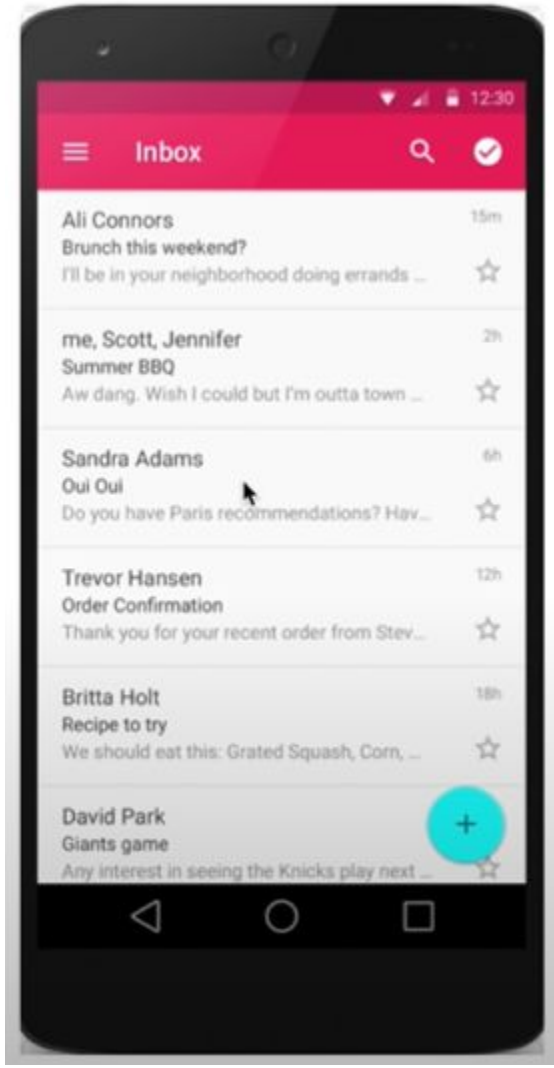
RecyclerView

RecyclerView vs TextView

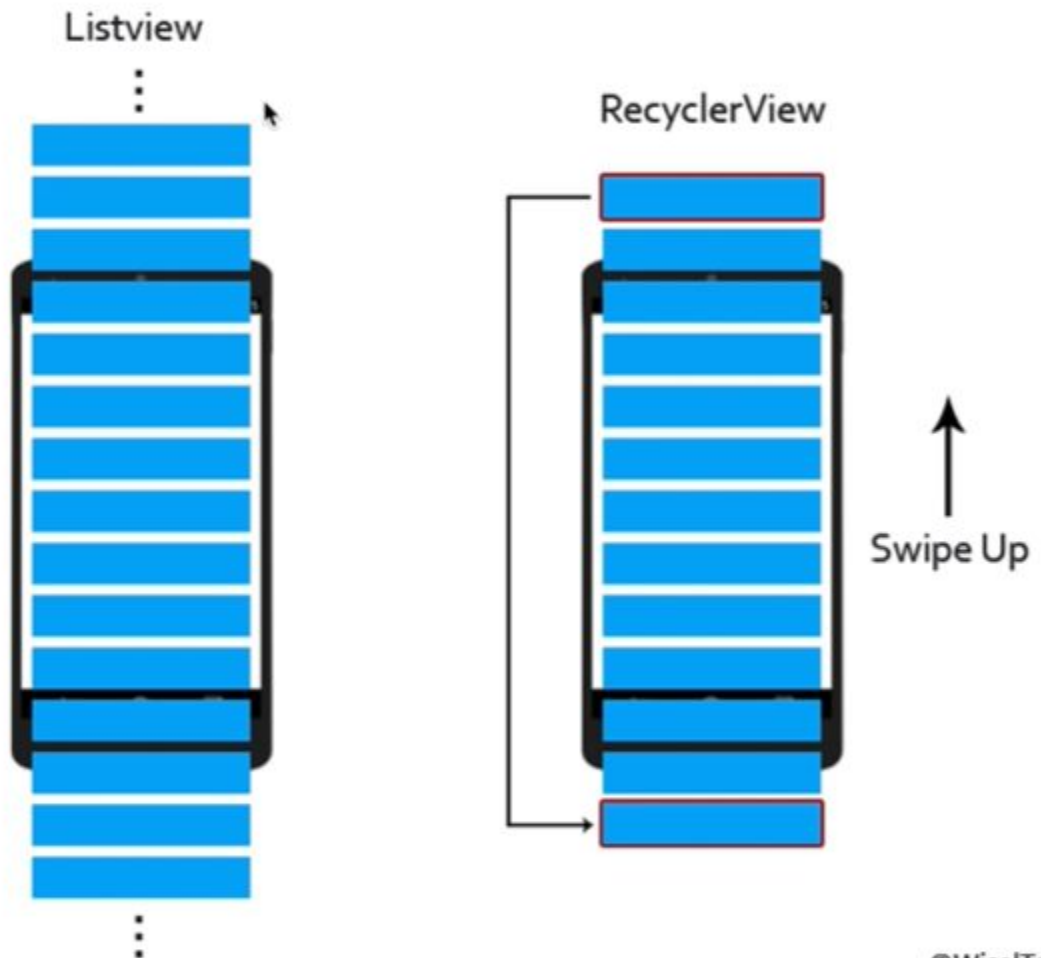


- **Вместо отображения всей информации в одном TextView, мы отображаем информацию в отдельных компонентах - ячейках.**
- **Можно работать с конкретным элементом списка (обработка нажатия)**
- **Эффективно**

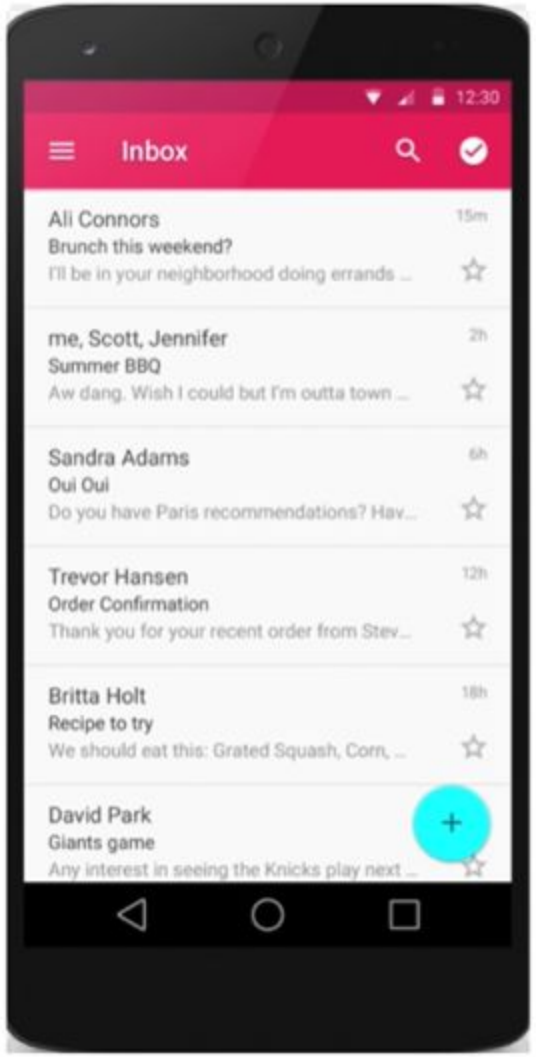
ListView vs. RecyclerView

- RecyclerView эффективней ListView так как повторно использует компоненты списка при прокрутке
- RecyclerView поддерживает как вертикальную, так и горизонтальную прокрутку. ListView поддерживает только вертикальную прокрутку.
- RecyclerView поддерживает не только списки (lists), но и сетки (grids). ListView поддерживает только списки (lists).
- RecyclerView предоставляет некоторые анимации для списков по-умолчанию. ListView нет.
- RecyclerView в целом более новый и продвинутый инструмент, который используется повсеместно. ListView устаревший.

Как работает RecyclerView?



Одни и те же компоненты используются повторно при прокрутке, что позволяет нам избегать задержек



- Создавать отдельный View (компонент) для каждого элемента в списке - это дорого (как по памяти, так и по быстродействию).
- ViewHolder - это элемент нашего списка. Но он может быть использован повторно (просто обновляются значения).
- Таким образом, RecyclerView создает столько ViewHolder'ов, чтобы заполнить экран устройства, плюс 2-4 дополнительных ViewHolder'ов.
- В ходе прокрутки списка, новый ViewHolder'ы не создаются. Вместо этого, старые ViewHolder'ы (которые ушли из поля зрения) обновляют свои значения.
- Поэтому RecyclerView и называется Recycler. Recycling (англ. переработка, повторное использование)

RecyclerView - лучший подход для отображения списков прокрутки на Android. Это обеспечивает высокую производительность и плавную прокрутку, предоставляя элементам списка гибкие макеты. В сочетании с современными языковыми возможностями Kotlin, накладные расходы на код RecyclerView значительно сокращаются по сравнению с традиционным подходом Java.

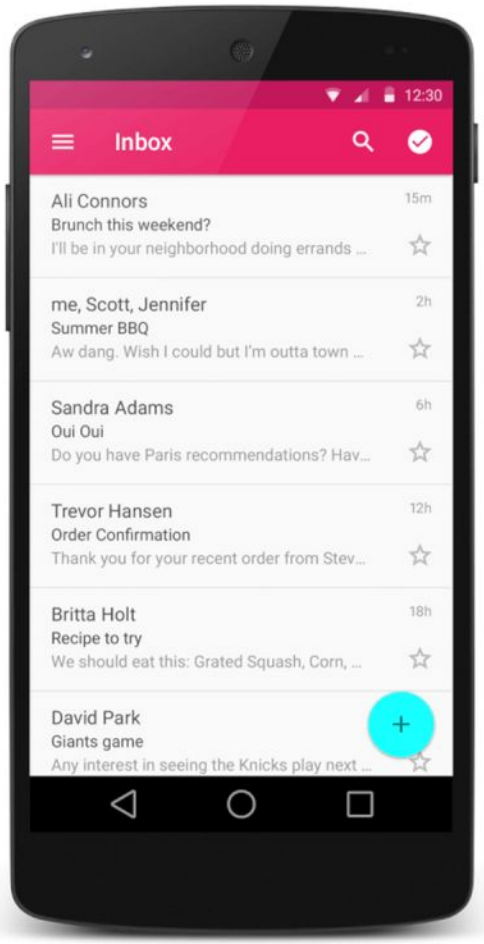
Экранный список в Android состоит из нескольких элементов. Каждый из них имеет макет с несколькими представлениями. Например, приложение электронной почты показывает вам несколько электронных писем; каждый из этих элементов включает тему, имя отправителя и множество другой информации.

Анализ определения XML-макета элемента и раздувание макета до экземпляров класса - дорогостоящая операция. При быстрой прокрутке это приводит к огромной нагрузке на производительность телефона. Цель состоит в том, чтобы всегда придерживаться 60 кадров в секунду. Однако это оставляет время вычисления менее 17 мс на кадр.

Главный трюк в *RecyclerView* заключается в том, что он повторно использует элементы просмотра списка. Как только элемент прокручивается за пределы видимой области, он помещается в очередь. В конце концов, он снова понадобится при прокрутке нового элемента. Затем содержимое в пользовательском интерфейсе элемента заменяются.

Следующий рисунок визуализирует этот (упрощенный) принцип *RecyclerView*:

