Adapters

android

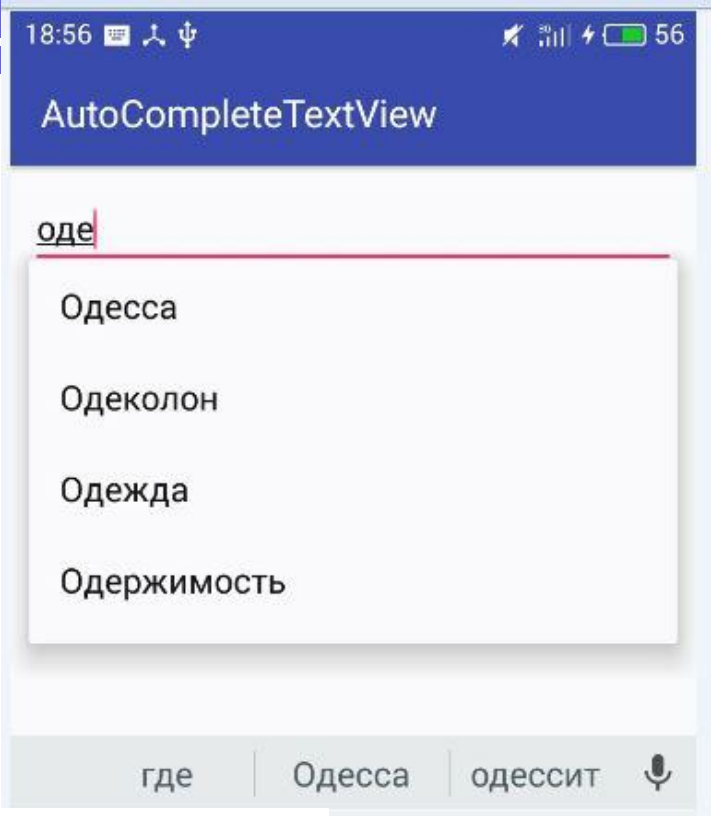
AutoCompleteTextView

Компонент **AutoCompleteTextView** - это текстовое поле с автозаполнением и возможностью редактирования вводимого текста. Использование компонента удобно в том случае, когда требуется ускорить процесс ввода текста. У **AutoCompleteTextView** есть свойство **completionThreshold** для указания минимального числа символов, которое должен ввести пользователь, чтобы включилась функция автозаполнения.

AutoCompleteTextView

```
<AutoCompleteTextView
    android:id="@+id/autoCompleteTextView1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_margin="10dp" />
```

```
<string-array name="array1">
    <item>Одесса</item>
    <item>Одеколон</item>
    <item>Одежда</item>
    <item>Одержимость</item>
    <item>Одеяло</item>
</string-array>
```

