18.02.2020

Recycleview

Необходимо сделать прокручиваемый список объектов.

Под объектами понимаем рамка, в которой будет распологаться image и name (наименование страны в данном случае) (нарисовать картинку с планом).

На данный момент при запуске приложении на главном экране (activity\_main) будет располагаться кнопка,

при нажатии на которую будет открываться другая активити, в котороу будет отображен прокручиваемый список объектов.

1. Создаем класс ObjectCountry, который будет сохранять 2 переменный: ссылка на картинку и имя страны.
2. Создаем класс **Countries2**, который и будет запускаться при нажатии на кнопку из главной активити.
3. К этому классу создаем layout: **activity\_countries2**, в котором будет только лишь <androidx.recyclerview.widget.RecyclerView>, то есть в ней и будет прокрутка другой активити, которая и будет строить наш необходимый прямоугольник с картинкой и названием страны.
4. Создаем **activity\_list\_countries2**, это как раз активити, которая отображает нужные данные. В нем будет такая структура примерно:

<LianerLayout> - основная обертка

<ImageView> - картинка

<LeanerLayout> - второстеменная обертка

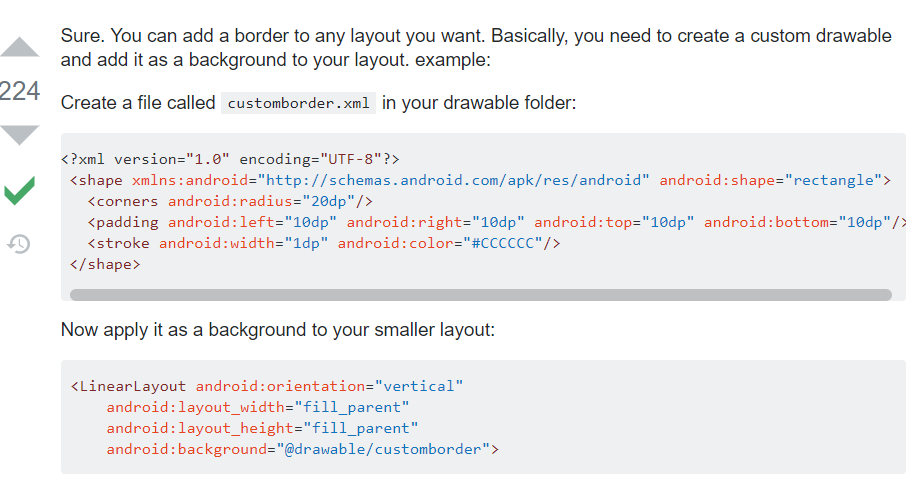
<TextView> - название страны

1. Создаем новый адаптер DataAdapter2

**Как сделать border к LinearLayout**

В папке **res/drawable** необходимо создать файл .xml и назовем его border. Он будет содержать рисунок рамки, который потом подключается к LinearLayout.

Ответ найден с stackoverflow: <https://stackoverflow.com/questions/15111402/how-can-i-create-a-border-around-an-android-linearlayout>



**Делаем переход на новую активити при нажатии на какой-либо объект**

Прям вот то что нужно: <https://codingwithmitch.com/blog/android-recyclerview-onclicklistener/>

Чтобы повесить событие на каждый элемент Recyclerview, то нам необходимо:

* В лайауте **activity\_list\_countries** на родительском LenearLayout проставить id 
* Далее все остальное делаем в классе адаптера **DataAdapter.**

- в методе ViewHolder обьявляем еще одну переменную, которая относится как раз к LinearLayout из шага выше 

- в конструкторе инициализируем его и приводим к LinearLayout 

* Далее идем в метод **onBindViewHolder**. К новой переменной пишем



Это пока что только лишь выводит в консоль “dfer2222”

**Нам необходимо при нажатии на любой item перейти на новую активити с информацией по нажатому item’му.**

1. Создаем новый layout и называем его AudioList. Делаем в нем TextView, который будет отображать имя страны, по которой прошел пользователь.

public class AudioList extends AppCompatActivity {  
  
 TextView t1, t2;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_audio\_list*);  
  
 t1 = (TextView) findViewById(R.id.*textView*);  
 t2 = (TextView) findViewById(R.id.*textView2*);  
  
 Bundle passedValue = getIntent().getExtras(); // получаем переданные значения в переменную типа Bundle  
 String nameCountry = passedValue.get("nameCountry").toString(); // По ключу "nameCountry" получаем значение  
  
 t2.setText(nameCountry);  
  
 }  
}

t2 – переменная типа TextView, которая будет отображать имя страны.

t2 = (TextView) findViewById(R.id.textView2); - этим действием мы инициализируем переменную.

Объявляем объект Bundle c переменной **passedValue**, которая принимает переданный объект.

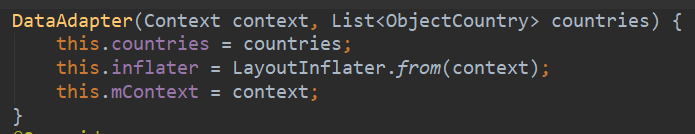
**Bundle passedValue = getIntent().getExtras();** // получаем переданные значения в переменную типа Bundle

**String nameCountry = passedValue.get("nameCountry").toString();** // По ключу "nameCountry" получаем значение

t2.setText(nameCountry); // устанавливаем значение необходимой страной

1. В классе DataAdapter первое, что мы сделаем, так это объявим новую перменную класса .