RecyclerView



1. Экземпляр макета прокручивается вне поля зр

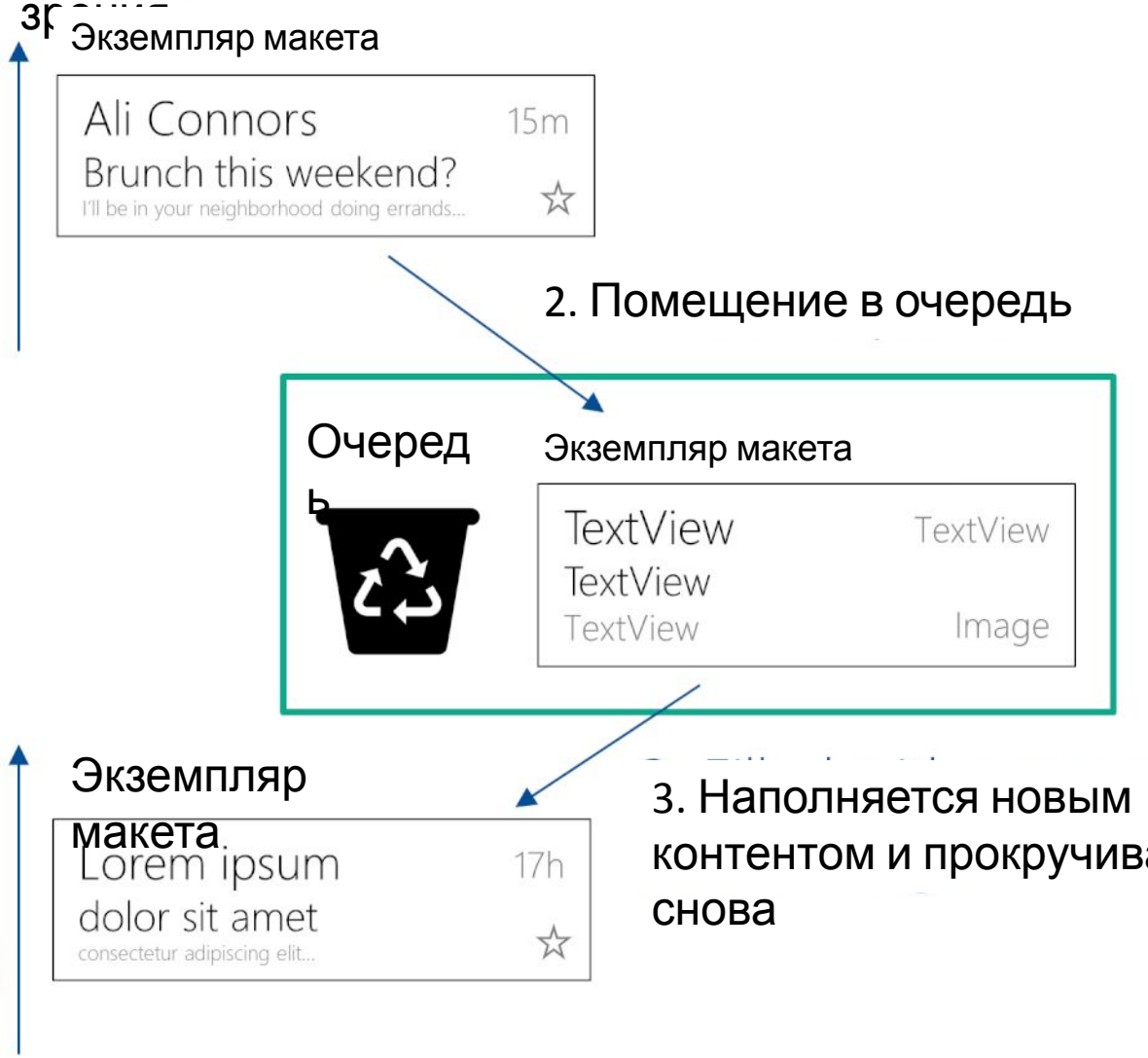
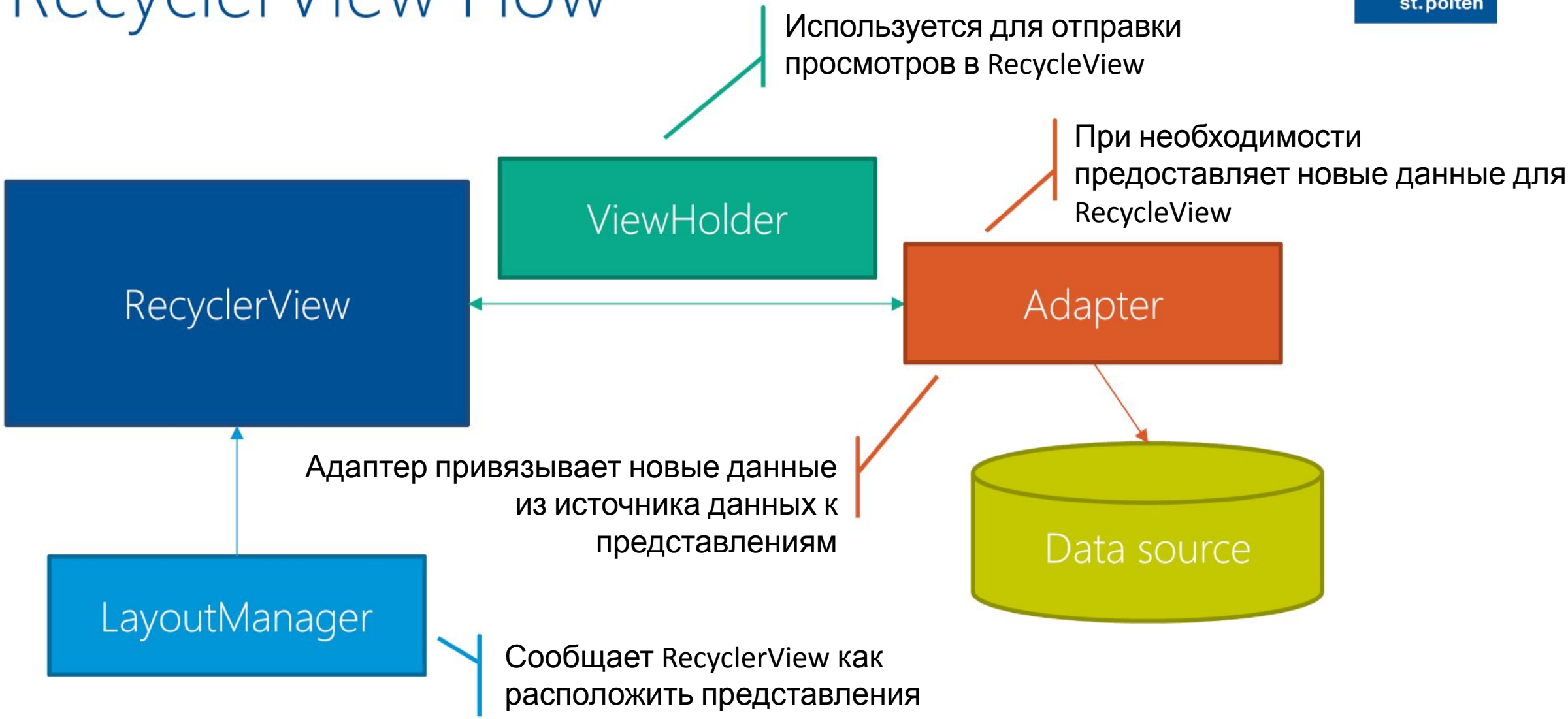


Image credits: Google, https://developer.android.com/training/material/lists-cards.html

RecyclerView Flow



Использование *RecyclerView* требует настройки / внедрения следующих компонентов:

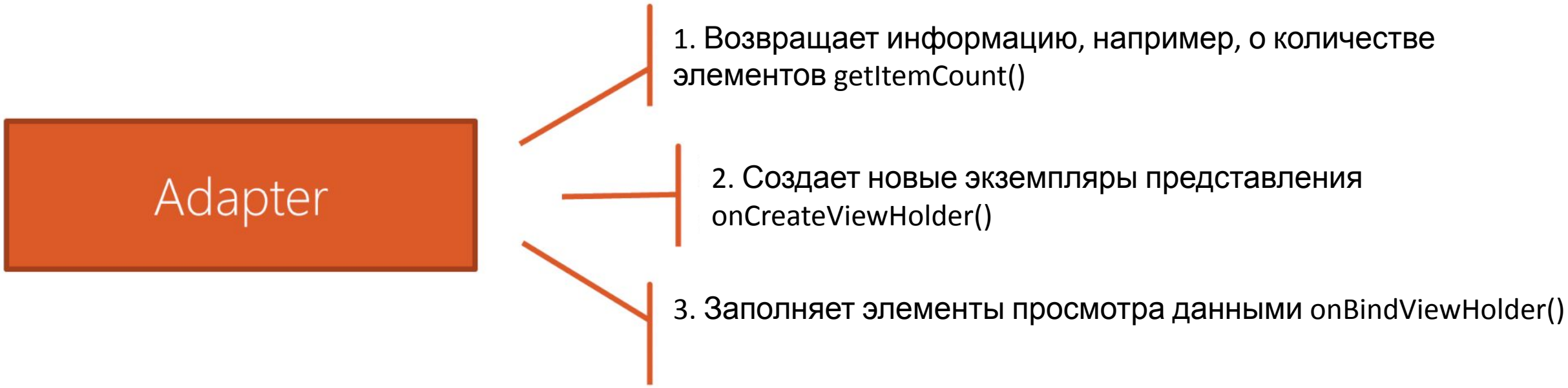
- **RecyclerView:** управляет всем. В основном это предварительно написано Android. Вы предоставляете компоненты и конфигурацию.

- **Адаптер:** вы потратите большую часть своего времени на кодирование этого класса. Он подключается к источнику данных. По указанию *RecyclerView* он создает или обновляет отдельные элементы в списке.

- **ViewHolder:** простой класс, который присваивает / обновляет данные в элементах просмотра. При повторном использовании представления предыдущие данные перезаписываются.

- **Источник данных:** все, что угодно – от простого массива до полноценного источника данных. Ваш адаптер взаимодействует с ним.

- **LayoutManager:** отвечает за размещение всех отдельных элементов просмотра на экране и за то, чтобы они занимали необходимое им пространство на экране.


Adapter
1. Возвращает информацию, например, о количестве элементов getItemCount()
2. Создает новые экземпляры представления onCreateViewHolder()
3. Заполняет элементы просмотра данными onBindViewHolder()

Adapter

1. Возвращает информацию, например, о количестве элементов getItemCount()

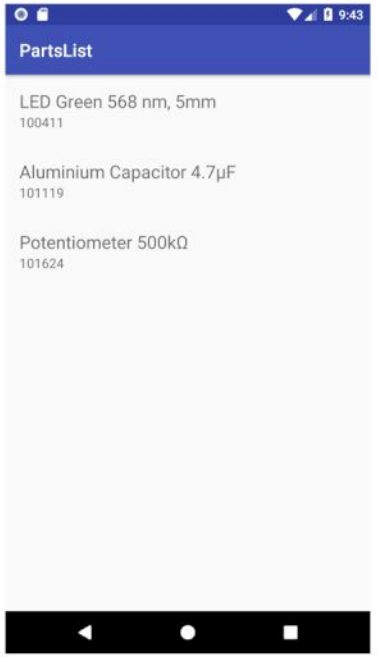
RecyclerView

Adapter

Data source

id	itemName
100411	LED Green 568 nm, 5mm
101119	Aluminium Capacitor 4.7µF
101629	Potentiometer 500kΩ

ViewHolder



2. Создает новые представления onCreateViewHolder()

3. Заполняет элементы просмотра данными onBindViewHolder()

Адаптер выполняет большую часть работы для RecyclerView – он подключает источник данных для просмотра элементов.

1. Для отображения списка на экране RecyclerView запросит у адаптера, сколько всего элементов. Наш адаптер возвращает эту информацию в getItemCount().

2. Всякий раз, когда RecyclerView решает, что ему нужен другой экземпляр view в памяти, он вызывает onCreateViewHolder(). В этом методе адаптер раздувает и возвращает макет XML, который мы создали на предыдущем шаге.

3. Каждый раз, когда ранее созданный ViewHolder используется (повторно), RecyclerView дает указание адаптеру обновить свои данные. Мы настраиваем этот процесс, переопределяя onBindViewHolder().

Вам не нужно писать код для переопределения этих трех функций вручную. Просто расширьте определение нашего класса-адаптера. Сделайте его наследуемым от RecyclerView Adapter. Android Studio автоматически предложит вам внедрить все необходимые элементы:

Пример использования RecyclerView

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help RecycleViewSample - build.gradle (:app)

RecycleViewSample > app > build.gradle

app | Pixel 4 XL API 33

Android

- app
  - manifests
  - java
    - com.example.recycleviewsample
      - MainActivity
    - com.example.recycleviewsample (androidTest)
    - com.example.recycleviewsample (test)
  - res
- Gradle Scripts
  - build.gradle (Project: RecycleViewSample)
  - build.gradle (Module :app)
  - proguard-rules.pro (ProGuard Rules for ":app")
  - gradle.properties (Project Properties)
  - gradle-wrapper.properties (Gradle Version)
  - local.properties (SDK Location)
  - settings.gradle (Project Settings)

activity_main.xml | MainActivity.kt | build.gradle (:app)

You can use the Project Structure dialog to view and edit your project configuration — Open (Ctrl+Alt+Shift+S) — Hide notification

```
}

android {
    namespace 'com.example.recycleviewsample'
    compileSdk 33

    defaultConfig {
        applicationId "com.example.recycleviewsample"
        minSdk 24
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
    buildFeatures{
        viewBinding = true
    }
}

dependencies {
```

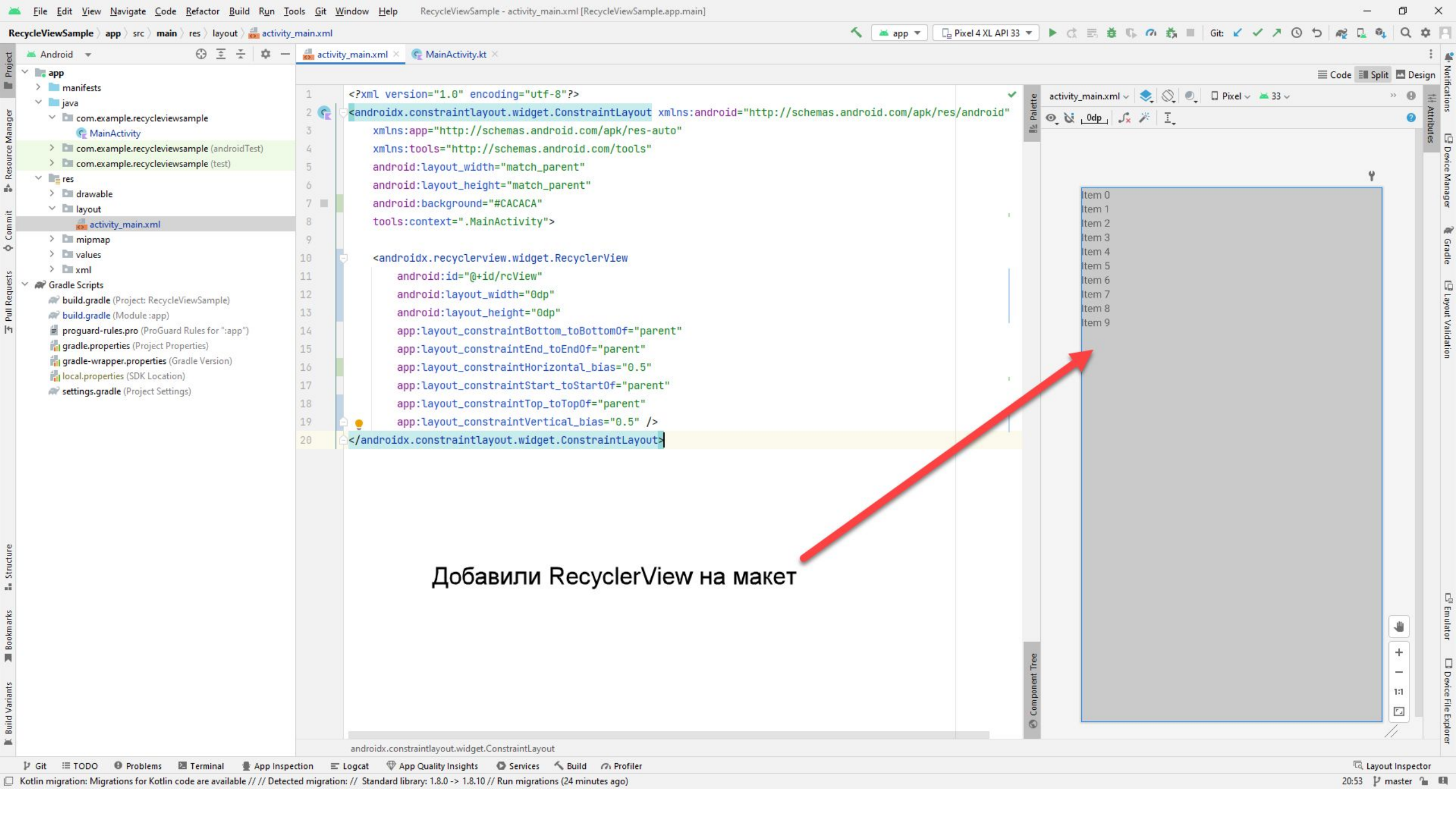
Внесем необходимые изменения в build.gradl (Module: app)