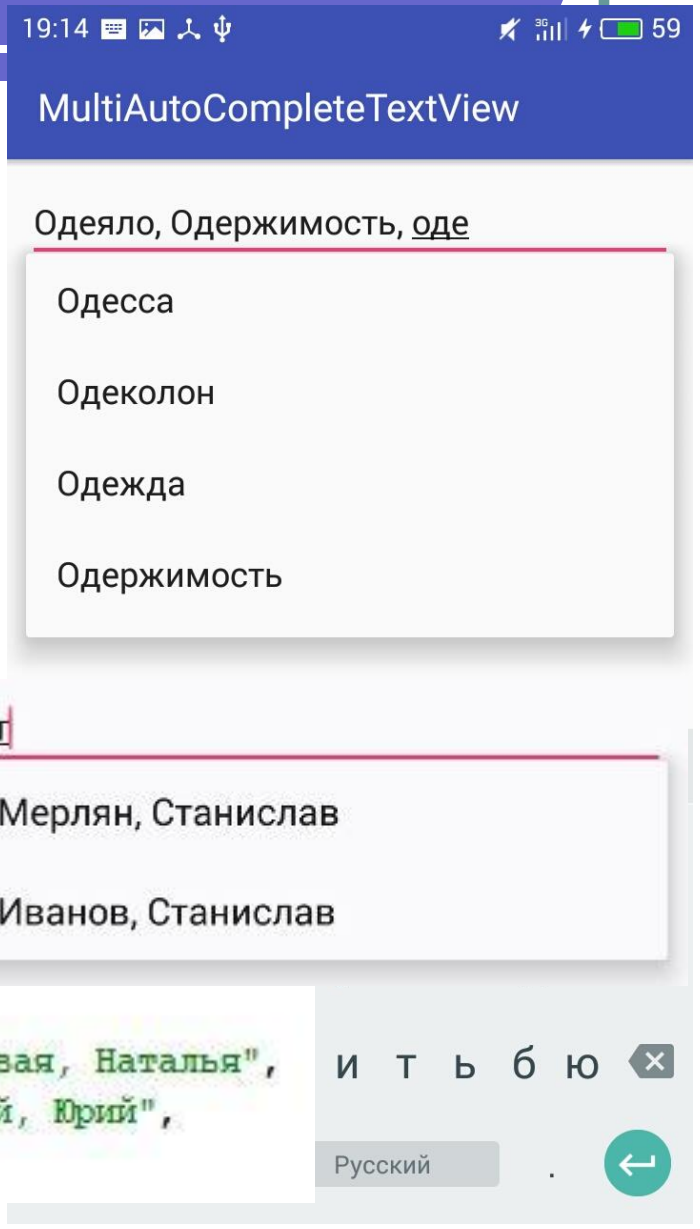



```
AutoCompleteTextView actv =
        (AutoCompleteTextView) findViewById(R.id.autoCompleteTextView1);

String[] suggestions  = getResources().getStringArray(R.array.array1);
ArrayAdapter<String> adapter = new ArrayAdapter<>
        (this, android.R.layout.simple_list_item_1, suggestions);

actv.setAdapter(adapter);
```

MultiAutoCompleteTextView

Этот элемент управления сможет выдавать предположительные варианты-подсказки несколько раз, и при выборе варианта добавляет его в строку, плюс ставит токенайзер. Но, очень важно не забыть выставить что-то вроде

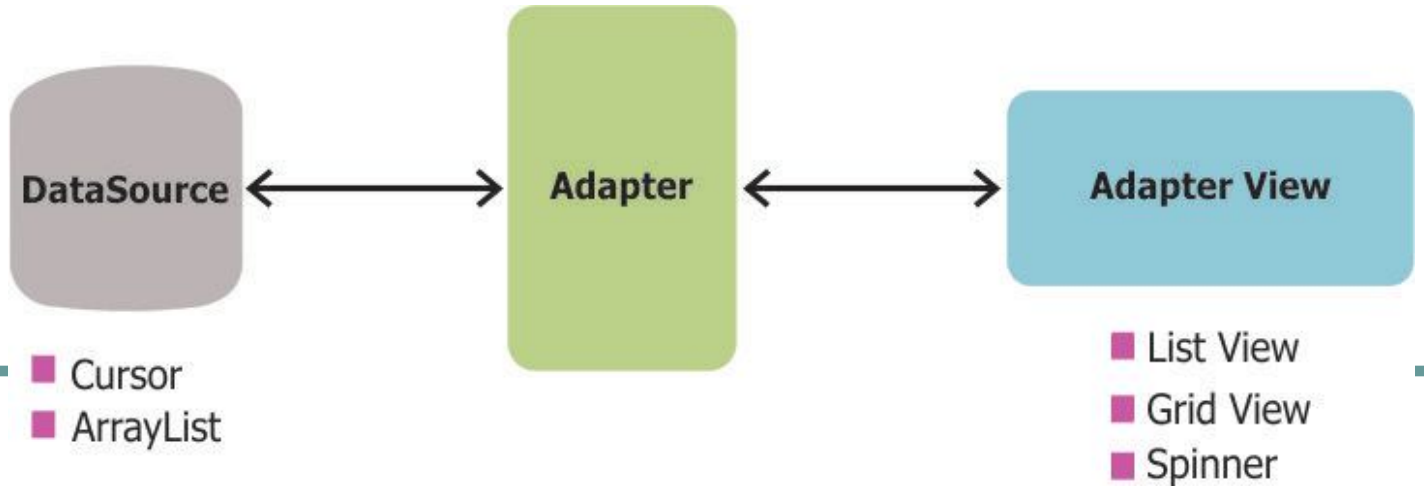
actv.setTokenizer(
new MultiAutoCompleteTextView.
CommaTokenizer());




```
// массив строк может быть взят из кода Java
final String[] mContacts = {
        "Мерлян, Станислав", "Саламаха, Валентин", "Комлевая, Наталья",
        "Топор, Елена", "Серебряков, Сергей", "Евтушевский, Юрий",
        "Иванов, Станислав"};
```

