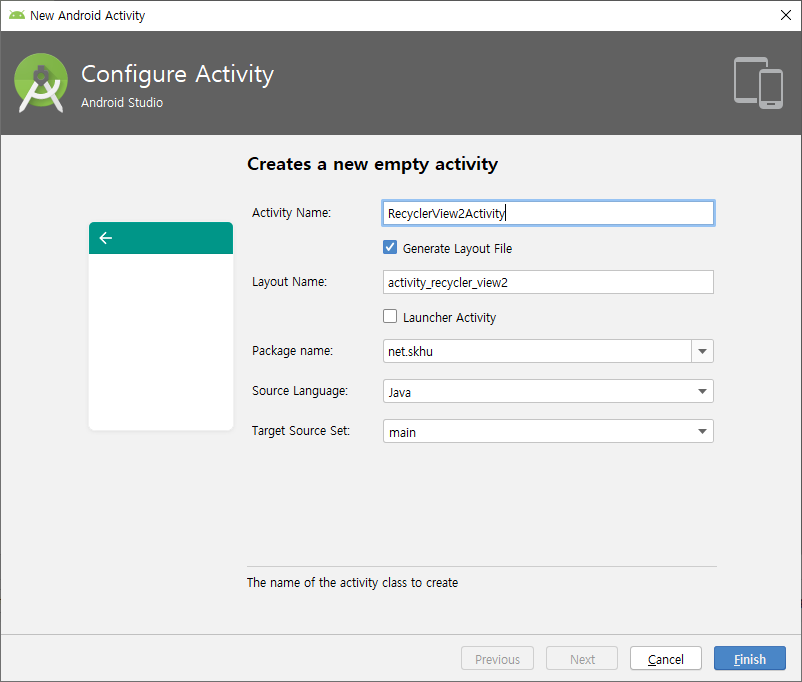
그러기 위해서는 데이터 항목에 해당하는 클래스를 구현해야 한다. (Item2.java)

## 새 액티비티 생성

E03List 프로젝트에 새 액티비티를 추가하자.

Project 창에서 app 폴더를 마우스 오른쪽 버튼으로 클릭하고

메뉴에서 New - Activity - Empty Activity 클릭

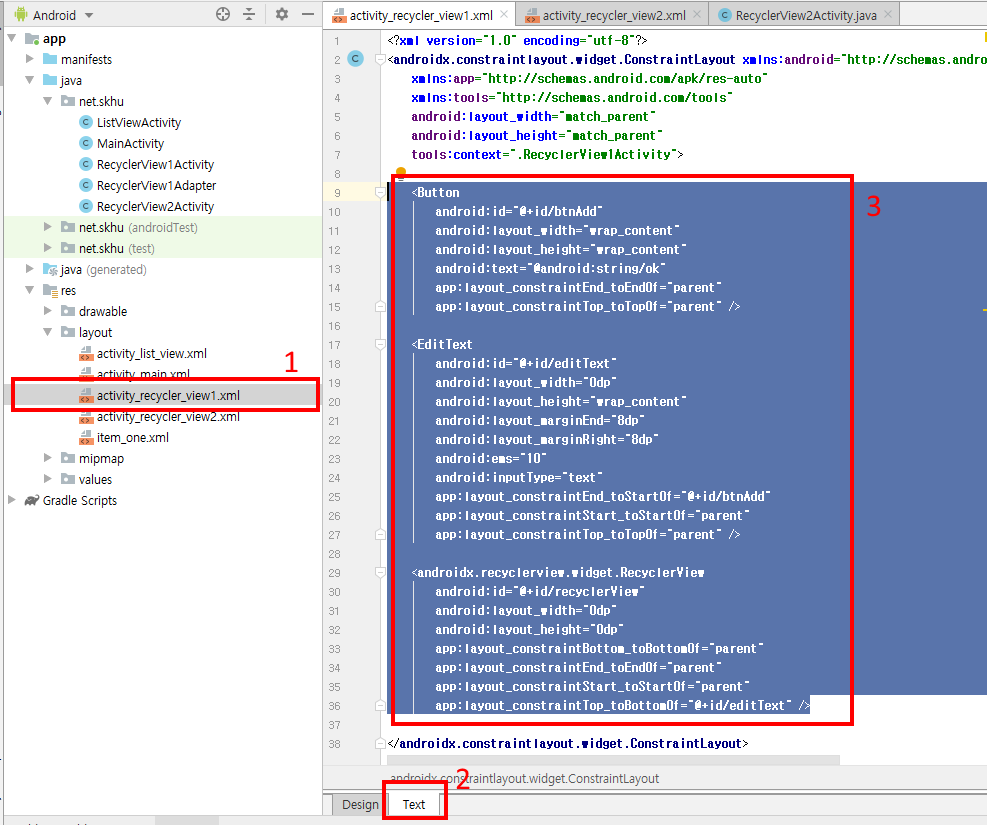


## activity\_recycler\_view2.xml 수정

activity\_recycler\_view1.xml 파일과 동일한 레이아웃 리소스를 만들려고 한다.

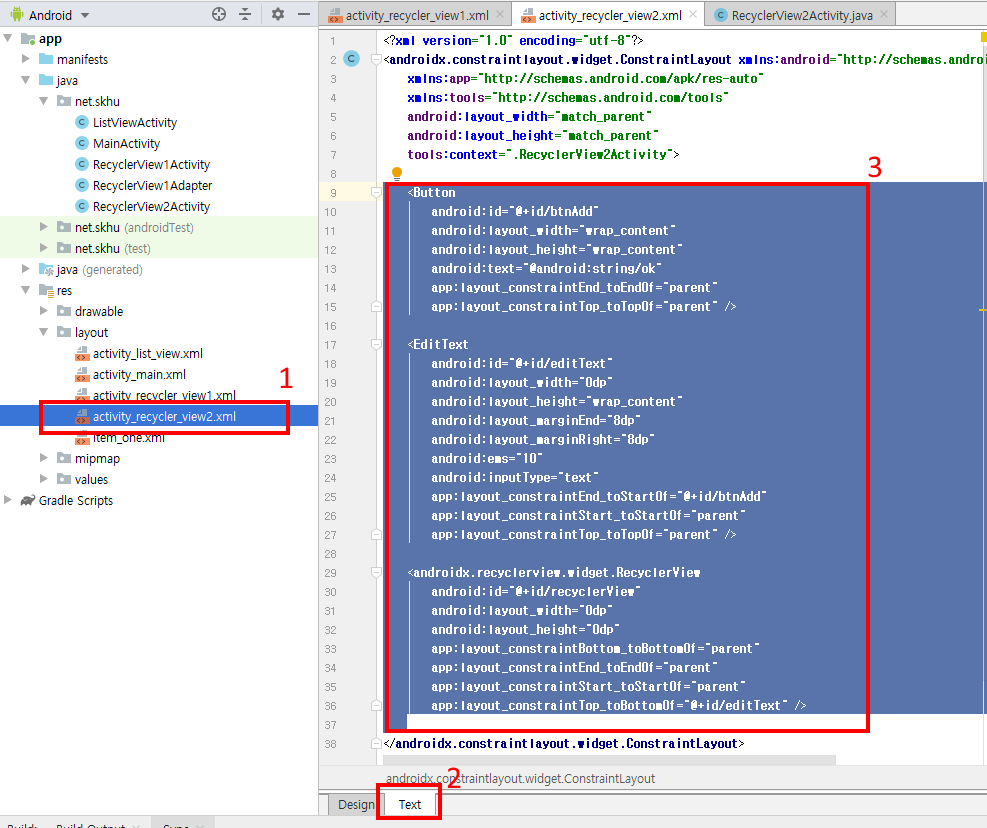
따라서 activity\_recycler\_view1.xml 파일의 소스코드를 복사해서 붙여 넣자.

### activity\_recycler\_view1.xml 에서 복사하기



activity\_recycler\_view1.xml 파일의 소스코드를 선택하고 Ctrl+C

### activity\_recycler\_view2.xml 에 붙여넣기



### activity\_recycler\_view2.xml 소스코드

|  |
| --- |
| *<?*xml version="1.0" encoding="utf-8"*?>*  <androidx.constraintlayout.widget.ConstraintLayout 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"  tools:context=".RecyclerView2Activity">  <Button  android:id="@+id/btnAdd"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="@android:string/ok"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <EditText  android:id="@+id/editText"  android:layout\_width="0dp"  android:layout\_height="wrap\_content"  android:layout\_marginEnd="8dp"  android:layout\_marginRight="8dp"  android:ems="10"  android:inputType="text"  app:layout\_constraintEnd\_toStartOf="@+id/btnAdd"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toTopOf="parent" />  <androidx.recyclerview.widget.RecyclerView  android:id="@+id/recyclerView"  android:layout\_width="0dp"  android:layout\_height="0dp"  app:layout\_constraintBottom\_toBottomOf="parent"  app:layout\_constraintEnd\_toEndOf="parent"  app:layout\_constraintStart\_toStartOf="parent"  app:layout\_constraintTop\_toBottomOf="@+id/editText" />  </androidx.constraintlayout.widget.ConstraintLayout> |

## RecyclerView 라이브러리 등록 확인하기

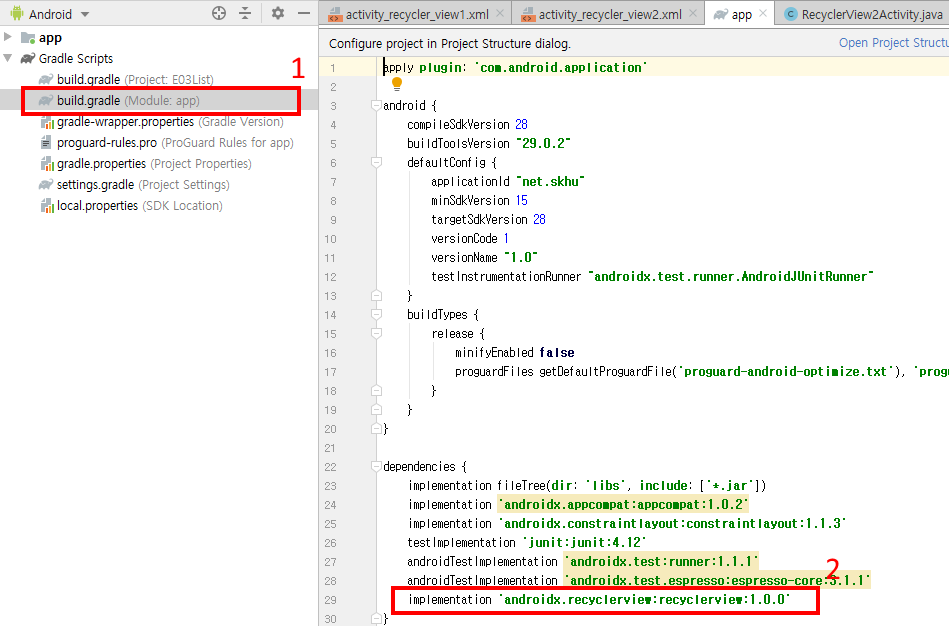
이전 화면에서 RecyclerView 객체를 드래그 드롭할 때,

RecyclerView 라이브러리가 프로젝트 설정 파일에 자동으로 등록되었다.