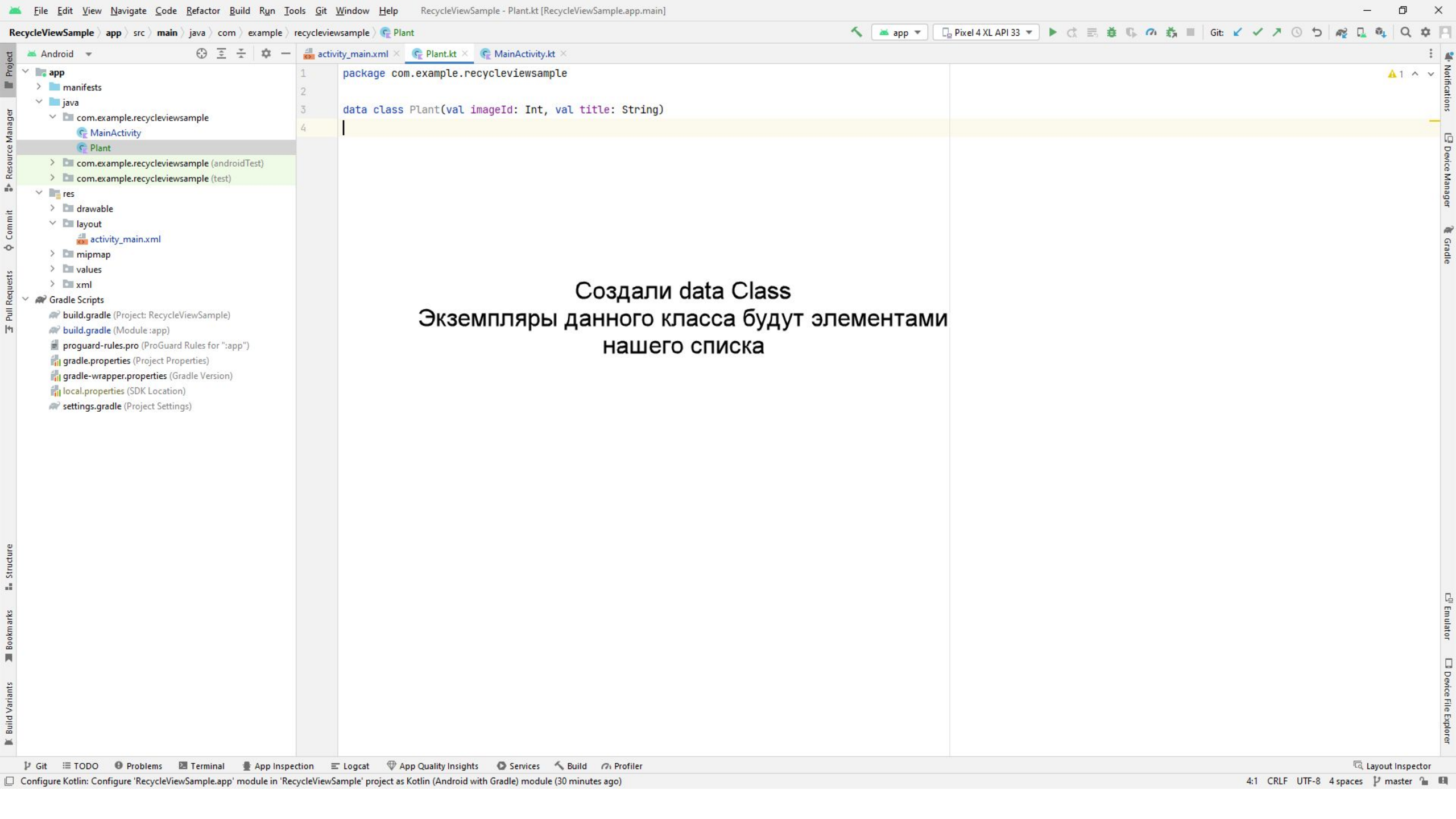
dependencies{}

Version Control | TODO | Problems | Terminal | App Inspection | Logcat | App Quality Insights | Services | Build | Profiler | Layout Inspector

Kotlin migration: Migrations for Kotlin code are available // // Detected migration: // Standard library: 1.8.0 -> 1.8.10 // Run migrations (a minute ago)

44:37 LF UTF-8 4 spaces


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - activity_main.xml [RecycleViewSample.app.main]
RecycleViewSample app src main res layout activity_main.xml
app
Pixel 4 XL API 33
Git:
Android
activity_main.xml
MainActivity.kt
Code Split Design
app
manifests
java
com.example.recycleviewsample
MainActivity
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
<?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:background="#CACACA"
tools:context=".MainActivity">
<androidx.recyclerview.widget.RecyclerView
android:id="@+id/rcView"
android:layout_width="0dp"
android:layout_height="0dp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.5"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.5" />
</androidx.constraintlayout.widget.ConstraintLayout>
Добавили RecyclerView на макет
Palette
Component Tree
Attributes
activity_main.xml
Pixel
33
0dp
Item 0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8
Item 9
androidx.constraintlayout.widget.ConstraintLayout
Git TODO Problems Terminal App Inspection Logcat App Quality Insights Services Build Profiler
Layout Inspector
Kotlin migration: Migrations for Kotlin code are available // // Detected migration: // Standard library: 1.8.0 -> 1.8.10 // Run migrations (24 minutes ago)
20:53 master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - Plant.kt [RecycleViewSample.app.main]
RecycleViewSample app src main java com example recycleviewsample Plant
app
Pixel 4 XL API 33
Git:
Android
activity_main.xml
Plant.kt
MainActivity.kt
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
1 package com.example.recycleviewsample
2
3 data class Plant(val imageId: Int, val title: String)
4
Создали data Class
Экземпляры данного класса будут элементами
нашего списка
Project
Resource Manager
Commit
Pull Requests
Structure
Bookmarks
Build Variants
Notifications
Device Manager
Gradle
Emulator
Device File Explorer
Git
TODO
Problems
Terminal
App Inspection
Logcat
App Quality Insights
Services
Build
Profiler
Layout Inspector
Configure Kotlin: Configure 'RecycleViewSample.app' module in 'RecycleViewSample' project as Kotlin (Android with Gradle) module (30 minutes ago)
4:1 CRLF UTF-8 4 spaces master

File Edit View Navigate Code Refactor Build Run Tools Git Window Help RecycleViewSample - Plant.kt [RecycleViewSample.app.main]

RecycleViewSample 〉 app 〉 src 〉 main 〉 res 〉 drawable

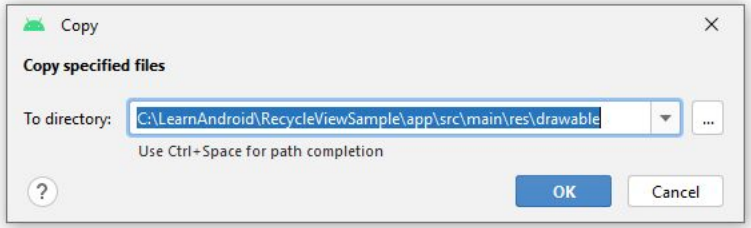
app Pixel 4 XL API 33 Git:

Android

- app
  - manifests
  - java
    - com.example.recycleviewsample
      - MainActivity
      - Plant
    - com.example.recycleviewsample (androidTest)
    - com.example.recycleviewsample (test)
  - res
    - drawable
      - ic_launcher_background.xml
      - ic_launcher_foreground.xml (v24)
    - layout
      - activity_main.xml
    - mipmap
    - values
    - xml
- Gradle Scripts
  - build.gradle (Project: RecycleViewSample)
  - build.gradle (Module :app)
  - proguard-rules.pro (ProGuard Rules for ":app")
  - gradle.properties (Project Properties)
  - gradle-wrapper.properties (Gradle Version)
  - local.properties (SDK Location)
  - settings.gradle (Project Settings)

activity_main.xml | Plant.kt | MainActivity.kt

```
1  package com.example.recycleviewsample
2
3  data class Plant(val imageId: Int, val title: String)
4
```

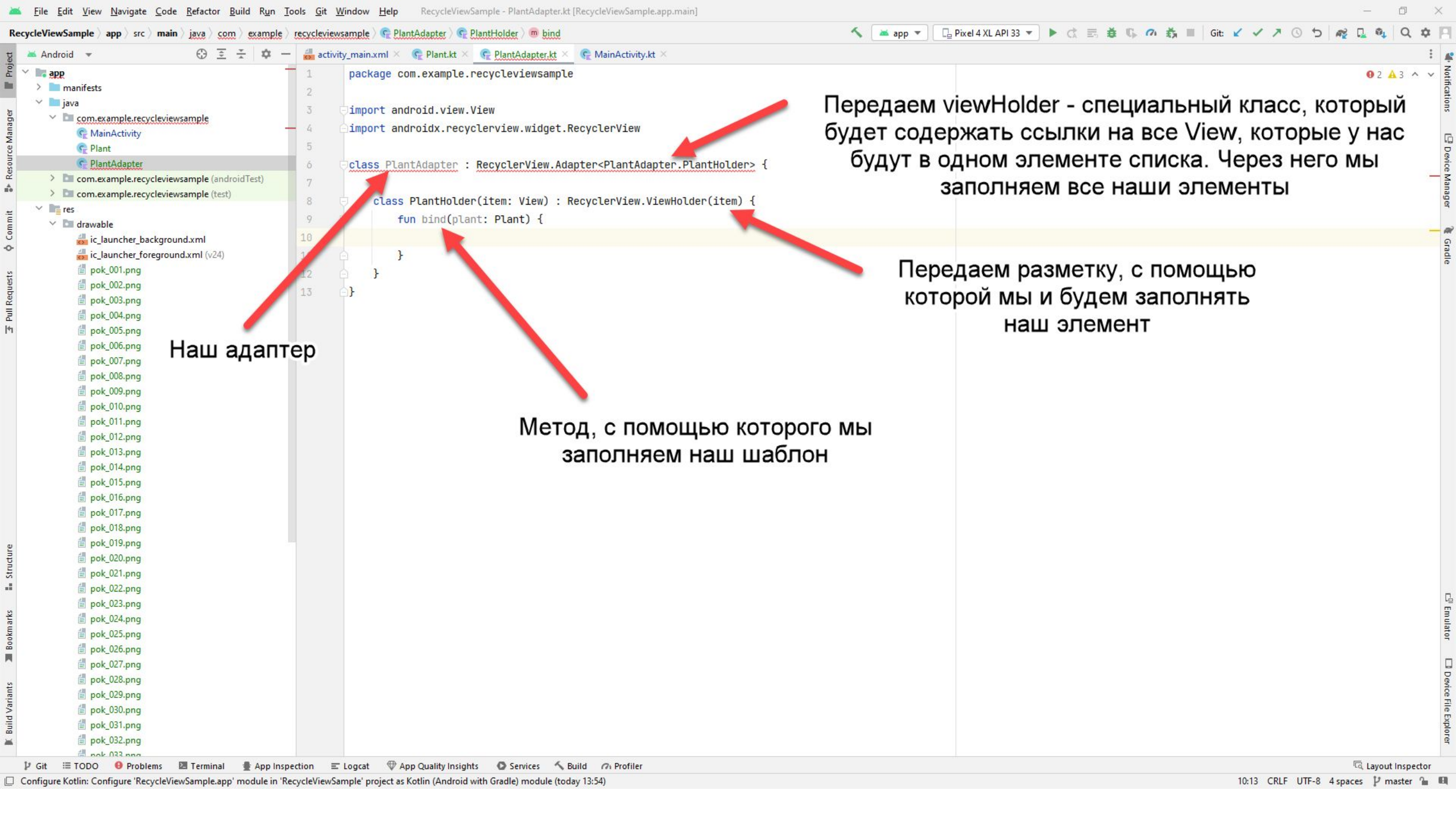


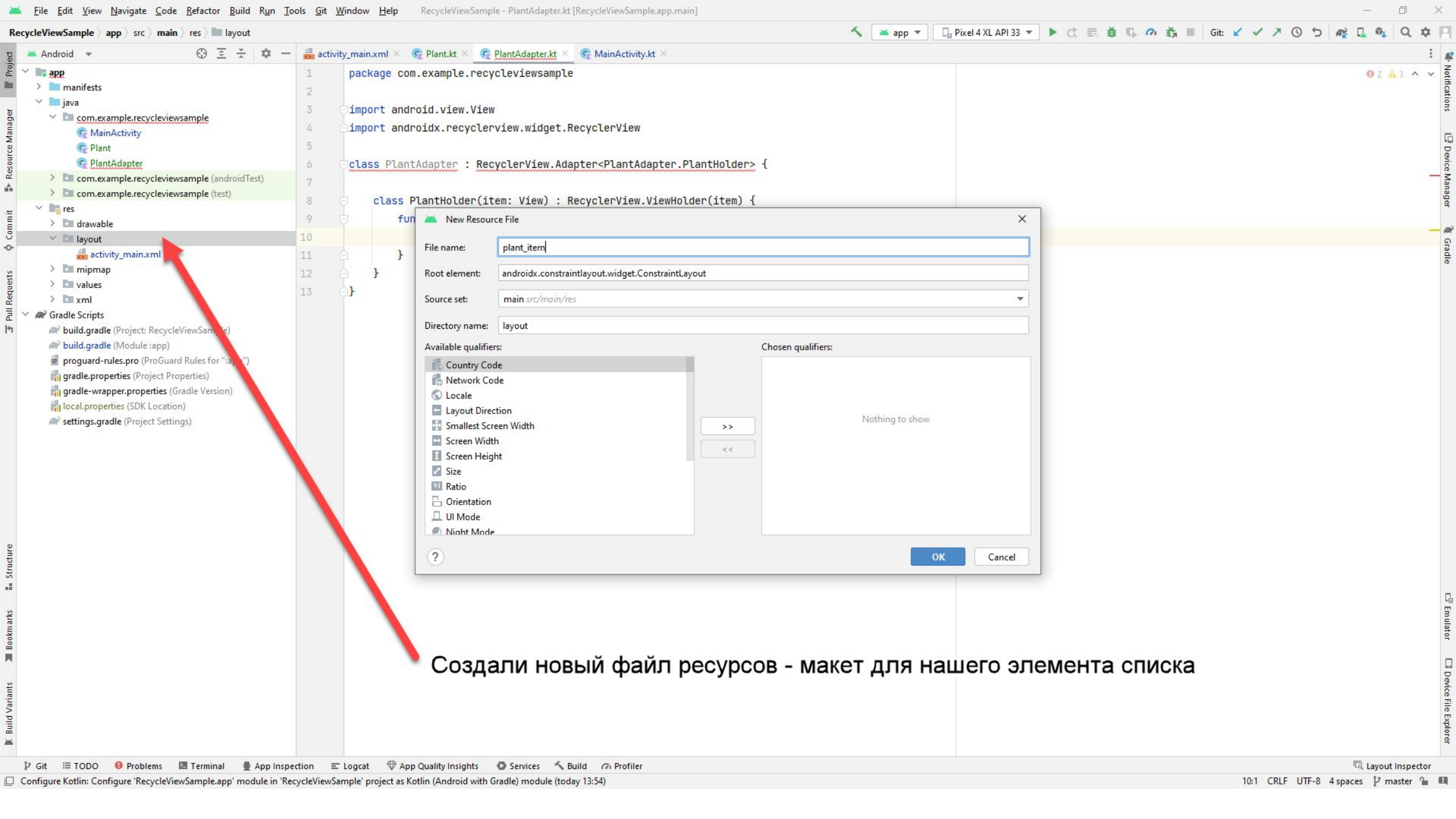
Копируем картинки, которые будем использовать в приложении, в папку drawable