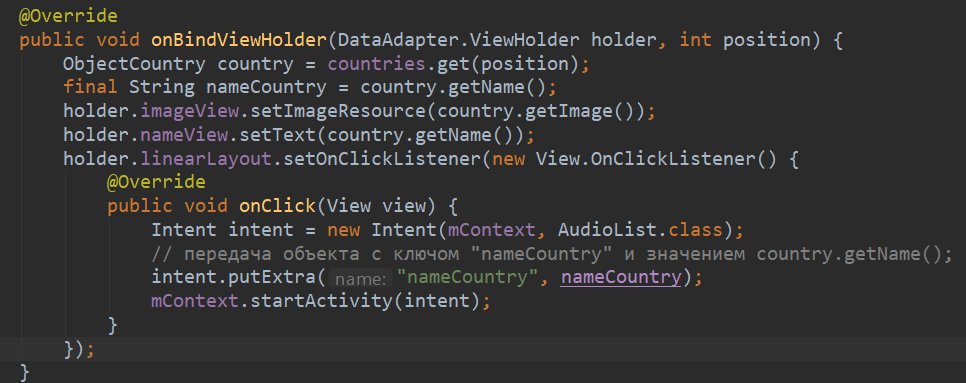
В конструкторе инициализируем ее.



Эта переменная необходима, для того чтобы брать значение текущего контекта в внутреннем классе, когда будем названачать на каждый итем

событие.

Далее, немного переделаем метод onClick, который выводил нам в консоль «dfer2222» таким образом:



Тут нужно обратить внимание, что мы добавили в методе новую переменную **final String nameCountry**, в ней мы сохраняем поле name

класса ObjectCountry, чтобы без проблем передать в метод onClick внутреннего (или ананимного? изучить) класса, а именно,

в строку intent.putExtra("nameCountry", nameCountry);

Кстати, вот в этом методе onClick нам и требуется новая добавленная отдельная переменная **mContext**, так как **this** тут не работает.

19.02.2020

to collate – сопостовлять

approach – подход

**Делаем заставку при запуске приложения, а иными словами это Splush Screen**

Заставка делается следующим образом. При запуске приложения первое, что будет открываться это activity\_main. Из нее и будем делать заставку, потому как эта активити с классом ничего не делают, кроме как имеет кнопку «Countries», при нажатии на которую мы и проходим к нужной активити с списком стран.

Если бы мы решили сделать отдельную активити с заставкой, после которой открывался бы наш текущий activity\_main, то нам необходимо было бы в файле манифесте

сделать изменения

<activity android:name=".MainActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
</activity>

то есть интент фильтры присвоить новой активити, так как она должна была бы запускаться первой.

В активити main делаем ImageView для картинки и TextView для текста под картинкой.

<?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"  
 android:orientation="vertical"  
 tools:context=".MainActivity">  
  
  
 <ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:src="@drawable/samsebetouristimage"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
  
 <TextView  
 android:id="@+id/textView3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="144dp"  
 android:text="Audio"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent" />  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Завтавка пока так себе, с ней поработать над дизайном, стилистикой и разметкой, но это потом. Пока это сойдет для программы, чтобы тестить и изучить как работает всё.

В классе MainActivity (ну тот класс, который связан с активити заставки):

public class MainActivity extends AppCompatActivity {  
  
 private final int SPLUSH\_SCREEN\_LENGHT = 2000;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 new Handler().postDelayed(new Runnable() {  
 @Override  
 public void run() {  
 Intent mainIntent = new Intent(MainActivity.this, Countries.class);  
 MainActivity.this.startActivity(mainIntent);  
 MainActivity.this.finish();  
 }  
 }, SPLUSH\_SCREEN\_LENGHT);  
 }  
  
}

тут добавляем переменную SPLUSH\_SCREEN\_LENGHT, которая будет int и которая будет содержать кол-во времени, которое будет работать заставка,

например 2000 это 2 секунды.

Далее

new Handler() – в общем изучить что он и кто он, а пока тут делается вот что.

При запуске этого класса метод **onCreate** запускает поток new Runnable, который после “;” начинает действовать, то есть начинает отсчет этих 2 секунд,

А в то же время наш метод **onCreate** идет дальше, то есть завершается, а, следовательно, создает activity с заставкой, которая будет отображаться, пока

метод run, который мы запустили не отработает. А в нем мы установили логику, которая запускает и открывает другую активити, с странами, а также

закрывает текущую активити (то есть заставку) методом finish(). Этот метод вызывается, когда вы хотите принудительно закрыть активити.

Это нужно (finish() ) для того, чтобы когда мы перешли на новую активити и нажали на системную кнопку «назад», то чтобы мы не вернулись к активити заставки, а просто вышли из

приложения (ну в данном случае), так как главная активити у нас считается Countries, а до нее якобы нет ничего.

**Нужно знать!**

Есть такой метод **onBackPressed,** который можно переопределить в любом классе активити. В нем можно выстроить логику того, что необходимо делать

при нажатии пользователем системной кнопки «назад»