만약 제대로 추가되지 않았다면, 직접 등록해주어야 한다.

그 부분을 확인해 보자.

### app 모듈 build.gradle 파일



app 폴더 아래의 build.gradle 파일에서 dependencies 항목은

프로젝트에 필요한 라이브러리 목록이다.

RecyclerView를 사용하려면 필요한 라이브러리를 여기에 등록해야 한다.

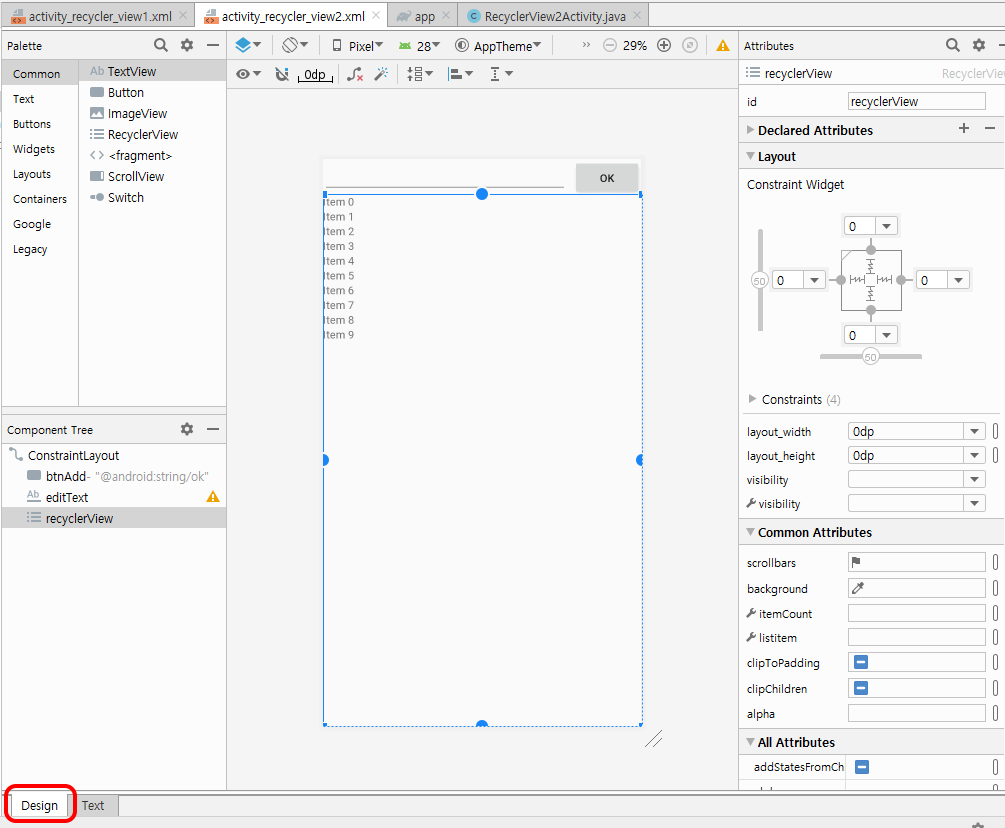
레이아웃 리소스의 디자인 모드에서 RecyclerView를 드래그 드롭하여 추가할 때는

필요한 라이브러리가 자동으로 등록된다.

그런데, 레이아웃 리소스 XML 파일의 소스코드에 RecyclerView 항목을 직접 입력하거나 붙여 넣을 때는

필요한 라이브러리가 자동으로 등록되지 않는다.

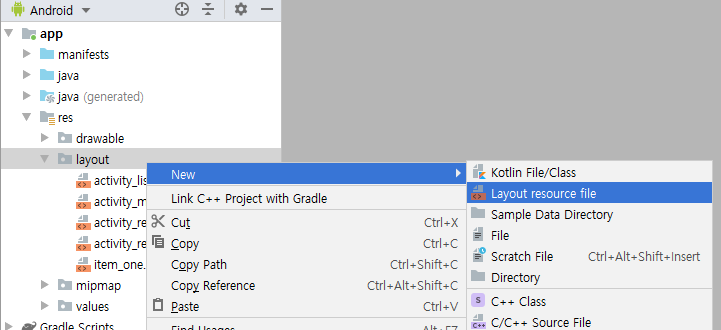
그래서 직접 등록해 주어야 한다.

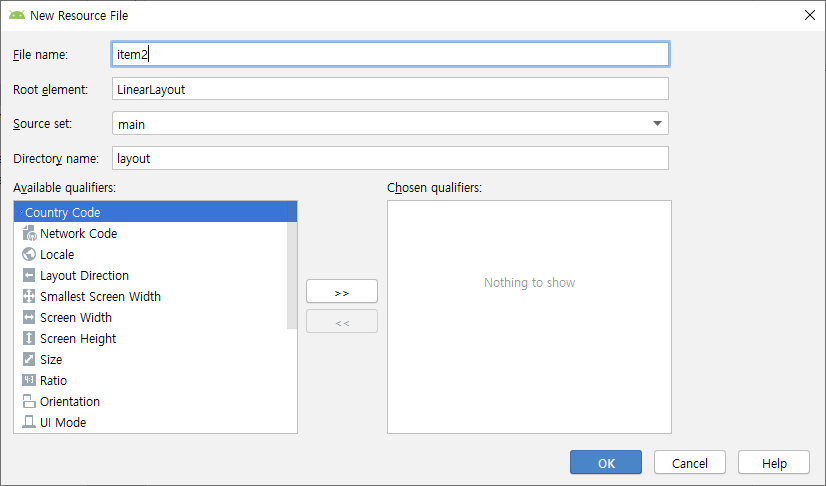


위 작업들을 잘 수행했다면, 위와 같이 보일 것이다.

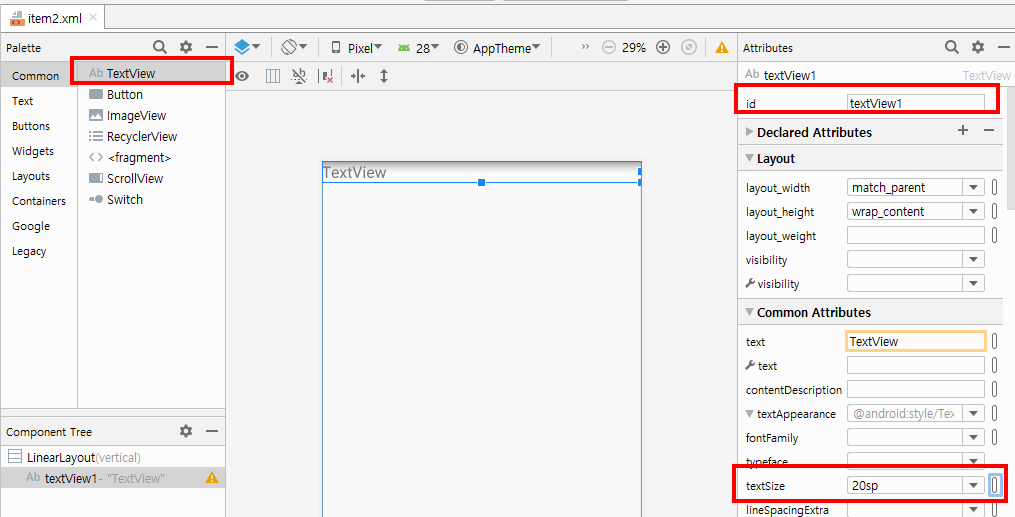
## item2.xml 파일 생성

데이터 항목 한 개에 대한 레이아웃 리소스 파일을 만들자.

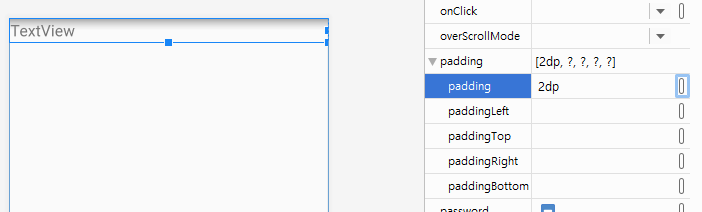




### TextView 추가

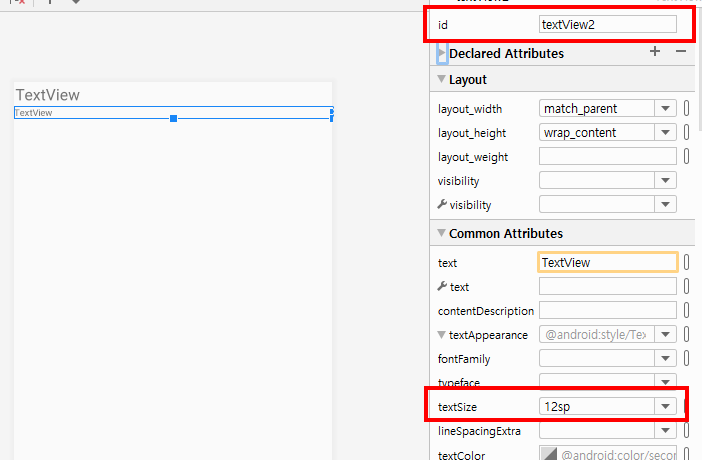


|  |  |
| --- | --- |
| 속성 | 값 |
| ID | textView1 |
| layout\_width | match\_parent |
| layout\_height | warp\_content |
| textSize | 20sp |