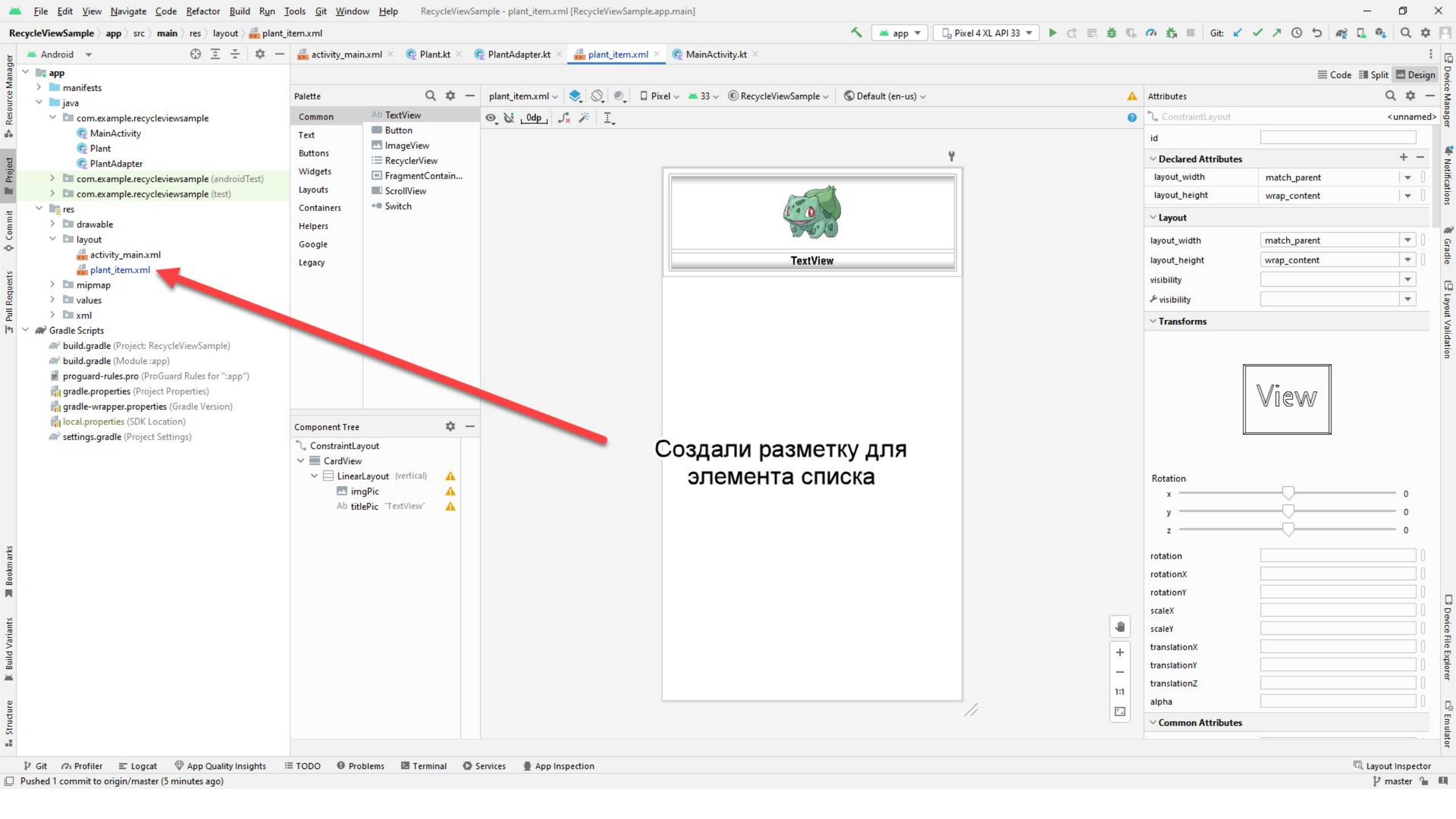
Git TODO Problems Terminal App Inspection Logcat App Quality Insights Services Build Profiler Layout Inspector

Configure Kotlin: Configure 'RecycleViewSample.app' module in 'RecycleViewSample' project as Kotlin (Android with Gradle) module (today 13:54)

4:1 CRLF UTF-8 4 spaces master

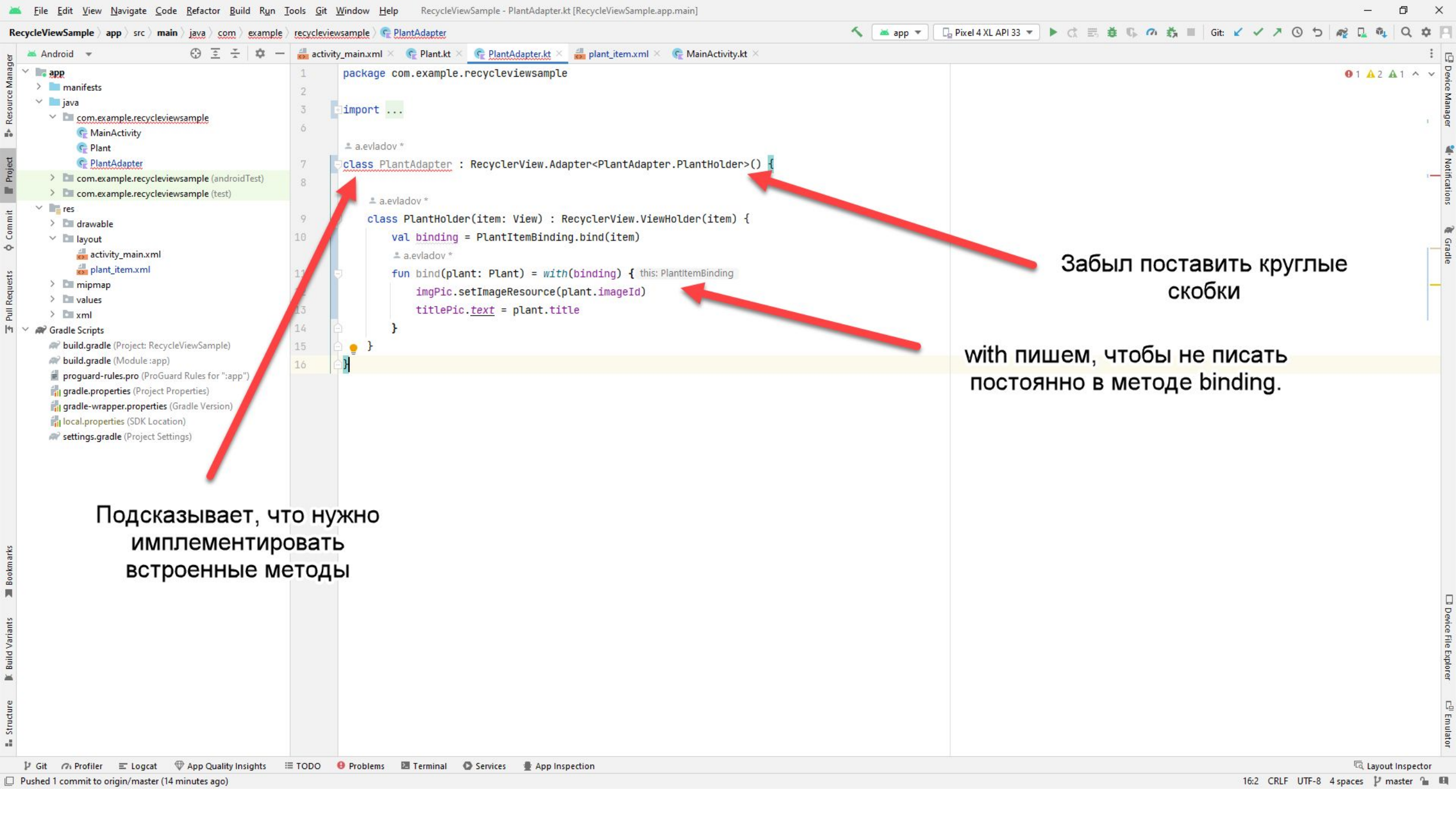

File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]
RecycleViewSample app src main java com example recycleviewsample PlantAdapter PlantHolder bind
Android
activity_main.xml
Plant.kt
PlantAdapter.kt
MainActivity.kt
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
pok_001.png
pok_002.png
pok_003.png
pok_004.png
pok_005.png
pok_006.png
pok_007.png
pok_008.png
pok_009.png
pok_010.png
pok_011.png
pok_012.png
pok_013.png
pok_014.png
pok_015.png
pok_016.png
pok_017.png
pok_018.png
pok_019.png
pok_020.png
pok_021.png
pok_022.png
pok_023.png
pok_024.png
pok_025.png
pok_026.png
pok_027.png
pok_028.png
pok_029.png
pok_030.png
pok_031.png
pok_032.png
1 package com.example.recycleviewsample
3 import android.view.View
4 import androidx.recyclerview.widget.RecyclerView
6 class PlantAdapter : RecyclerView.Adapter<PlantAdapter.PlantHolder> {
8 class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
9 fun bind(plant: Plant) {
11 }
12 }
13 }
Передаем viewHolder - специальный класс, который будет содержать ссылки на все View, которые у нас будут в одном элементе списка. Через него мы заполняем все наши элементы
Передаем разметку, с помощью которой мы и будем заполнять наш элемент
Наш адаптер
Метод, с помощью которого мы заполняем наш шаблон
Git TODO Problems Terminal App Inspection Logcat App Quality Insights Services Build Profiler
Layout Inspector
Configure Kotlin: Configure 'RecycleViewSample.app' module in 'RecycleViewSample' project as Kotlin (Android with Gradle) module (today 13:54)
10:13 CRLF UTF-8 4 spaces master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]
RecycleViewSample app src main res layout
app
Pixel 4 XL API 33
Git:
Android
activity_main.xml
Plant.kt
PlantAdapter.kt
MainActivity.kt
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
1 package com.example.recycleviewsample
2
3 import android.view.View
4 import androidx.recyclerview.widget.RecyclerView
5
6 class PlantAdapter : RecyclerView.Adapter<PlantAdapter.PlantHolder> {
7
8 class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
9 fun
10
11 }
12 }
13 }
New Resource File
File name: plant_item
Root element: androidx.constraintlayout.widget.ConstraintLayout
Source set: main src/main/res
Directory name: layout
Available qualifiers:
Country Code
Network Code
Locale
Layout Direction
Smallest Screen Width
Screen Width
Screen Height
Size
Ratio
Orientation
UI Mode
Night Mode
>>
<<
Chosen qualifiers:
Nothing to show
OK
Cancel
Создали новый файл ресурсов - макет для нашего элемента списка
Git TODO Problems Terminal App Inspection Logcat App Quality Insights Services Build Profiler Layout Inspector
Configure Kotlin: Configure 'RecycleViewSample.app' module in 'RecycleViewSample' project as Kotlin (Android with Gradle) module (today 13:54)
10:1 CRLF UTF-8 4 spaces master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - plant_item.xml [RecycleViewSample.app.main]
RecycleViewSample app src main res layout plant_item.xml
app
Pixel 4 XL API 33
Git:
Android
activity_main.xml
Plant.kt
PlantAdapter.kt
plant_item.xml
MainActivity.kt
Code
Split
Design
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentContain...
ScrollView
Switch
Component Tree
ConstraintLayout
CardView
LinearLayout (vertical)
imgPic
titlePic "TextView"
plant_item.xml
Pixel
33
RecycleViewSample
Default (en-us)
0dp
TextView
Создали разметку для элемента списка
Attributes
ConstraintLayout
<unnamed>
id
Declared Attributes
layout_width
match_parent
layout_height
wrap_content
Layout
layout_width
match_parent
layout_height
wrap_content
visibility
visibility
Transforms
View
Rotation
x
y
z
rotation
rotationX
rotationY
scaleX
scaleY
translationX
translationY
translationZ
alpha
Common Attributes
Git
Profiler
Logcat
App Quality Insights
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Pushed 1 commit to origin/master (5 minutes ago)
master

activity_main.xml | Plant.kt | PlantAdapter.kt | plant_item.xml | MainActivity.kt

```xml
<?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="wrap_content">

    <androidx.cardview.widget.CardView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        app:cardCornerRadius="5dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="3dp"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/imgPic"
                android:layout_width="match_parent"
                android:layout_height="100dp"
                android:layout_gravity="center"
                app:srcCompat="@drawable/pok_001" />

            <TextView
                android:id="@+id/titlePic"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="5dp"
                android:fontFamily="sans-serif-condensed"
                android:gravity="center"
                android:text="TextView"
                android:textColor="@color/black"
                android:textSize="16sp"
                android:textStyle="bold" />

        </LinearLayout>
    </androidx.cardview.widget.CardView>
</androidx.constraintlayout.widget.ConstraintLayout>
```


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]
RecycleViewSample app src main java com example recycleviewsample PlantAdapter
Android
app
Pixel 4 XL API 33
activity_main.xml
Plant.kt
PlantAdapter.kt
plant_item.xml
MainActivity.kt
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
1 package com.example.recycleviewsample
2
3 import ...
6
a.evladov *
7 class PlantAdapter : RecyclerView.Adapter<PlantAdapter.PlantHolder>() {
8
a.evladov *
9 class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
10 val binding = PlantItemBinding.bind(item)
a.evladov *
11 fun bind(plant: Plant) = with(binding) { this: PlantItemBinding
imgPic.setImageResource(plant.imageId)
13 titlePic.text = plant.title
14 }
15 }
16 }
Подсказывает, что нужно имплементировать встроенные методы
Забыл поставить круглые скобки
with пишем, чтобы не писать постоянно в методе binding.
Git Profiler Logcat App Quality Insights TODO Problems Terminal Services App Inspection
Layout Inspector
Pushed 1 commit to origin/master (14 minutes ago)
16:2 CRLF UTF-8 4 spaces master

LayoutInflater - это класс, который умеет из содержимого layout-файла создать View-элемент. Метод, который это делает, называется inflate. Есть несколько реализаций этого метода с различными параметрами. Но все они используют друг друга и результат их выполнения один - View.

Рассмотрим эту реализацию - public View inflate (int resource, ViewGroup root, boolean attachToRoot)

Как видим, на вход метод принимает три параметра:

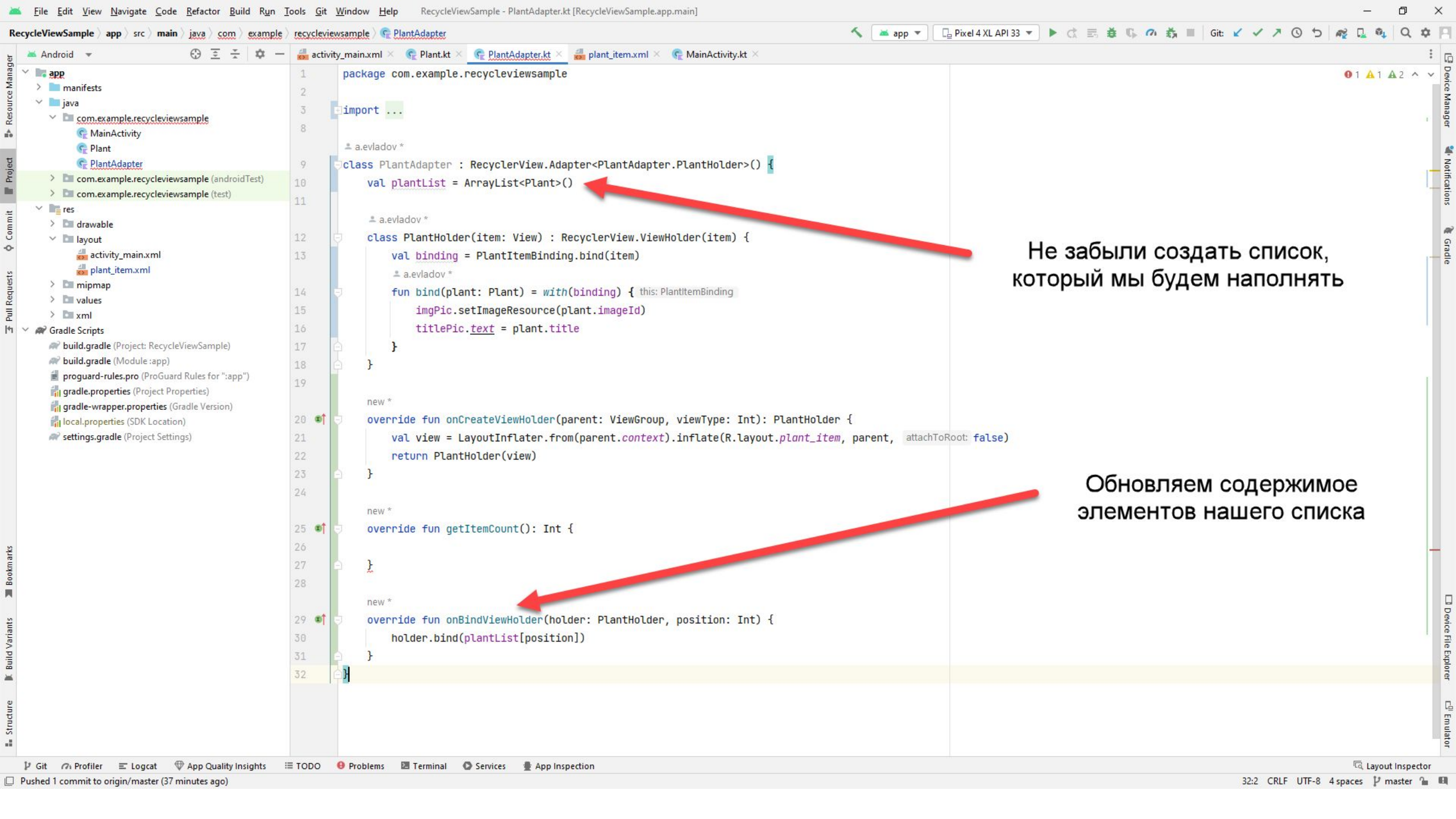
resource - ID layout-файла, который будет использован для создания View. Например - R.layout.main

root - родительский ViewGroup-элемент для создаваемого View. LayoutParams от этого ViewGroup присваиваются создаваемому View.

attachToRoot - присоединять ли создаваемый View к root. Если true, то root становится родителем создаваемого View. Т.е. это равносильно команде root.addView(View). Если false - то создаваемый View просто получает LayoutParams от root, но его дочерним элементом не становится.

Берем контекст, который есть во ViewGroup

Ресурс, который мы надуваем


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]
RecycleViewSample 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 recycleviewsample 〉 PlantAdapter
Android
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
xml
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
activity_main.xml
Plant.kt
PlantAdapter.kt
plant_item.xml
MainActivity.kt
package com.example.recycleviewsample
import ...
a.evladov *
class PlantAdapter : RecyclerView.Adapter<PlantAdapter.PlantHolder>() {
val plantList = ArrayList<Plant>()
a.evladov *
class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
val binding = PlantItemBinding.bind(item)
a.evladov *
fun bind(plant: Plant) = with(binding) { this: PlantItemBinding
imgPic.setImageResource(plant.imageId)
titlePic.text = plant.title
}
}
new *
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): PlantHolder {
val view = LayoutInflater.from(parent.context).inflate(R.layout.plant_item, parent, attachToRoot: false)
return PlantHolder(view)
}
new *
override fun getItemCount(): Int {
}
new *
override fun onBindViewHolder(holder: PlantHolder, position: Int) {
holder.bind(plantList[position])
}
}
Не забыли создать список,
который мы будем наполнять
Обновляем содержимое
элементов нашего списка
Git
Profiler
Logcat
App Quality Insights
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Pushed 1 commit to origin/master (37 minutes ago)
32:2 CRLF UTF-8 4 spaces master

File Edit View Navigate Code Refactor Build Run Tools Git Window Help RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]

RecycleViewSample › app › src › main › java › com › example › recycleviewsample › PlantAdapter

app | Pixel 4 XL API 33 | Git:

Android

- app
  - manifests
  - java
    - com.example.recycleviewsample
      - MainActivity
      - Plant
      - PlantAdapter
    - com.example.recycleviewsample (androidTest)
    - com.example.recycleviewsample (test)
  - res
    - drawable
    - layout
      - activity_main.xml
      - plant_item.xml
    - mipmap
    - values
    - xml
- Gradle Scripts
  - build.gradle (Project: RecycleViewSample)
  - build.gradle (Module :app)
  - proguard-rules.pro (ProGuard Rules for ":app")
  - gradle.properties (Project Properties)
  - gradle-wrapper.properties (Gradle Version)
  - local.properties (SDK Location)
  - settings.gradle (Project Settings)

activity_main.xml | Plant.kt | PlantAdapter.kt | plant_item.xml | MainActivity.kt

```
1  package com.example.recycleviewsample
2
3  import ...
8
   a.evladov *
9  class PlantAdapter : RecyclerView.Adapter<PlantAdapter.PlantHolder>() {
10     val plantList = ArrayList<Plant>()
11
       a.evladov *
12     class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
13         val binding = PlantItemBinding.bind(item)
           a.evladov *
14         fun bind(plant: Plant) = with(binding) { this: PlantItemBinding
15             imgPic.setImageResource(plant.imageId)
16             titlePic.text = plant.title
17         }
18     }
19
       new *
20     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): PlantHolder {
21         val view = LayoutInflater.from(parent.context).inflate(R.layout.plant_item, parent, attachToRoot: false)
22         return PlantHolder(view)
23     }
24
       new *
25     override fun getItemCount(): Int {
26         return plantList.size
27     }
28
       new *
29     override fun onBindViewHolder(holder: PlantHolder, position: Int) {
30         holder.bind(plantList[position])
31     }
32 }
```

Передали размер нашего ArrayList.

Git | Profiler | Logcat | App Quality Insights | TODO | Problems | Terminal | Services | App Inspection | Layout Inspector

Pushed 1 commit to origin/master (41 minutes ago)

32:2 CRLF UTF-8 4 spaces master

File Edit View Navigate Code Refactor Build Run Tools Git Window Help RecycleViewSample - PlantAdapter.kt [RecycleViewSample.app.main]

RecycleViewSample › app › src › main › java › com › example › recycleviewsample › PlantAdapter

app | Pixel 4 XL API 33 | Git:

Android

- app
  - manifests
  - java
    - com.example.recycleviewsample
      - MainActivity
      - Plant
      - PlantAdapter
    - com.example.recycleviewsample (androidTest)
    - com.example.recycleviewsample (test)
  - res
    - drawable
    - layout
      - activity_main.xml
      - plant_item.xml
    - mipmap
    - values
    - xml
- Gradle Scripts
  - build.gradle (Project: RecycleViewSample)
  - build.gradle (Module :app)
  - proguard-rules.pro (ProGuard Rules for ":app")
  - gradle.properties (Project Properties)
  - gradle-wrapper.properties (Gradle Version)
  - local.properties (SDK Location)
  - settings.gradle (Project Settings)

activity_main.xml | Plant.kt | PlantAdapter.kt | plant_item.xml | MainActivity.kt

```
10  val plantList = ArrayList<Plant>()
11
    a.evladov
12  class PlantHolder(item: View) : RecyclerView.ViewHolder(item) {
13      val binding = PlantItemBinding.bind(item)
        a.evladov
14      fun bind(plant: Plant) = with(binding) { this: PlantItemBinding
15          imgPic.setImageResource(plant.imageId)
16          titlePic.text = plant.title
17      }
18  }
19
    a.evladov
20  override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): PlantHolder {
21      val view = LayoutInflater.from(parent.context).inflate(R.layout.plant_item, parent, attachToRoot: false)
22      return PlantHolder(view)
23  }
24
    a.evladov
25  override fun getItemCount(): Int {
26      return plantList.size
27  }
28
    a.evladov
29  override fun onBindViewHolder(holder: PlantHolder, position: Int) {
30      holder.bind(plantList[position])
31  }
32
    new *
33  fun addPlant(plant: Plant) {
34      plantList.add(plant)
35      notifyDataSetChanged()
36  }
37  }
```

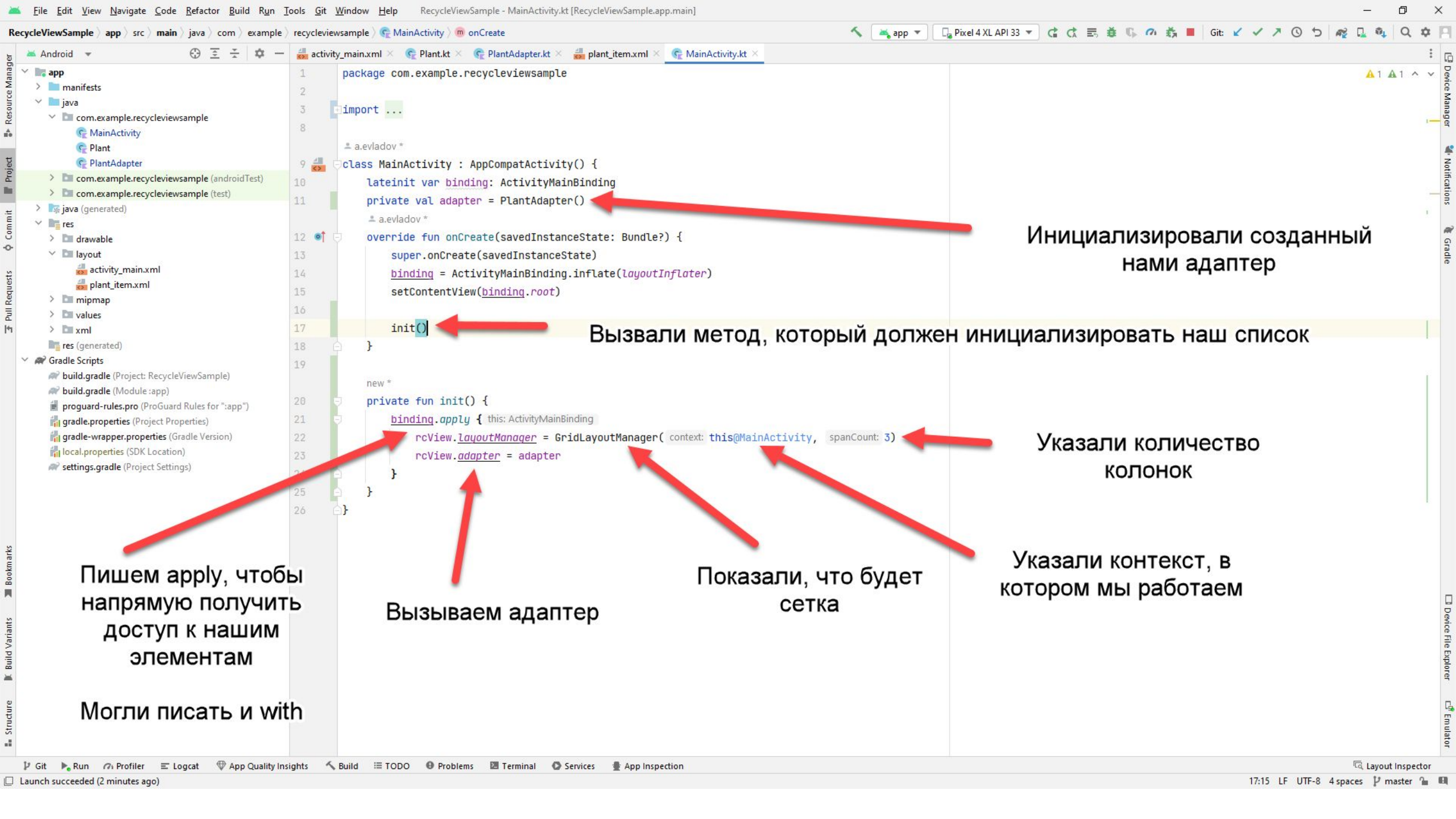
Добавили элемент в список

Перерисовали список после добавления элемента

Git | Profiler | Logcat | App Quality Insights | TODO | Problems | Terminal | Services | App Inspection | Layout Inspector