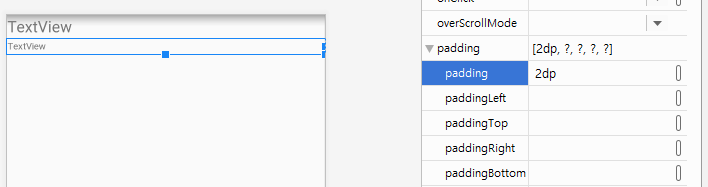


|  |  |
| --- | --- |
| 속성 | 값 |
| padding | 2dp |

### TextView 한개 더 추가

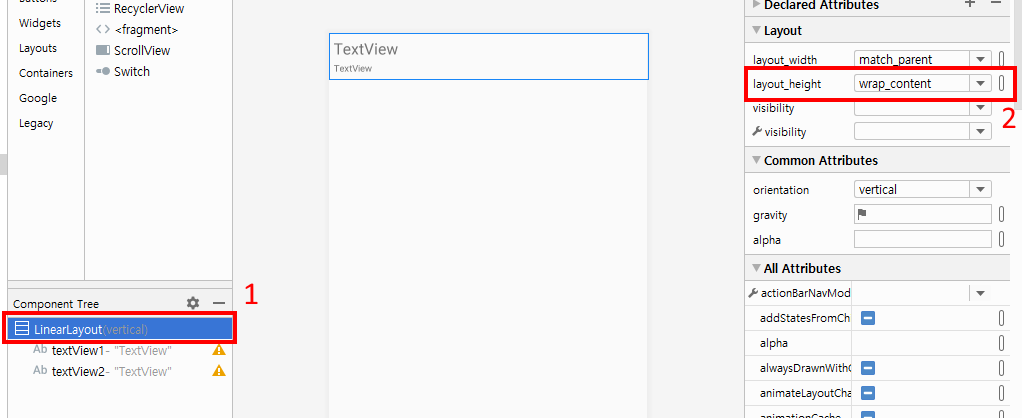


|  |  |
| --- | --- |
| 속성 | 값 |
| ID | textView2 |
| layout\_width | match\_parent |
| layout\_height | warp\_content |
| textSize | 12sp |

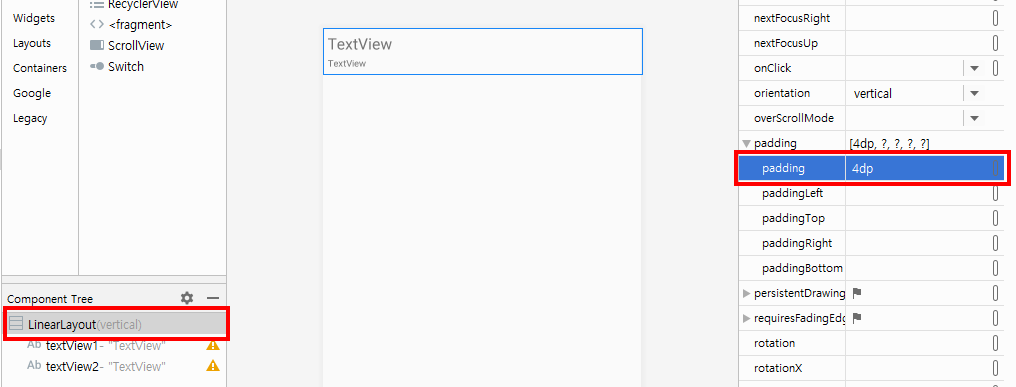


|  |  |
| --- | --- |
| 속성 | 값 |
| padding | 2dp |

### LinearLayout 속성 변경



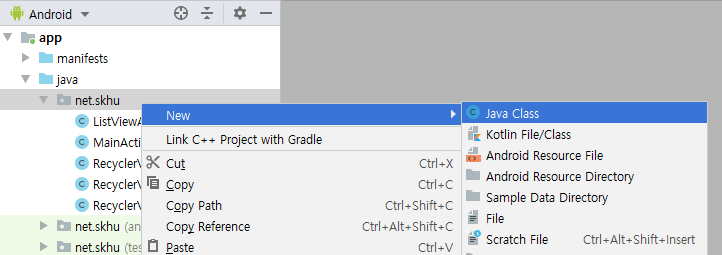
|  |  |
| --- | --- |
| 속성 | 값 |
| layout\_height | wrap\_content |



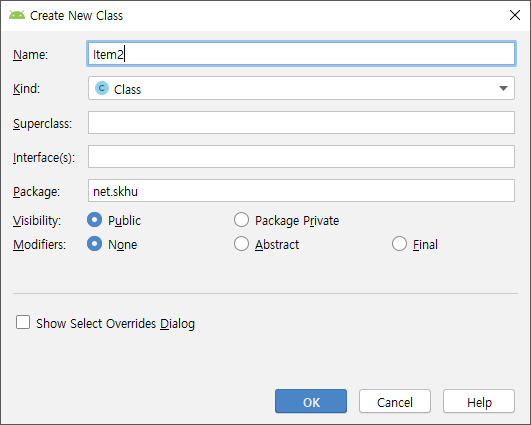
|  |  |
| --- | --- |
| 속성 | 값 |
| padding | 4dp |

## Item2.java 생성

데이터 항목에 해당하는 클래스를 구현하자.



메뉴: File - New - Java Class



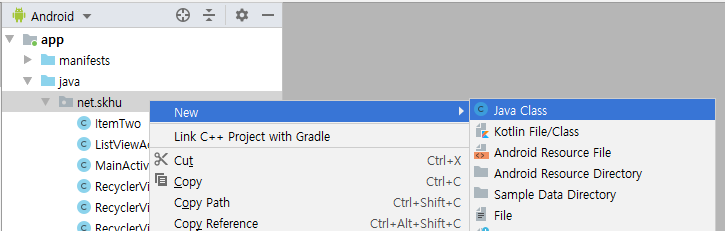
### Item2.java

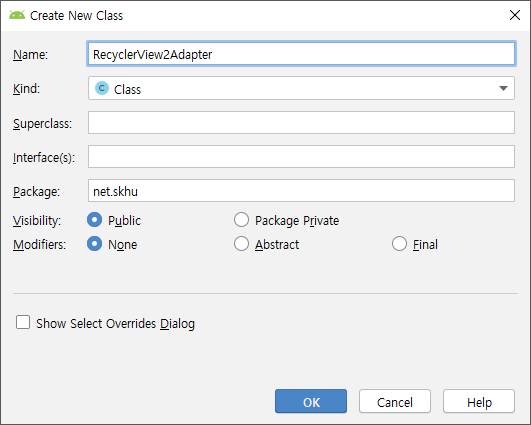
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | **package** net.skhu;  **import** java.text.SimpleDateFormat;  **import** java.util.Date;  **public class** Item2 {  **final static** SimpleDateFormat ***format*** = **new** SimpleDateFormat(**"yyyy-MM-dd HH:mm:ss"**);  String **title**;  Date **date**;  **public** Item2(String title, Date date) {  **this**.**title** = title;  **this**.**date** = date;  }  **public** String getTitle() {  **return title**;  }  **public** Date getDate() {  **return date**;  }  **public** String getDateFormatted() {  **return *format***.format(**date**);  }  } |

date 값으로부터 "2018-11-15 13:20:35" 형태의 문자열을 생성하기 위해

SimpleDateFormat 클래스를 사용하여 구현하였다.

## RecyclerView2Adapter.java 생성





### RecyclerView2Adapter.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52 | **package** net.skhu;  **import** android.content.Context;  **import** android.view.LayoutInflater;  **import** android.view.View;  **import** android.view.ViewGroup;  **import** android.widget.TextView;  **import** androidx.recyclerview.widget.RecyclerView;  **import** java.util.ArrayList;  **public class** RecyclerView2Adapter **extends** RecyclerView.Adapter<RecyclerView2Adapter.ViewHolder> {  **class** ViewHolder **extends** RecyclerView.ViewHolder {  TextView **textView1**, **textView2**;  **public** ViewHolder(View view) {  **super**(view);  **textView1** = view.findViewById(R.id.***textView1***);  **textView2** = view.findViewById(R.id.***textView2***);  }  **public void** setData() {  Item2 item = **arrayList**.get(getAdapterPosition());  **textView1**.setText(item.getTitle());  **textView2**.setText(item.getDateFormatted());  }  }  LayoutInflater **layoutInflater**;  ArrayList<Item2> **arrayList**;  **public** RecyclerView2Adapter(Context context, ArrayList<Item2> arrayList) {  **this**.**layoutInflater** = LayoutInflater.*from*(context);  **this**.**arrayList** = arrayList;  }  @Override  **public int** getItemCount() {  **return arrayList**.size();  }  @Override  **public** ViewHolder onCreateViewHolder(ViewGroup viewGroup, **int** viewType) {  View view = **layoutInflater**.inflate(R.layout.***item2***, viewGroup, **false**);  **return new** ViewHolder(view);  }  @Override  **public void** onBindViewHolder(**final** ViewHolder viewHolder, **final int** index) {  viewHolder.setData();  }  } |

지난 시간에 구현했던 RecyclerView2Adapter 소스코드와 비교해서 달라진 부분을 살펴보자.