Pushed 1 commit to origin/master (a minute ago)

37:2 CRLF UTF-8 4 spaces master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - MainActivity.kt [RecycleViewSample.app.main]
RecycleViewSample app src main java com example recycleviewsample MainActivity onCreate
Android
app
Pixel 4 XL API 33
activity_main.xml Plant.kt PlantAdapter.kt plant_item.xml MainActivity.kt
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
java (generated)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
xml
res (generated)
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
1 package com.example.recycleviewsample
3 import ...
a.evladov *
9 class MainActivity : AppCompatActivity() {
10 lateinit var binding: ActivityMainBinding
11 private val adapter = PlantAdapter()
a.evladov *
12 override fun onCreate(savedInstanceState: Bundle?) {
13 super.onCreate(savedInstanceState)
14 binding = ActivityMainBinding.inflate(layoutInflater)
15 setContentView(binding.root)
17 init()
18 }
new *
20 private fun init() {
21 binding.apply { this: ActivityMainBinding
22 rcView.layoutManager = GridLayoutManager(context: this@MainActivity, spanCount: 3)
23 rcView.adapter = adapter
25 }
26 }
Инициализировали созданный нами адаптер
Вызвали метод, который должен инициализировать наш список
Указали количество колонок
Пишем apply, чтобы напрямую получить доступ к нашим элементам
Могли писать и with
Вызываем адаптер
Показали, что будет сетка
Указали контекст, в котором мы работаем
Git Run Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection
Layout Inspector
Launch succeeded (2 minutes ago)
17:15 LF UTF-8 4 spaces master

File Edit View Navigate Code Refactor Build Run Tools Git Window Help RecycleViewSample - colors.xml [RecycleViewSample.app.main]

RecycleViewSample › app › src › main › res › values › colors.xml

app | Pixel 4 XL API 33 | Git:

Android

- app
  - manifests
  - java
    - com.example.recycleviewsample
      - MainActivity
      - Plant
      - PlantAdapter
    - com.example.recycleviewsample (androidTest)
    - com.example.recycleviewsample (test)
  - java (generated)
  - res
    - drawable
    - layout
      - activity_main.xml
      - plant_item.xml
    - mipmap
    - values
      - colors.xml
      - strings.xml
      - themes (2)
        - themes.xml
        - themes.xml (night)
    - xml
  - res (generated)
- Gradle Scripts
  - build.gradle (Project: RecycleViewSample)
  - build.gradle (Module :app)
  - proguard-rules.pro (ProGuard Rules for ":app")
  - gradle.properties (Project Properties)
  - gradle-wrapper.properties (Gradle Version)
  - local.properties (SDK Location)
  - settings.gradle (Project Settings)

activity_main.xml | Plant.kt | PlantAdapter.kt | plant_item.xml | MainActivity.kt | themes.xml | colors.xml

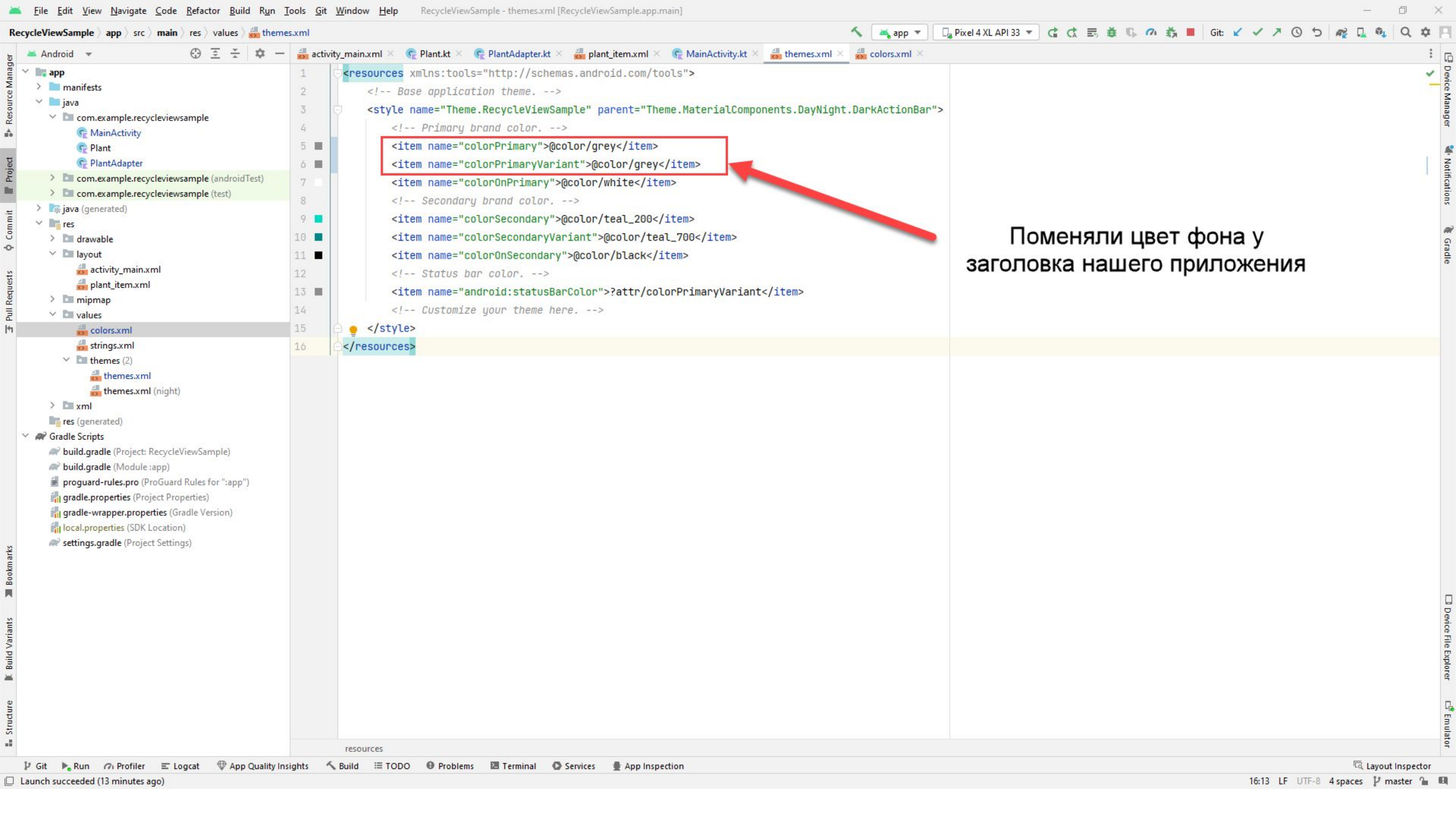
```
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <color name="purple_200">#FFBB86FC</color>
4      <color name="purple_500">#FF6200EE</color>
5      <color name="purple_700">#FF3700B3</color>
6      <color name="teal_200">#FF03DAC5</color>
7      <color name="teal_700">#FF018786</color>
8      <color name="black">#FF000000</color>
9      <color name="white">#FFFFFFFF</color>
10     <color name="grey">#8E8E8E</color>
11 </resources>
```