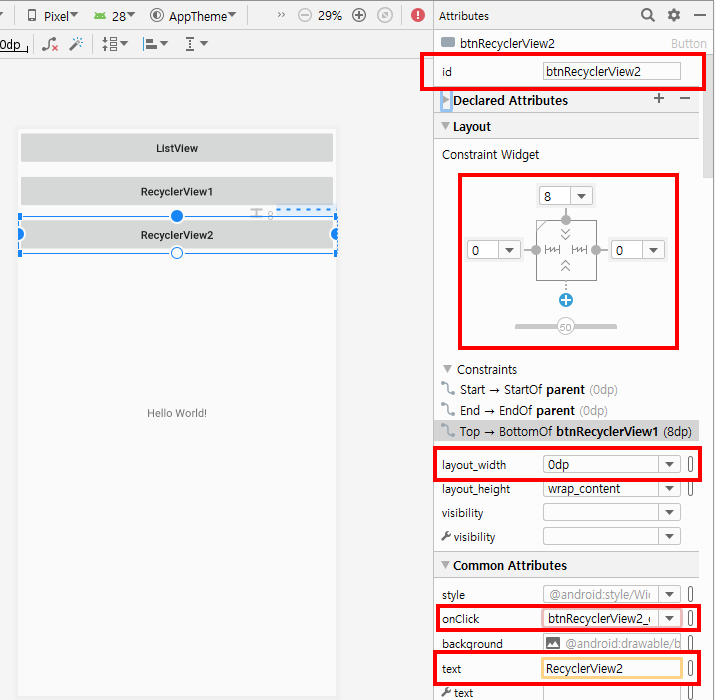
## RecyclerView2Activity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48 | **package** net.skhu;  **import** androidx.appcompat.app.AppCompatActivity;  **import** androidx.recyclerview.widget.DefaultItemAnimator;  **import** androidx.recyclerview.widget.DividerItemDecoration;  **import** androidx.recyclerview.widget.LinearLayoutManager;  **import** androidx.recyclerview.widget.RecyclerView;  **import** android.os.Bundle;  **import** android.view.View;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** java.util.ArrayList;  **import** java.util.Date;  **public class** RecyclerView2Activity **extends** AppCompatActivity {  RecyclerView2Adapter **recyclerView2Adapter**;  ArrayList<Item2> **arrayList**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_recycler\_view2***);  **arrayList** = **new** ArrayList<Item2>();  **arrayList**.add(**new** Item2(**"one"**, **new** Date()));  **arrayList**.add(**new** Item2(**"two"**, **new** Date()));  **recyclerView2Adapter** = **new** RecyclerView2Adapter(**this**, **arrayList**);  RecyclerView recyclerView = (RecyclerView) findViewById(R.id.***recyclerView***);  recyclerView.addItemDecoration(**new** DividerItemDecoration(**this**, DividerItemDecoration.***VERTICAL***));  recyclerView.setLayoutManager(**new** LinearLayoutManager(**this**));  recyclerView.setItemAnimator(**new** DefaultItemAnimator());  recyclerView.setAdapter(**recyclerView2Adapter**);  Button b = (Button)findViewById(R.id.***btnAdd***);  b.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View arg0) {  EditText e = (EditText) findViewById(R.id.***editText***);  String s = e.getText().toString();  e.setText(**""**);  **arrayList**.add(**new** Item2(s, **new** Date()));  **recyclerView2Adapter**.notifyDataSetChanged();  }  });  }  } |

지난 시간에 구현했던 RecyclerView1Activity 소스코드와 비교해서 달라진 부분을 살펴보자.

# MainActivity 수정

## activity\_main.xml 수정



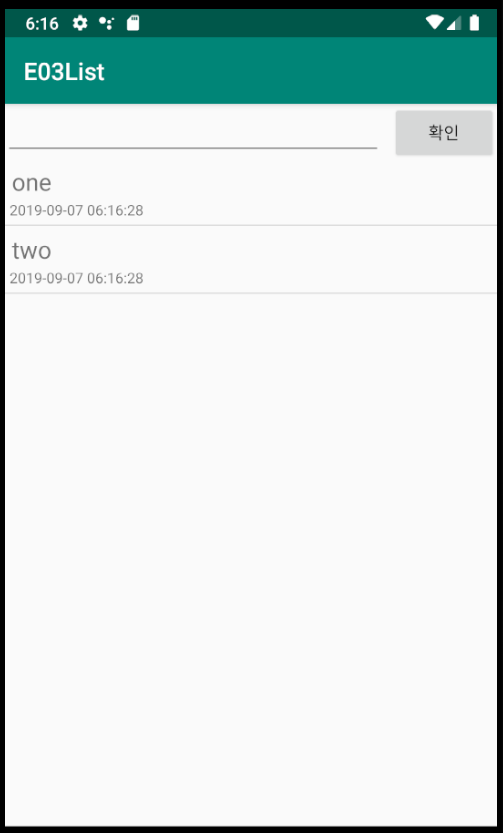
|  |  |
| --- | --- |
| id | btnRecyclerView2 |
| layout\_width | 0dp |
| onClick | btnRecyclerView2\_clicked |
| text | RecyclerView2 |

## MainActivity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | **package** net.skhu;  **import** androidx.appcompat.app.AppCompatActivity;  **import** android.content.Intent;  **import** android.os.Bundle;  **import** android.view.View;  **public class** MainActivity **extends** AppCompatActivity {  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_main***);  }  **public void** btnListView\_clicked(View view) {  Intent intent = **new** Intent(**this**, ListViewActivity.**class**);  startActivity(intent);  }  **public void** btnRecyclerView1\_clicked(View view) {  Intent intent = **new** Intent(**this**, RecyclerView1Activity.**class**);  startActivity(intent);  }  **public void** btnRecyclerView2\_clicked(View view) {  Intent intent = **new** Intent(**this**, RecyclerView2Activity.**class**);  startActivity(intent);  }  } |

## 실행

E03List\_src3.zip



# RecyclerView 항목 클릭 기능 구현

## RecyclerView2Adapter.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62 | **package** net.skhu;  **import** android.content.Context;  **import** android.view.LayoutInflater;  **import** android.view.View;  **import** android.view.ViewGroup;  **import** android.widget.TextView;  **import** android.widget.Toast;  **import** androidx.recyclerview.widget.RecyclerView;  **import** java.util.ArrayList;  **public class** RecyclerView2Adapter **extends** RecyclerView.Adapter<RecyclerView2Adapter.ViewHolder> {  **class** ViewHolder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener {  TextView **textView1**, **textView2**;  **public** ViewHolder(View view) {  **super**(view);  **textView1** = view.findViewById(R.id.***textView1***);  **textView2** = view.findViewById(R.id.***textView2***);  view.setOnClickListener(**this**);  }  **public void** setData() {  Item2 item = **arrayList**.get(getAdapterPosition());  **textView1**.setText(item.getTitle());  **textView2**.setText(item.getDateFormatted());  }  @Override  **public void** onClick(View view) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  String s = String.*format*(**"index: %d, title: %s"**, **super**.getAdapterPosition(), item.getTitle());  Toast.*makeText*(view.getContext(), s, Toast.***LENGTH\_SHORT***).show();  }  }  LayoutInflater **layoutInflater**;  ArrayList<Item2> **arrayList**;  **public** RecyclerView2Adapter(Context context, ArrayList<Item2> arrayList) {  **this**.**layoutInflater** = LayoutInflater.*from*(context);  **this**.**arrayList** = arrayList;  }  @Override  **public int** getItemCount() {  **return arrayList**.size();  }  @Override  **public** ViewHolder onCreateViewHolder(ViewGroup viewGroup, **int** viewType) {  View view = **layoutInflater**.inflate(R.layout.***item2***, viewGroup, **false**);  **return new** ViewHolder(view);  }  @Override  **public void** onBindViewHolder(**final** ViewHolder viewHolder, **final int** index) {  viewHolder.setData();  }  } |

### ViewHolder 클래스에 View.OnClickListener 인터페이스를 구현한 이유

ViewHolder 클래스에 View.OnClickListener 인터페이스를 구현하였다.

지금까지 어노니머스 이너 클래스 문법을 사용하여 리스너 클래스를 구현했는데,

반드시 그래야 하는 것은 아니다.

리스너 클래스는 그냥 평범한 클래스로 구현해도 된다.

그리고 기존의 클래스들 중 하나가 리스너 클래스의 역할도 같이 담당해도 된다.

RecyclerView 에서 데이터 항목 리스너 객체는, 데이터 항목 뷰 객체 마다 하나씩 달려있어야 한다.

즉 RecyclerView 에서 데이터 항목 리스너 객체의 수는, 데이터 항목 뷰 객체의 수와 일치해야 한다.

데이터 항목 뷰 객체란, RecyclerView2Adapter 클래스의 onCreateViewHolder 메소드에서 레이아웃 인플레이션으로 생성된 뷰(view) 객체이다. (줄54)

|  |
| --- |
| View view = **layoutInflater**.inflate(R.layout.***item2***, viewGroup, **false**); |

이 데이터 항목 뷰 객체는 ViewHolder 객체에 들어있다. (줄55)

|  |
| --- |
| **return new** ViewHolder(view, **arrayList**); |

즉 데이터 항목 뷰 객체의 수와 ViewHolder 객체의 수가 일치한다.

ViewHolder 객체의 수 == 데이터 항목 뷰 객치의 수 == 데이터 항목 리스너 객체의 수

위 3 객체의 수는 동일해야 한다.

따라서 리스너 클래스를 새로 구현해서 리스너 객체를 새로 생성하는 것 보다

이미 생성된 ViewHolder 객체들이 리스너 객체의 역할도 같이 담당하는 편이 효율적이다.

### 기존의 클래스에 View.OnClickListener 인터페이스 구현하기

(1) implements View.OnClickListener 이 필요하다. (줄15)

|  |
| --- |
| **static class** ViewHolder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener { |

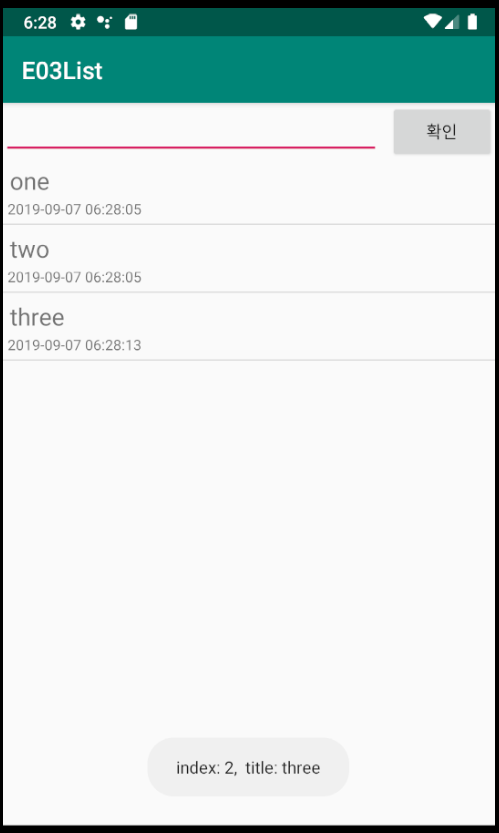
(2) onClick 메소드를 재정의해야 한다. (줄31)

|  |
| --- |
| @Override  **public void** onClick(View view) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  String s = String.*format*(**"index: %d, title: %s"**, **super**.getAdapterPosition(), item.getTitle());  Toast.*makeText*(view.getContext(), s, Toast.***LENGTH\_SHORT***).show();  } |