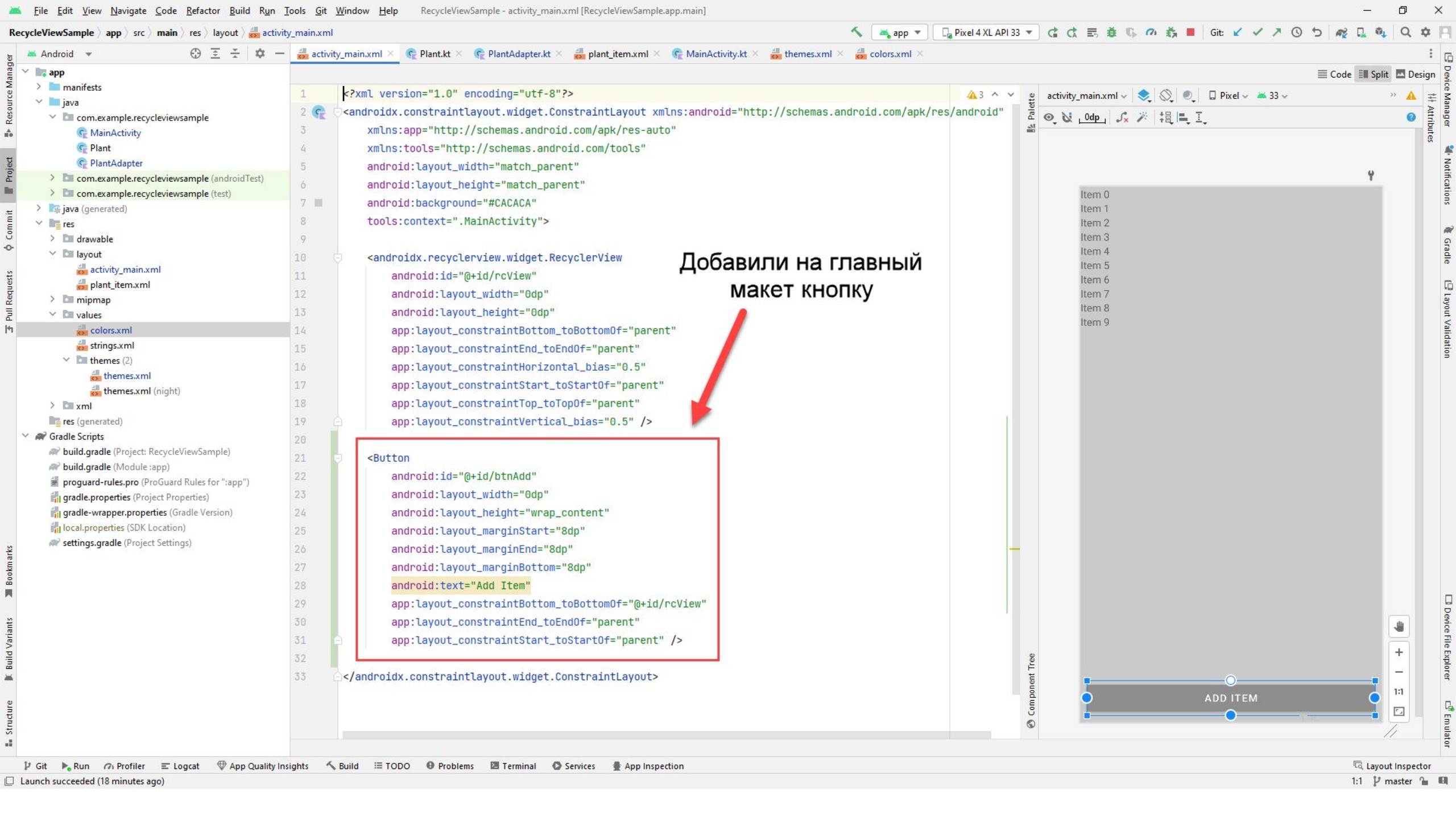
Создали в ресурсах новый цвет

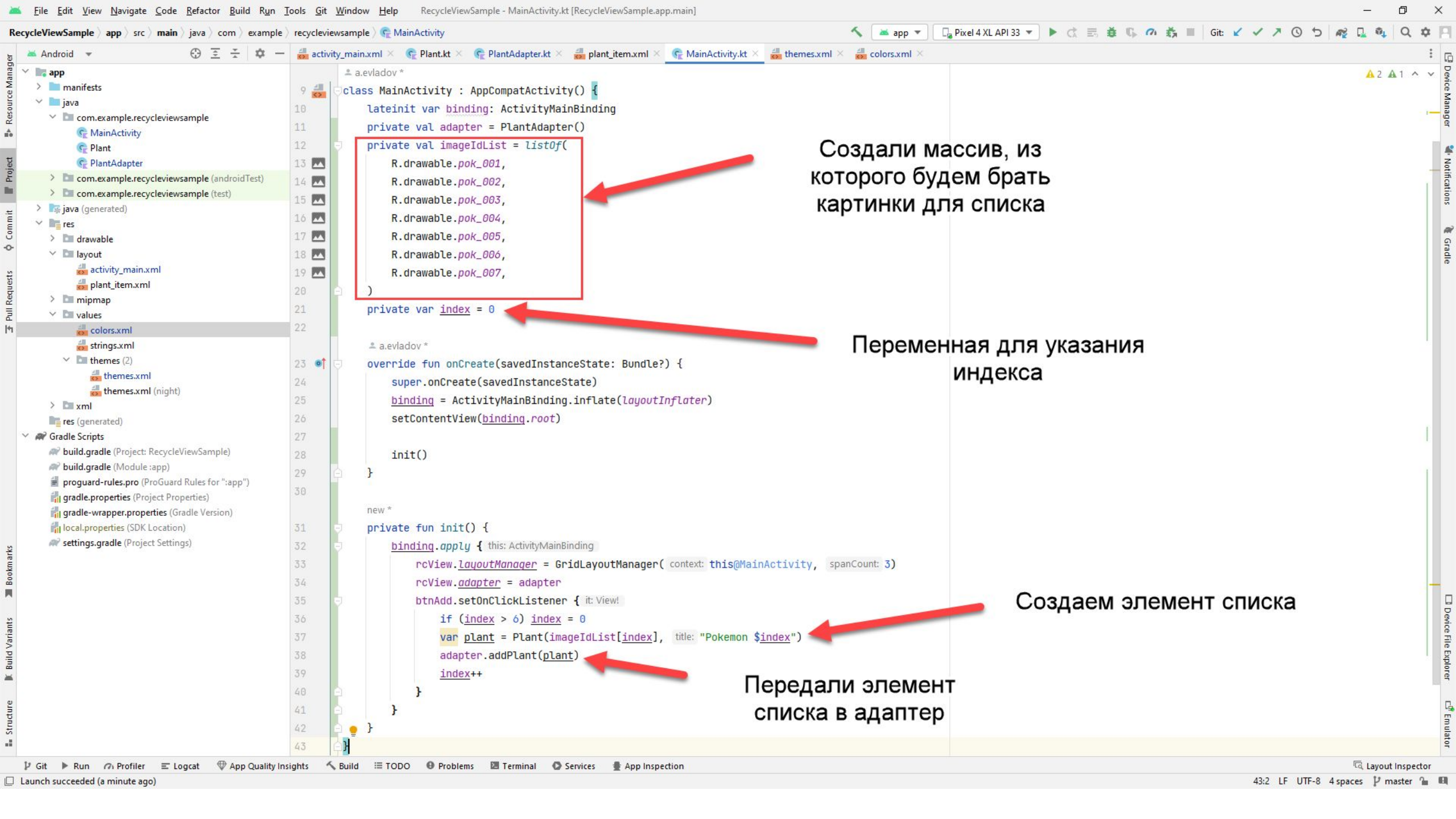
resources

Git | Run | Profiler | Logcat | App Quality Insights | Build | TODO | Problems | Terminal | Services | App Inspection | Layout Inspector

Launch succeeded (12 minutes ago) 11:13 LF UTF-8 4 spaces master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - themes.xml [RecycleViewSample.app.main]
RecycleViewSample app src main res values themes.xml
app
Pixel 4 XL API 33
Git:
Android
activity_main.xml
Plant.kt
PlantAdapter.kt
plant_item.xml
MainActivity.kt
themes.xml
colors.xml
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
java (generated)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
colors.xml
strings.xml
themes (2)
themes.xml
themes.xml (night)
xml
res (generated)
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
1 <resources xmlns:tools="http://schemas.android.com/tools">
2 <!-- Base application theme. -->
3 <style name="Theme.RecycleViewSample" parent="Theme.MaterialComponents.DayNight.DarkActionBar">
4 <!-- Primary brand color. -->
5 <item name="colorPrimary">@color/grey</item>
6 <item name="colorPrimaryVariant">@color/grey</item>
7 <item name="colorOnPrimary">@color/white</item>
8 <!-- Secondary brand color. -->
9 <item name="colorSecondary">@color/teal_200</item>
10 <item name="colorSecondaryVariant">@color/teal_700</item>
11 <item name="colorOnSecondary">@color/black</item>
12 <!-- Status bar color. -->
13 <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
14 <!-- Customize your theme here. -->
15 </style>
16 </resources>
Поменяли цвет фона у заголовка нашего приложения
resources
Git Run Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection
Layout Inspector
Launch succeeded (13 minutes ago)
16:13 LF UTF-8 4 spaces master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RecycleViewSample - activity_main.xml [RecycleViewSample.app.main]
RecycleViewSample app src main res layout activity_main.xml
app
Pixel 4 XL API 33
Android
activity_main.xml
Plant.kt
PlantAdapter.kt
plant_item.xml
MainActivity.kt
themes.xml
colors.xml
Code Split Design
app
manifests
java
com.example.recycleviewsample
MainActivity
Plant
PlantAdapter
com.example.recycleviewsample (androidTest)
com.example.recycleviewsample (test)
java (generated)
res
drawable
layout
activity_main.xml
plant_item.xml
mipmap
values
colors.xml
strings.xml
themes (2)
themes.xml
themes.xml (night)
xml
res (generated)
Gradle Scripts
build.gradle (Project: RecycleViewSample)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)
<?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:background="#CACACA"
tools:context=".MainActivity">
<androidx.recyclerview.widget.RecyclerView
android:id="@+id/rcView"
android:layout_width="0dp"
android:layout_height="0dp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.5"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.5" />
Добавили на главный макет кнопку
<Button
android:id="@+id/btnAdd"
android:layout_width="0dp"
android:layout_height="wrap_content"
android:layout_marginStart="8dp"
android:layout_marginEnd="8dp"
android:layout_marginBottom="8dp"
android:text="Add Item"
app:layout_constraintBottom_toBottomOf="@+id/rcView"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
Item 0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8
Item 9
ADD ITEM
Git Run Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection
Layout Inspector
Launch succeeded (18 minutes ago)
master


Создали массив, из которого будем брать картинки для списка
Переменная для указания индекса
Создаем элемент списка
Передали элемент списка в адаптер


Emulator:
Pixel 4 XL API 33
2:13
Tue, Apr 11
Gmail
Photos
YouTube
1:1

Основные недостатки ListView

Сложно добавить анимации и декорации элементам списка,

Компонент ListView отвечает не только за переиспользование View, но и за порядок и структуру списка. Это значит, что реализовать горизонтальный список, используя ListView и не затрагивая большой части кодовой базы — проблематично,

Базовая реализация адаптера подразумевает поиск элементов View (findViewById) для отображения данных при каждом новом биндинге данных. И хотя шаблон ViewHolder широко известен, но применялся далеко не всегда.

RecyclerView — замена ListView

Основные отличия RecyclerView от ListView

RecyclerView отвечает только за переиспользование View, способ отображения данных задается с помощью отдельного менеджера — LayoutManager , существует ряд базовых реализаций

LinearLayoutManager — отображение горизонатльных и вертикальных последовательных и инвертированных списков,

GridLayoutManager — отображение табличных списков,

StaggeredGridLayoutManager — отображение табличных списков с контейнерами динамического размера

За анимации отдельных элементов при изменении, добавлении, удалении отвечает отдельный компонент ItemAnimator

Принудительно используется шаблон ViewHolder. Это позволяет оптимизировать производительность за счет сохранения ссылок на View в контейнере.

Компонентный подход в реализации RecyclerView позволяет гибко настраивать по отдельности каждый аспект списка, дописывая и переопределяя только необходимые части реализации.

ЗАДАНИЕ

№ 0001

Bulbasaur

Grass Poison

№ 0002

Ivysaur

Grass Poison

№ 0003

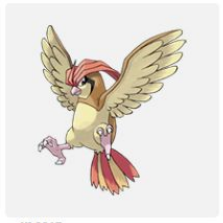
Venusaur

Grass Poison

№ 0004

Charmander

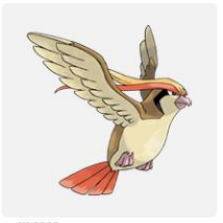
Fire



№ 0017

Pidgeotto

Normal Flying



№ 0018

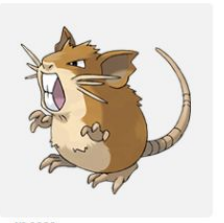
Pidgeot

Normal Flying

№ 0019

Rattata

Normal



№ 0020

Raticate

Normal

№ 0033

Nidorino

Poison

№ 0034

Nidoking

Poison Ground

№ 0005

Charmeleon

Fire

№ 0006

Charizard

Fire Flying

№ 0007

Squirtle

Water

№ 0008

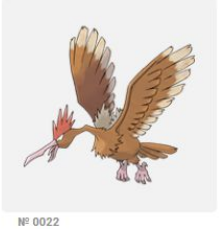
Wartortle

Water

№ 0021

Spearow

Normal Flying



№ 0022

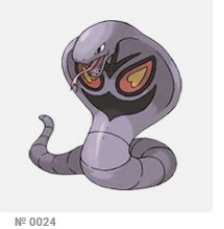
Fearow

Normal Flying

№ 0023

Ekans

Poison



№ 0024

Arbok

Poison

№ 0035

Clefairy

Fairy

№ 0036

Clefable

Fairy

№ 0009

Blastoise

Water

№ 0010

Caterpie

Bug

№ 0011

Metapod

Bug

№ 0012

Butterfree

Bug Flying

№ 0025

Pikachu

Electric

№ 0026

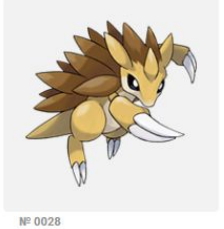
Raichu

Electric

№ 0027

Sandshrew

Ground



№ 0028

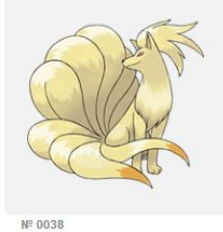
Sandslash

Ground

№ 0037

Vulpix

Fire



№ 0038

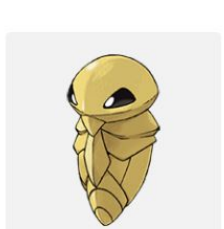
Ninetales

Fire

№ 0013

Weedle

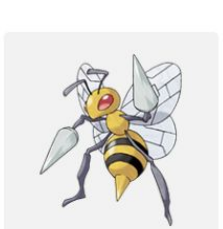
Bug Poison



№ 0014

Kakuna

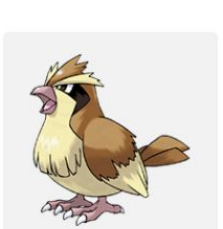
Bug Poison



№ 0015

Beedrill

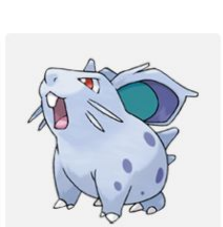
Bug Poison



№ 0016

Pidgey

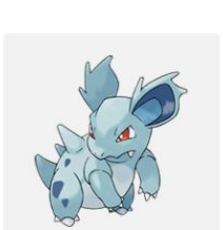
Normal Flying



№ 0029

Nidoran♀

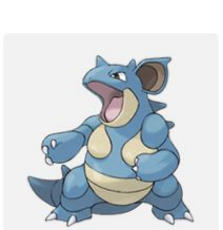
Poison



№ 0030

Nidorina

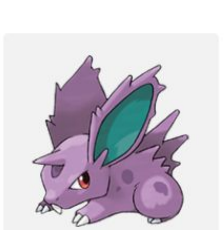
Poison



№ 0031

Nidoqueen

Poison Ground



№ 0032

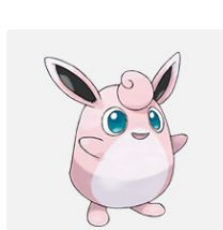
Nidoran♂

Poison

№ 0039

Jigglypuff

Normal Fairy



№ 0040

Wigglytuff

Normal Fairy

Изменить приложение, выполненное в качестве примера так, чтобы вместо подписи Pokemon ### было написано настоящее имя героя.

В качестве источника картинок можно использовать ресурс https://www.pokemon.com/ru/pokedex/

АДАПТЕРЫ

В Android часто используются адаптеры.

Если говорить в общих чертах, то адаптеры упрощают связывание данных с элементом управления. Адаптеры используются при работе с виджетами, которые дополняют android.widget.AdapterView: ListView, ExpandableListView, GridView, Spinner, Gallery, а также в активности ListActivity и др. Сам AdapterView дополняет android.widget.ViewGroup.

Итак, у нас есть набор объектов и есть компонент View. Назначение адаптера заключается в том, чтобы предоставлять дочерние виды для контейнера. Адаптер берет данные и метаданные определенного контейнера и строит каждый дочерний вид. Например, мы формируем пункты списка (мас строк) и передаём его списку ListView.

Готовые адаптеры

Все адаптеры, содержащиеся в Android, дополняют базовый адаптер BaseAdapter. Вот список некоторых готовых адаптеров:

ArrayAdapter<T> - предназначен для работы с ListView. Данные представлены в виде массива, которые размещаются в отдельных элементах TextView

ListAdapter - адаптер между ListView и данными. Строго говоря, это класс-интерфейс, который можно использовать и в ArrayAdapter и в SimpleAdapter и т.д.

SpinnerAdapter - адаптер для связки данных с элементом Spinner. Это тоже интерфейс, как

ListAdapter и работает по схожему принципу

SimpleAdapter - адаптер, позволяющий заполнить данными список более сложной структуры, например, два текста в одной строке списка.

SimpleCursorAdapter - дополняет ResourceCursorAdapter и создаёт компоненты TextView/ImageView из столбцов, содержащихся в курсоре. Компоненты определяются в ресурсах

CursorAdapter - предназначен для работы с ListView, предоставляет данные для списка через курсор, который должен иметь колонку с именем "_id"

ResourceCursorAdapter - этот адаптер дополняет CursorAdapter и может создавать виды из ресурсов

HeaderViewListAdapter - расширенный вариант ListAdapter, когда ListView имеет заголовки.

WrapperListAdapter - еще один адаптер для списков.

Адаптеры в Android представляют собой мост между представлением адаптера (например, ListView) и базовыми данными для этого представления. Это важнейшая концепция в архитектуре Android, а также требуется для сертификации Android. Представьте, каким был бы мир без адаптеров Android!

Почему Android Adapter?
Без адаптера Android для реализации функциональности ListView вам потребуется:

1. Создайте TextView в группе ScrollView.

2. Затем вам нужно будет реализовать концепцию разбивки на страницы для содержимого TextView.