(3) 데이터 항목 뷰 객체에 리스너 객체를 등록해야 한다. (줄22)

|  |
| --- |
| view.setOnClickListener(**this**); |

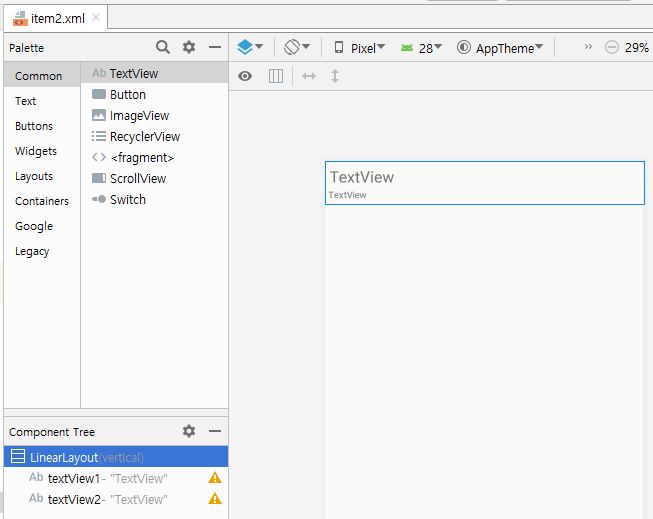
## 실행



# 삭제 기능 구현

## item2.xml

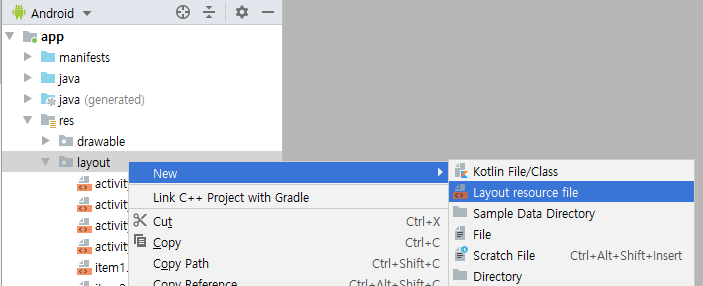
item2.xml 레이아웃 파일에서 루트 항목은 LinearLayout 이다.

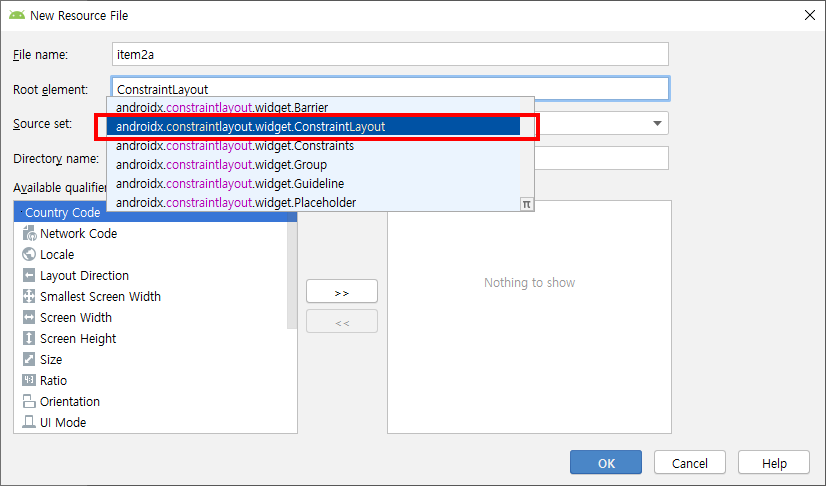


LinearLayout 객체가 루트 항목일 경우에, 자식 뷰 객체들을 일렬로 배치할 수만 있다.

그래서 TextView 오른쪽에 Checkbox를 추가하기 어렵다.

## item2a.xml 생성

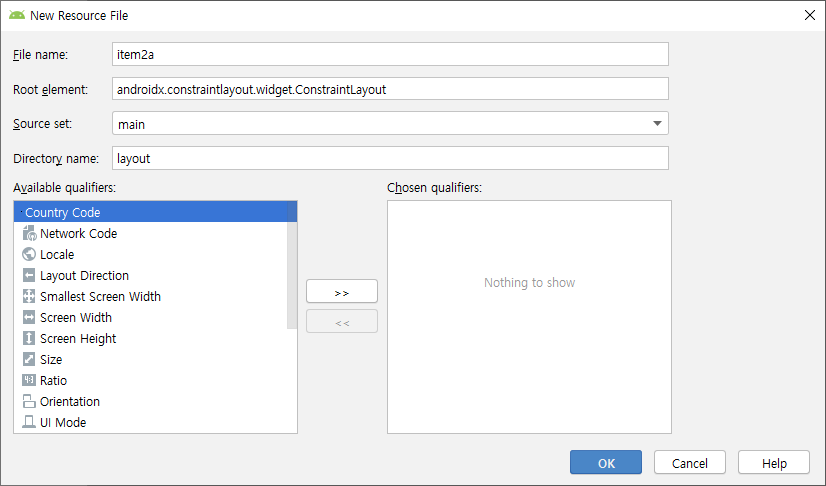




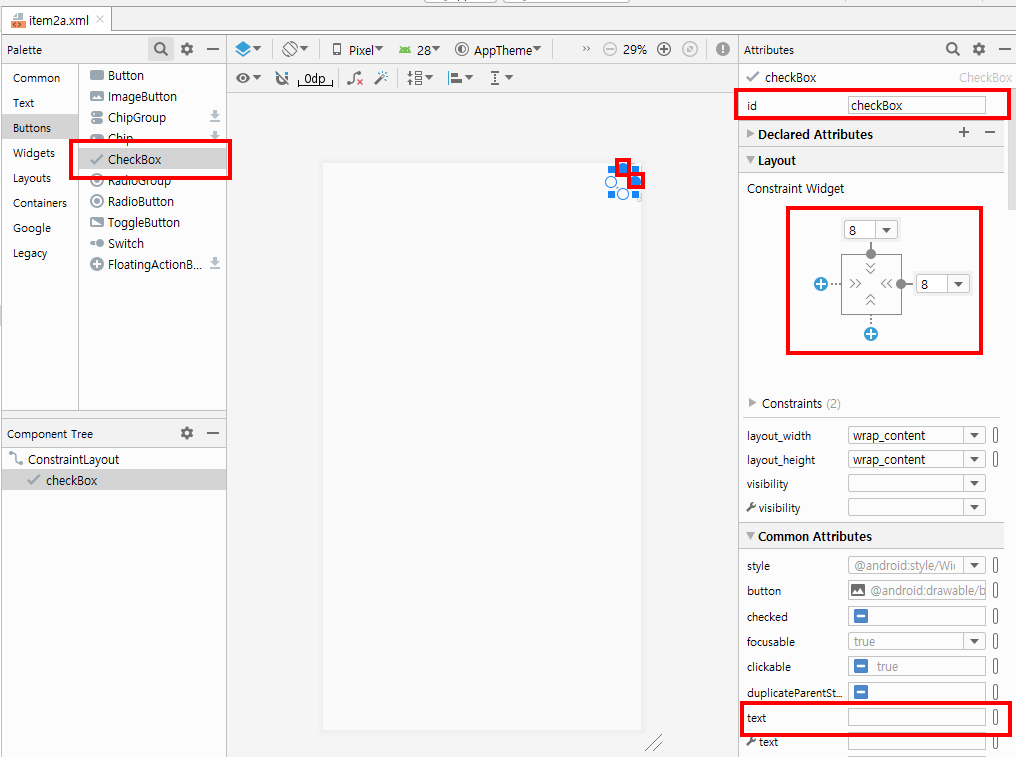
Root element 항목에 ConstraintLayout 이라고 입력하면,

아래 스크롤 목록에 androidx.constraintlayout.widget.ConstraintLayout 항목이 보인다.

이 항목을 선택하자.