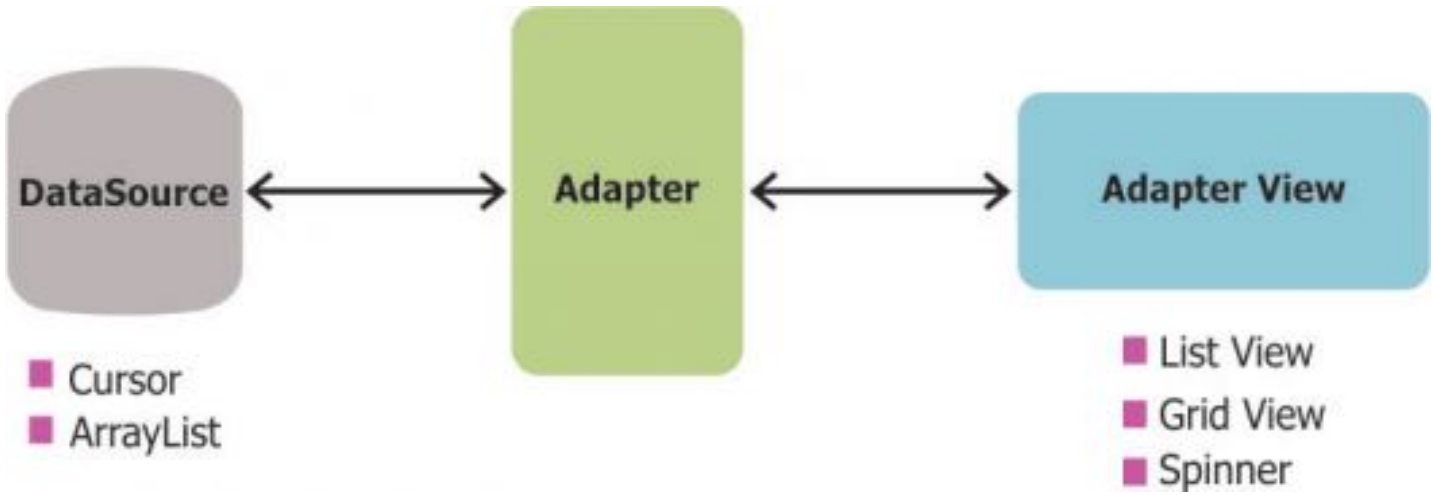
3. Вам также придется написать дополнительный код для идентификации события щелчка по определенной строке в TextView.

Довольно громоздкая работа! Не так ли?

Возможно, вы спрашиваете, зачем нам нужна разбивка на страницы?

Представьте, что вы создаете приложение, которое должно отображать все ваши электронные письма, и пользователь сможет прокручивать их. Я получаю около 100 электронных писем ежедневно, и даже если мы рассмотрим 10 электронных писем в день, данные электронной почты будут чрезвычайно огромными. Если вы не реализуете разбивку на страницы, у вас возникнут серьезные проблемы с производительностью при получении всех этих данных и отображении их на экране мобильного устройства. Слава богу, у нас есть адаптеры в Android и виды адаптеров!

Ниже приведена концептуальная схема, которая показывает высокий уровень работы адаптера Android:



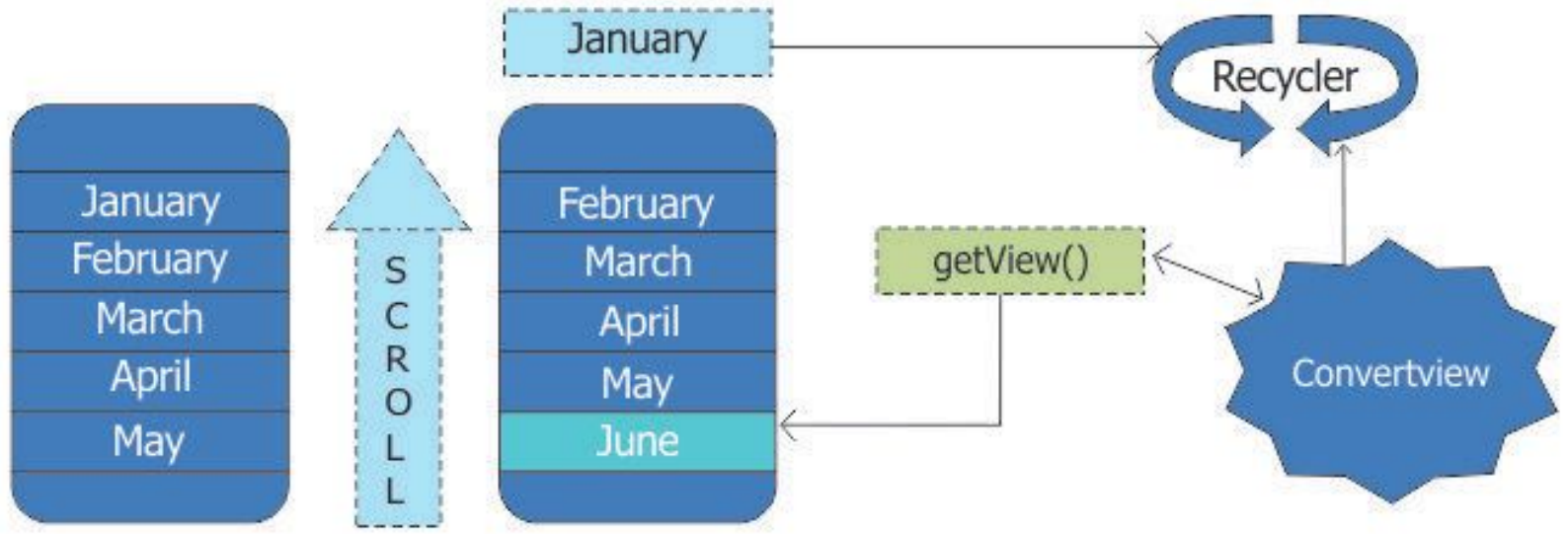
Теперь давайте разберемся во внутренней работе адаптера Android и в том, как он выполняет функцию перекачки данных в представление адаптера.

Адаптеры вызывают метод **getView()**, который возвращает представление для каждого элемента в представлении адаптера. Формат макета и соответствующие данные для элемента в представлении адаптера устанавливаются в методе **getView()**.

Теперь это будет кошмар производительности, если **getView ()** будет возвращать новое представление каждый раз, когда оно вызывается. Создание нового представления в Android обходится очень дорого, поскольку вам нужно будет перебирать иерархию представлений (используя метод find **ViewbyID ()**), а затем раздувать представление, чтобы окончательно отобразить его на экране.

Это также оказывает большое давление на сборщик мусора. Это потому, что когда пользователь прокручивает список, если создается новое представление; на старое представление (поскольку оно не перерабатывается) не ссылаются, и оно становится кандидатом на получение сборщиком мусора. Итак, что делает Android, так это то, что он перерабатывает представления и повторно использует представление, которое выходит из фокуса.

Ниже приведено визуальное представление этого процесса утилизации:



На приведенном выше рисунке давайте предположим, что мы отображаем месяцы в году в ListView.

Начнем с того, что на экране отображаются месяцы с января по май. При прокрутке просмотра месяц январь исчезает из области отображения экрана мобильного устройства. Как только представление за январь исчезает с экрана, представление адаптера (в данном случае ListView) отправляет представление во что-то, называемое утилизатором.

Итак, при прокрутке вверх вызывается метод getView (), чтобы получить следующий просмотр (который в июне). Этот метод getView() имеет параметр с именем convertview, который указывает на неиспользуемый вид в утилизаторе. Через convertview адаптер пытается получить неиспользуемый вид и повторно использовать его для отображения нового вида (в данном случае June).

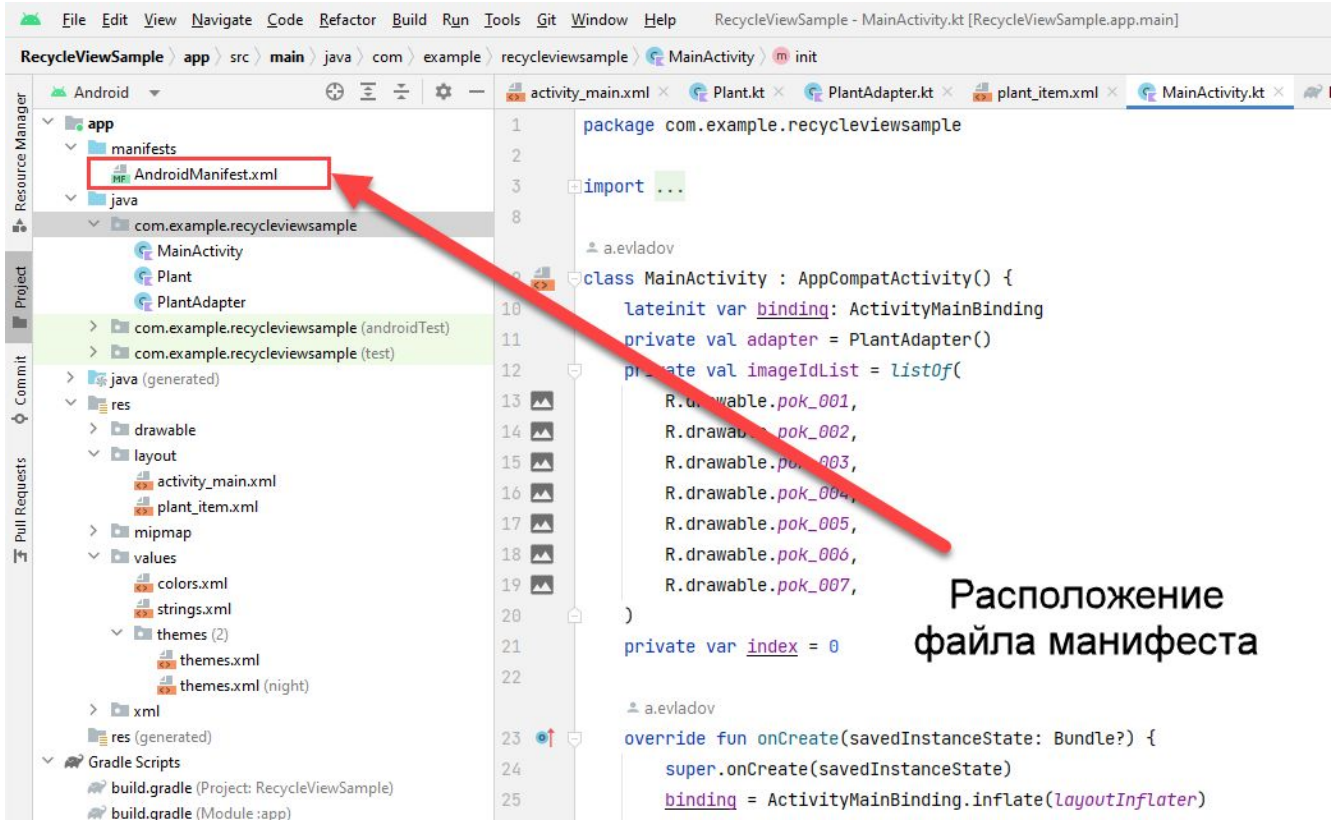
https://github.com/mikepenz/FastAdapter

Пуленепробиваемая, быстрая и простая в использовании библиотека адаптеров, которая сводит время разработки к минимуму… — Майк Пенз

https://habr.com/ru/articles/599851/

МАНИФЕСТ ПРИЛОЖЕНИЯ

Манифест - это набор правил, по которым работает приложение. Файл манифеста находится в корневой папке - AndroidManifest.xml - и содержит важную информацию, без которой система не сможет запустить приложение.



Основное содержимое манифеста:

- Имя пакета для приложения. Является идентификатором приложения.
- Описание возможностей компонентов приложения: Activity, Service, Broadcast Receiver.
- Процессы, в которых будут запускаться компоненты приложения.
- Разрешения, которые необходимо запросить у пользователя, чтобы у приложения был доступ к защищенным частям API, а также чтобы другие приложения могли обращаться к вашему приложению.
- Содержит список классов Instrumentation, которые при выполнении приложения предоставляют сведения о профиле и прочую информацию. Эти объявления присутствуют в файле манифеста только во время разработки и отладки приложения и удаляются перед его публикацией.
- Объявляет минимальный уровень API Android, необходимый для работы приложения.
- Перечисляет связанные библиотеки.

Основные правила манифеста:

- Обязательно надо указать в манифесте два тега: <manifest> и <application>. Оба можно указать только один раз.

- Большинство тегов можно указывать несколько раз или не указывать совсем. Это зависит от того, что будет делать приложение.

- Тег <activity-alias> должен следовать за тегом своей <activity>. Но многие теги одного уровня могут следовать в любом порядке.

- Все значения задаются только в виде атрибутов. Некоторые атрибуты надо прописывать обязательно, некоторые – нет. Если атрибут не прописан – используется значение по умолчанию.

- Большинство атрибутов должны начинаться с префикса android. Только некоторые атрибуты тега manifest могут не иметь этого префикса.

- Многие элементы соответствуют объектам Java. Например, если вы указываете имя для activity с помощью атрибута name, то в состав имени должно входить полное обозначение пакета.

- Если элементу нужно указать несколько значений, то он всегда приводится повторно, вместо перечисления нескольких значений в одном элементе.

https://developer.android.com/guide/topics/manifest/manifest-intro

https://bimlibik.github.io/posts/manifest-file/

```xml
<?xml version="1.0" encoding="utf-8"?>

<manifest>

    <uses-permission />
    <permission />
    <permission-tree />
    <permission-group />
    <instrumentation />
    <uses-sdk />
    <uses-configuration />
    <uses-feature />
    <supports-screens />
    <compatible-screens />
    <supports-gl-texture />

    <application>

        <activity>
            <intent-filter>
                <action />
                <category />
                <data />
            </intent-filter>
            <meta-data />
        </activity>

        <activity-alias>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </activity-alias>

        <service>
            <intent-filter> . . . </intent-filter>
            <meta-data/>
        </service>

        <receiver>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </receiver>

        <provider>
            <grant-uri-permission />
            <meta-data />
            <path-permission />
        </provider>

        <uses-library />

    </application>

</manifest>
```