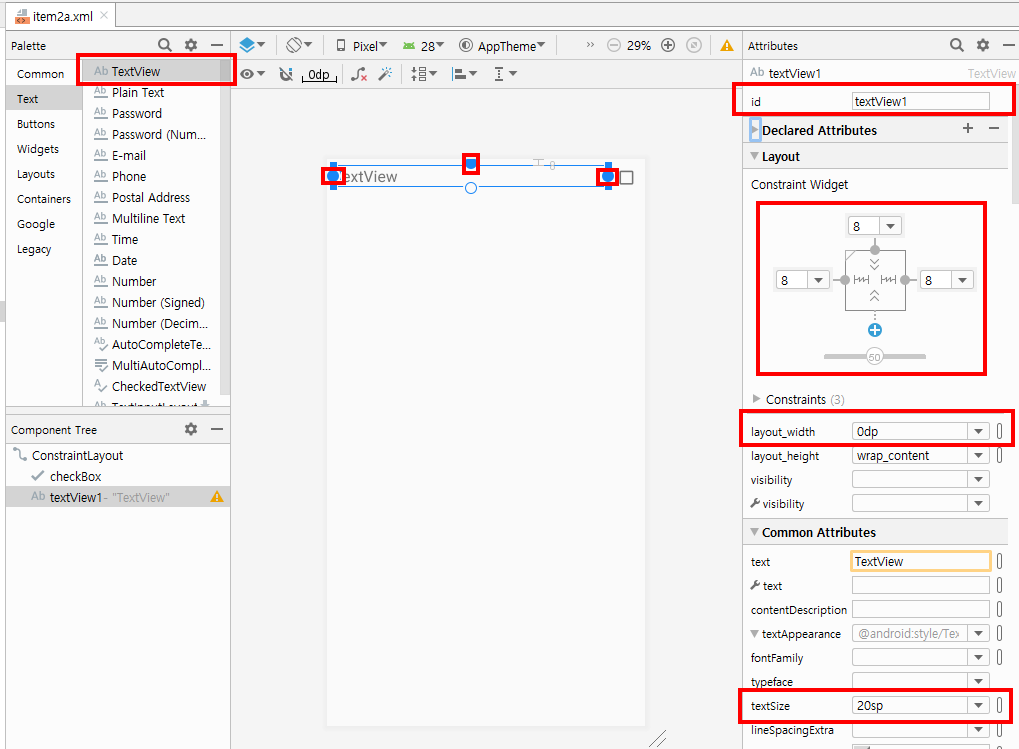


### CheckBox 추가



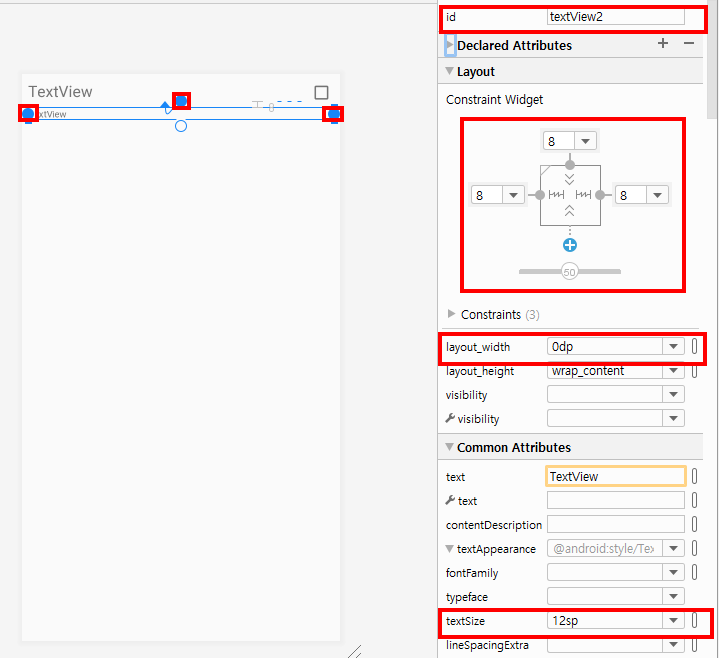
|  |  |
| --- | --- |
| 속성 | 값 |
| ID | checkBox |
| text |  |

### TextView 추가



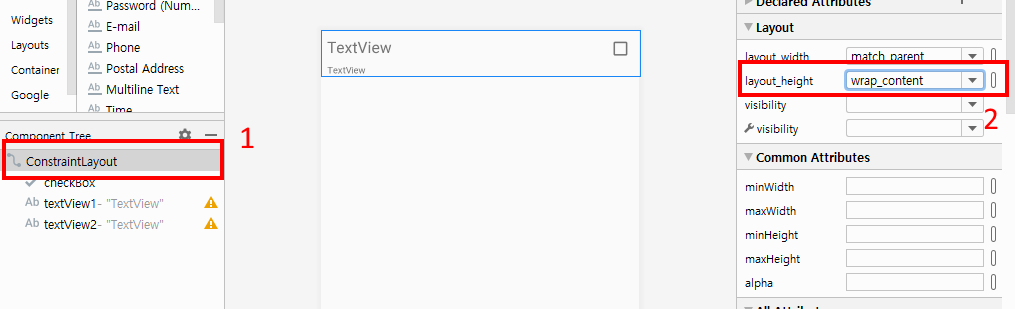
|  |  |
| --- | --- |
| 속성 | 값 |
| ID | textView1 |
| layout\_width | 0dp |
| textSize | 20sp |

### TextView 추가



|  |  |
| --- | --- |
| 속성 | 값 |
| ID | textView2 |
| layout\_width | 0dp |
| textSize | 12sp |

### ConstraintLayout 수정



## Item2.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | **package** net.skhu;  **import** java.text.SimpleDateFormat;  **import** java.util.Date;  **public class** Item2 {  **final static** SimpleDateFormat ***format*** = **new** SimpleDateFormat(**"yyyy-MM-dd HH:mm:ss"**);  String **title**;  Date **date**;  **boolean checked**;  **public** Item2(String title, Date date) {  **this**.**title** = title;  **this**.**date** = date;  }  **public** String getTitle() {  **return title**;  }  **public** Date getDate() {  **return date**;  }  **public** String getDateFormatted() {  **return *format***.format(**date**);  }  **public boolean** isChecked() { **return checked**; }  **public void** setChecked(**boolean** checked) { **this**.**checked** = checked; }  } |

앞에서 만든 체크박스의 상태 값을 저장할 멤버 변수를 구현하자.

boolean 타입의 get 메소드 이름은 get으로 시작해도 되지만, is로 시작하는 것이 더 바람직하다.

즉 getChecked/setChecked 이름도 가능하지만, isChecked/setChecked 이름이 더 바람직하다.

## RecyclerView2Adapter.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74 | **package** net.skhu;  **import** android.content.Context;  **import** android.view.LayoutInflater;  **import** android.view.View;  **import** android.view.ViewGroup;  **import** android.widget.CheckBox;  **import** android.widget.CompoundButton;  **import** android.widget.TextView;  **import** android.widget.Toast;  **import** androidx.recyclerview.widget.RecyclerView;  **import** java.util.ArrayList;  **public class** RecyclerView2Adapter **extends** RecyclerView.Adapter<RecyclerView2Adapter.ViewHolder> {  **class** ViewHolder **extends** RecyclerView.ViewHolder  **implements** View.OnClickListener, CompoundButton.OnCheckedChangeListener {  TextView **textView1**, **textView2**;  CheckBox **checkBox**;  **public** ViewHolder(View view) {  **super**(view);  **textView1** = view.findViewById(R.id.***textView1***);  **textView2** = view.findViewById(R.id.***textView2***);  **checkBox** = view.findViewById(R.id.***checkBox***);  **textView1**.setOnClickListener(**this**);  **checkBox**.setOnCheckedChangeListener(**this**);  }  **public void** setData() {  Item2 item = **arrayList**.get(getAdapterPosition());  **textView1**.setText(item.getTitle());  **textView2**.setText(item.getDateFormatted());  **checkBox**.setChecked(item.isChecked());  }  @Override  **public void** onClick(View view) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  String s = String.*format*(**"index: %d, title: %s"**, **super**.getAdapterPosition(), item.getTitle());  Toast.*makeText*(view.getContext(), s, Toast.***LENGTH\_SHORT***).show();  }  @Override  **public void** onCheckedChanged(CompoundButton buttonView, **boolean** isChecked) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  item.setChecked(isChecked);  }  }  LayoutInflater **layoutInflater**;  ArrayList<Item2> **arrayList**;  **public** RecyclerView2Adapter(Context context, ArrayList<Item2> arrayList) {  **this**.**layoutInflater** = LayoutInflater.*from*(context);  **this**.**arrayList** = arrayList;  }  @Override  **public int** getItemCount() {  **return arrayList**.size();  }  @Override  **public** ViewHolder onCreateViewHolder(ViewGroup viewGroup, **int** viewType) {  View view = **layoutInflater**.inflate(R.layout.***item2a***, viewGroup, **false**);  **return new** ViewHolder(view);  }  @Override  **public void** onBindViewHolder(**final** ViewHolder viewHolder, **final int** index) {  viewHolder.setData();  }  } |

### 체크 박스 리스너 구현

체크 박스를 클릭하여 체크 상태가 변경된 경우에, Item 데이터 항목 객체의 checked 속성에 이 값을 저장해야 한다.

그러려면 체크 박스에 OnCheckedChangedListener 객체를 등록해야 한다.

체크 박스의 수와 리스너 객체의 수가 정확히 일치해야 바람직하므로,

ViewHolder 객체가 리스너 객체의 역할도 동시에 담당하는 것이 바람직하다.

그래서 ViewHolder 클래스에 OnCheckedChangedListener 인터페이스를 구현하였다.

(줄28)

그리고 CheckBox 객체에 리스너 객체를 등록하였다.

CheckBox 객체의 상태가 변경되면, 리스너 객체의 onCheckedChanged 메소드가 자동으로 호출된다.

(줄27)

TextView 객체에 리스너 객체를 등록하였다.

TextView 객체가 클릭되면, 리스너 객체의 onClick 메소드가 자동으로 호출된다.

(줄67)

|  |
| --- |
| View view = **layoutInflater**.inflate(R.layout.***item2a***, viewGroup, **false**); |

item2a.xml 레이아웃 파일을 사용하여 뷰 객체들을 생성함.

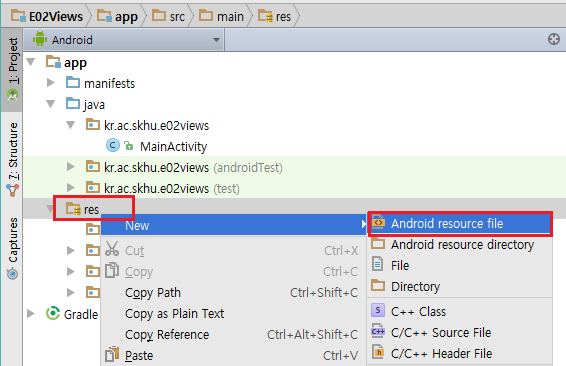
## 삭제 메뉴 구현

### app/res/values/strings.xml 수정

|  |
| --- |
| <**string name="action\_remove"**>삭제</**string**> |

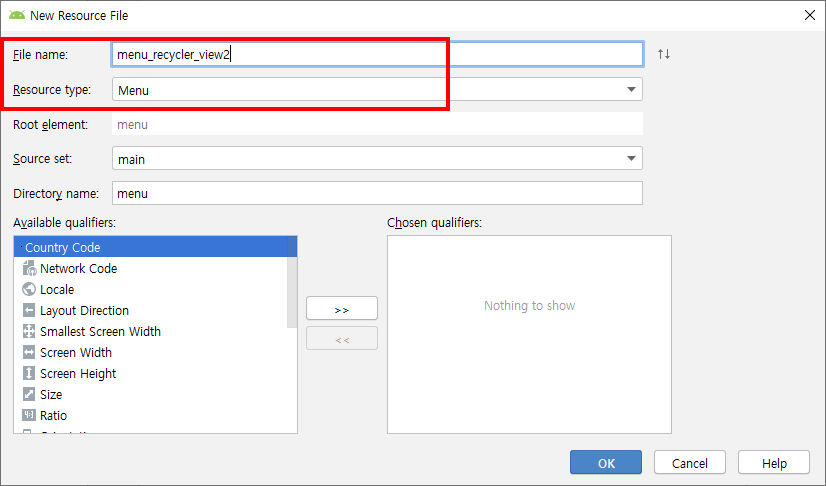
이 항목을 strings.xml 파일에 추가하자.

### 메뉴 리소스 파일 생성



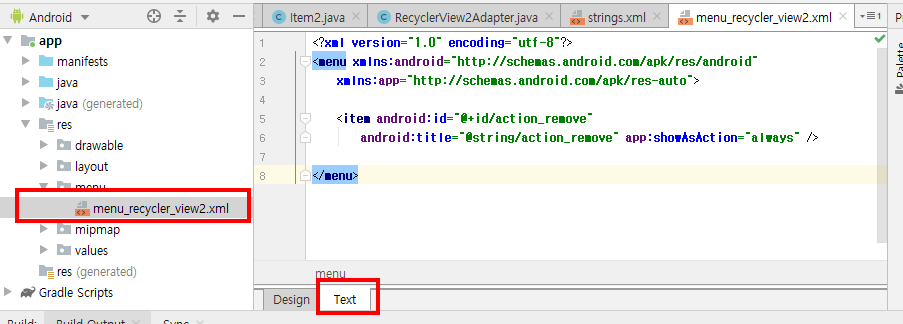
화면 왼쪽 Project 창에서 res 폴더를 마우스 오른쪽 버튼으로 클릭하고

메뉴에서 New - Android resource file 클릭



액티비티의 이름이 RecyvlerView2Activity 이므로, 메뉴 파일의 이름도 menu\_recycler\_view2 이라고 입력한다.

Ok 버튼 클릭.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | *<?*xml version="1.0" encoding="utf-8"*?>*  <menu xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto">  <item android:id="@+id/action\_remove"  android:title="@string/action\_remove" app:showAsAction="always" />  </menu> |

## RecyclerView2Activity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70 | **package** net.skhu;  **import** androidx.appcompat.app.AppCompatActivity;  **import** androidx.recyclerview.widget.DefaultItemAnimator;  **import** androidx.recyclerview.widget.DividerItemDecoration;  **import** androidx.recyclerview.widget.LinearLayoutManager;  **import** androidx.recyclerview.widget.RecyclerView;  **import** android.os.Bundle;  **import** android.view.Menu;  **import** android.view.MenuItem;  **import** android.view.View;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** java.util.ArrayList;  **import** java.util.Date;  **import** java.util.ListIterator;  **public class** RecyclerView2Activity **extends** AppCompatActivity {  RecyclerView2Adapter **recyclerView2Adapter**;  ArrayList<Item2> **arrayList**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_recycler\_view2***);  **arrayList** = **new** ArrayList<Item2>();  **arrayList**.add(**new** Item2(**"one"**, **new** Date()));  **arrayList**.add(**new** Item2(**"two"**, **new** Date()));  **recyclerView2Adapter** = **new** RecyclerView2Adapter(**this**, **arrayList**);  RecyclerView recyclerView = (RecyclerView) findViewById(R.id.***recyclerView***);  recyclerView.addItemDecoration(**new** DividerItemDecoration(**this**, DividerItemDecoration.***VERTICAL***));  recyclerView.setLayoutManager(**new** LinearLayoutManager(**this**));  recyclerView.setItemAnimator(**new** DefaultItemAnimator());  recyclerView.setAdapter(**recyclerView2Adapter**);  Button b = (Button)findViewById(R.id.***btnAdd***);  b.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View arg0) {  EditText e = (EditText) findViewById(R.id.***editText***);  String s = e.getText().toString();  e.setText(**""**);  **arrayList**.add(**new** Item2(s, **new** Date()));  **recyclerView2Adapter**.notifyDataSetChanged();  }  });  }  @Override  **public boolean** onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.***menu\_recycler\_view2***, menu);  **return true**;  }  @Override **public boolean** onOptionsItemSelected(MenuItem item) {  **int** id = item.getItemId();  **if** (id == R.id.***action\_remove***) {  ListIterator<Item2> iterator = **arrayList**.listIterator();  **while** (iterator.hasNext())  **if** (iterator.next().isChecked())  iterator.remove();  **recyclerView2Adapter**.notifyDataSetChanged();  **return true**;  }  **return super**.onOptionsItemSelected(item);  }  } |

### 선택된 항목 삭제하기 구현

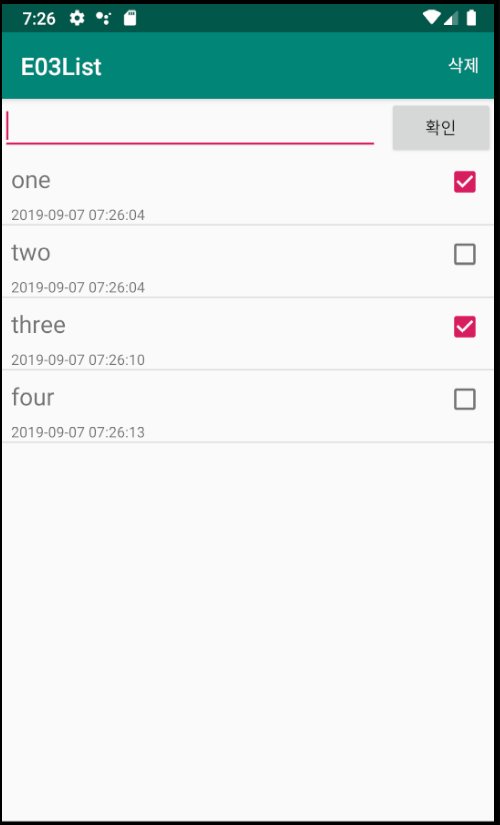
"삭제" 메뉴가 클릭되면, 아래 코드가 실행된다.

|  |
| --- |
| ListIterator<Item2> iterator = **arrayList**.listIterator();  **while** (iterator.hasNext())  **if** (iterator.next().isChecked())  iterator.remove();  **recyclerView2Adapter**.notifyDataSetChanged(); |

Java 언어의 ListIterator 인터페이스의 탐색 기능과 삭제 기능을 이용하여 구현하였다.

항목들을 삭제한 후 아답타 객체의 notifyDataSetChanged() 메소드를 호출해야 화면이 다시 그려진다.

## 실행



# 메뉴 상태 변경

체크된 항목이 있을 때에만, 삭제 메뉴가 표시되도록 수정하자.

## RecyclerView2Adapter.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82 | **package** net.skhu;  **import** android.app.Activity;  **import** android.content.Context;  **import** android.view.LayoutInflater;  **import** android.view.View;  **import** android.view.ViewGroup;  **import** android.widget.CheckBox;  **import** android.widget.CompoundButton;  **import** android.widget.TextView;  **import** android.widget.Toast;  **import** androidx.recyclerview.widget.RecyclerView;  **import** java.util.ArrayList;  **public class** RecyclerView2Adapter **extends** RecyclerView.Adapter<RecyclerView2Adapter.ViewHolder> {  **class** ViewHolder **extends** RecyclerView.ViewHolder  **implements** View.OnClickListener, CompoundButton.OnCheckedChangeListener {  TextView **textView1**, **textView2**;  CheckBox **checkBox**;  **public** ViewHolder(View view) {  **super**(view);  **textView1** = view.findViewById(R.id.***textView1***);  **textView2** = view.findViewById(R.id.***textView2***);  **checkBox** = view.findViewById(R.id.***checkBox***);  **view**.setOnClickListener(**this**);  **this**.**checkBox**.setOnCheckedChangeListener(**this**);  }  **public void** setData() {  Item2 item = **arrayList**.get(getAdapterPosition());  **textView1**.setText(item.getTitle());  **textView2**.setText(item.getDateFormatted());  **checkBox**.setChecked(item.isChecked());  }  @Override  **public void** onClick(View view) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  String s = String.*format*(**"index: %d, title: %s"**, **super**.getAdapterPosition(), item.getTitle());  Toast.*makeText*(view.getContext(), s, Toast.***LENGTH\_SHORT***).show();  }  @Override  **public void** onCheckedChanged(CompoundButton buttonView, **boolean** isChecked) {  Item2 item = **arrayList**.get(**super**.getAdapterPosition());  item.setChecked(isChecked);  **boolean** invalidateMenu = **false**;  **if** (isChecked) invalidateMenu = (++**checkedItemCount** == 1);  **else** invalidateMenu = (--**checkedItemCount** == 0);  **if** (invalidateMenu)  ((Activity) **textView1**.getContext()).invalidateOptionsMenu();  }  }  LayoutInflater **layoutInflater**;  ArrayList<Item2> **arrayList**;  **int checkedItemCount** = 0;  **public** RecyclerView2Adapter(Context context, ArrayList<Item2> arrayList) {  **this**.**layoutInflater** = LayoutInflater.*from*(context);  **this**.**arrayList** = arrayList;  }  @Override  **public int** getItemCount() {  **return arrayList**.size();  }  @Override  **public** ViewHolder onCreateViewHolder(ViewGroup viewGroup, **int** viewType) {  View view = **layoutInflater**.inflate(R.layout.***item2a***, viewGroup, **false**);  **return new** ViewHolder(view);  }  @Override  **public void** onBindViewHolder(**final** ViewHolder viewHolder, **final int** index) {  viewHolder.setData();  }  } |

|  |
| --- |
| **boolean** invalidateMenu = **false**;  **if** (isChecked) invalidateMenu = (++**checkedItemCount** == 1);  **else** invalidateMenu = (--**checkedItemCount** == 0);  **if** (invalidateMenu)  ((Activity) **textView1**.getContext()).invalidateOptionsMenu(); |

체크박스의 체크 상태가 변경될 때 마다 이 코드가 실행된다.

체크가 한 개 켜졌으면, checkItemCount 값을 ++ 하고,

한 개 꺼졌으면, checkItemCount 값을 -- 한다.

checkItemCount 값이 0 에서 1 로 변경된 경우이거나

1 에서 0 으로 변경된 경우에는 invalidateMenu 변수 값이 true 가 된다.

invalidateMenu 값이 true 이면,

액티비티의 메뉴가 다시 생성되도록 강제한다 (invalidateOptionMenu 메소드)

액티비티의 메뉴가 다시 생성될 때, 액티비티의 onCreateOptionsMenu 메소드가 자동으로 호출된다.

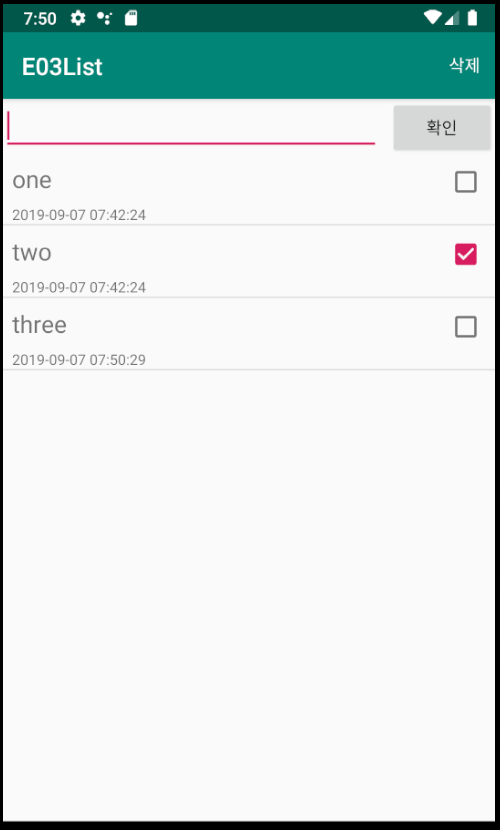
## RecyclerView2Activity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72 | **package** net.skhu;  **import** androidx.appcompat.app.AppCompatActivity;  **import** androidx.recyclerview.widget.DefaultItemAnimator;  **import** androidx.recyclerview.widget.DividerItemDecoration;  **import** androidx.recyclerview.widget.LinearLayoutManager;  **import** androidx.recyclerview.widget.RecyclerView;  **import** android.os.Bundle;  **import** android.view.Menu;  **import** android.view.MenuItem;  **import** android.view.View;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** java.util.ArrayList;  **import** java.util.Date;  **import** java.util.ListIterator;  **public class** RecyclerView2Activity **extends** AppCompatActivity {  RecyclerView2Adapter **recyclerView2Adapter**;  ArrayList<Item2> **arrayList**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_recycler\_view2***);  **arrayList** = **new** ArrayList<Item2>();  **arrayList**.add(**new** Item2(**"one"**, **new** Date()));  **arrayList**.add(**new** Item2(**"two"**, **new** Date()));  **recyclerView2Adapter** = **new** RecyclerView2Adapter(**this**, **arrayList**);  RecyclerView recyclerView = (RecyclerView) findViewById(R.id.***recyclerView***);  recyclerView.addItemDecoration(**new** DividerItemDecoration(**this**, DividerItemDecoration.***VERTICAL***));  recyclerView.setLayoutManager(**new** LinearLayoutManager(**this**));  recyclerView.setItemAnimator(**new** DefaultItemAnimator());  recyclerView.setAdapter(**recyclerView2Adapter**);  Button b = (Button)findViewById(R.id.***btnAdd***);  b.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View arg0) {  EditText e = (EditText) findViewById(R.id.***editText***);  String s = e.getText().toString();  e.setText(**""**);  **arrayList**.add(**new** Item2(s, **new** Date()));  **recyclerView2Adapter**.notifyDataSetChanged();  }  });  }  @Override  **public boolean** onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.***menu\_recycler\_view2***, menu);  MenuItem menuItem = menu.findItem(R.id.***action\_remove***);  menuItem.setVisible(**recyclerView2Adapter**.**checkedItemCount** > 0);  **return true**;  }  @Override **public boolean** onOptionsItemSelected(MenuItem item) {  **int** id = item.getItemId();  **if** (id == R.id.***action\_remove***) {  ListIterator<Item2> iterator = **arrayList**.listIterator();  **while** (iterator.hasNext())  **if** (iterator.next().isChecked())  iterator.remove();  **recyclerView2Adapter**.notifyDataSetChanged();  **return true**;  }  **return super**.onOptionsItemSelected(item);  }  } |

|  |
| --- |
| MenuItem menuItem = menu.findItem(R.id.***action\_remove***);  menuItem.setVisible(**recyclerView2Adapter**.**checkedItemCount** > 0); |

체크 박스가 체크된 항목이 1개 이상인 경우에만, 삭제 메뉴가 보이게 된다.

## 실행



# 삭제 확인 대화상자

## strings.xml 수정

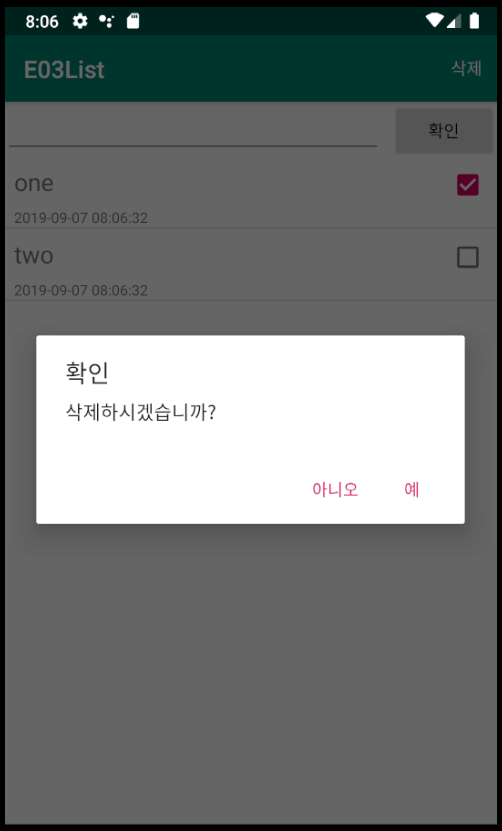
|  |
| --- |
| <**resources**>  <**string name="app\_name"**>E04Firebase</**string**>  <**string name="saveIntoServer"**>서버에 저장</**string**>  <**string name="action\_remove"**>삭제</**string**>  <**string name="confirm"**>확인</**string**>  <**string name="doYouWantToDelete"**>삭제하시겠습니까?</**string**>  <**string name="yes"**>예</**string**>  <**string name="no"**>아니오</**string**> </**resources**> |

## RecyclerView2Activity.java 수정

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92 | **package** net.skhu;  **import** androidx.appcompat.app.AlertDialog;  **import** androidx.appcompat.app.AppCompatActivity;  **import** androidx.recyclerview.widget.DefaultItemAnimator;  **import** androidx.recyclerview.widget.DividerItemDecoration;  **import** androidx.recyclerview.widget.LinearLayoutManager;  **import** androidx.recyclerview.widget.RecyclerView;  **import** android.content.DialogInterface;  **import** android.os.Bundle;  **import** android.view.Menu;  **import** android.view.MenuItem;  **import** android.view.View;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** android.widget.Toast;  **import** java.util.ArrayList;  **import** java.util.Date;  **import** java.util.ListIterator;  **public class** RecyclerView2Activity **extends** AppCompatActivity {  RecyclerView2Adapter **recyclerView2Adapter**;  ArrayList<Item2> **arrayList**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_recycler\_view2***);  **arrayList** = **new** ArrayList<Item2>();  **arrayList**.add(**new** Item2(**"one"**, **new** Date()));  **arrayList**.add(**new** Item2(**"two"**, **new** Date()));  **recyclerView2Adapter** = **new** RecyclerView2Adapter(**this**, **arrayList**);  RecyclerView recyclerView = (RecyclerView) findViewById(R.id.***recyclerView***);  recyclerView.addItemDecoration(**new** DividerItemDecoration(**this**, DividerItemDecoration.***VERTICAL***));  recyclerView.setLayoutManager(**new** LinearLayoutManager(**this**));  recyclerView.setItemAnimator(**new** DefaultItemAnimator());  recyclerView.setAdapter(**recyclerView2Adapter**);  Button b = (Button)findViewById(R.id.***btnAdd***);  b.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View arg0) {  EditText e = (EditText) findViewById(R.id.***editText***);  String s = e.getText().toString();  e.setText(**""**);  **arrayList**.add(**new** Item2(s, **new** Date()));  **recyclerView2Adapter**.notifyDataSetChanged();  }  });  }  @Override  **public boolean** onCreateOptionsMenu(Menu menu) {  getMenuInflater().inflate(R.menu.***menu\_recycler\_view2***, menu);  MenuItem menuItem = menu.findItem(R.id.***action\_remove***);  menuItem.setVisible(**recyclerView2Adapter**.**checkedItemCount** > 0);  **return true**;  }  @Override **public boolean** onOptionsItemSelected(MenuItem item) {  **int** id = item.getItemId();  **if** (id == R.id.***action\_remove***) {  deleteItems();  **return true**;  }  **return super**.onOptionsItemSelected(item);  }  **private void** deleteItems() {  AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  builder.setTitle(R.string.***confirm***);  builder.setMessage(R.string.***doYouWantToDelete***);  builder.setPositiveButton(R.string.***yes***, **new** DialogInterface.OnClickListener() {  @Override  **public void** onClick(DialogInterface dialog, **int** index) {  ListIterator<Item2> iterator = **arrayList**.listIterator();  **while** (iterator.hasNext())  **if** (iterator.next().isChecked())  iterator.remove();  **recyclerView2Adapter**.notifyDataSetChanged();  }  });  builder.setNegativeButton(R.string.***no***, **null**);  AlertDialog dialog = builder.create();  dialog.show();  }  } |

## 실행

E03List\_src4.zip



# 부록

ArrayList에서, 조건을 만족하는 항목들을 삭제하는 기능은

위 예제에서처럼 Iterator 탐색으로 구현하는 것이 안전하다.

|  |
| --- |
| for (Item2 item : arrayList)  if (item.isChecked())  arrayList.remove(item); |

체크된 항목 삭제를 위와같이 구현했을 때의 문제점은 무엇인가?

|  |
| --- |
| for (int i = 0; i < arrayList.size(); ++i)  if (arrayList.get(i).isChecked())  arrayList.remove(i); |

체크된 항목 삭제를 위와같이 구현했을 때의 문제점은 무엇인가?

|  |
| --- |
| for (int i = 0; i < arrayList.size(); ++i) {  Item2 item = arrayList.get(i);  if (item.isChecked())  arrayList.remove(item);  } |

체크된 항목 삭제를 위와같이 구현했을 때의 문제점은 무엇인가?