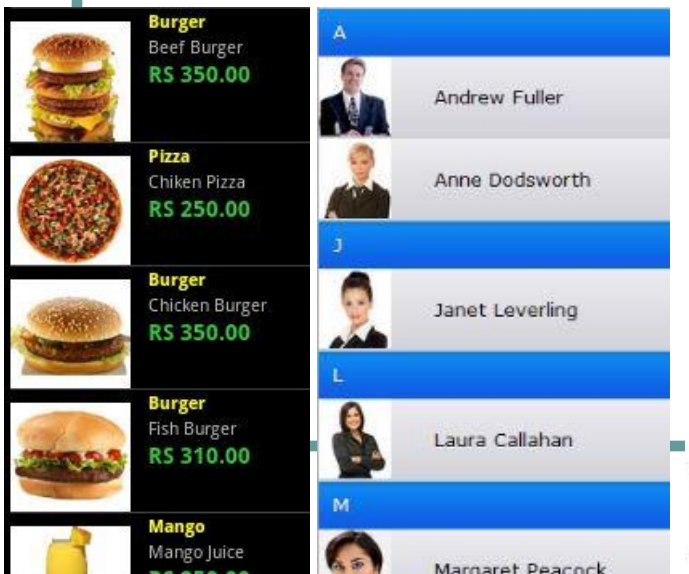
Назначение адаптеров

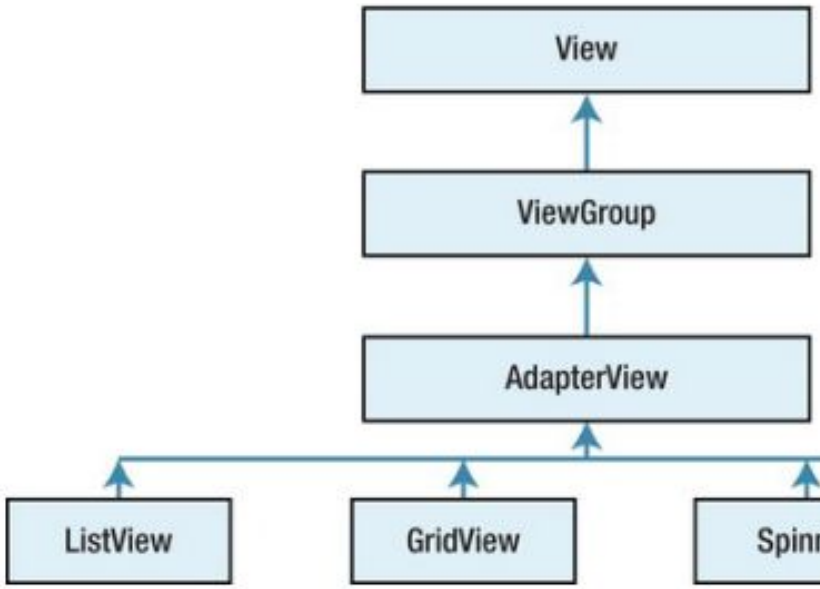
Классы-адаптеры, вроде ArrayAdapter, используются в Android довольно часто. В общих чертах, адаптеры упрощают связывание произвольных данных, взятых из коллекций или БД, с некоторым элементом управления. Адаптеры используются при работе с виджетами вроде **AutoCompleteTextView**, **ListView**, **ExpandableListView**, **GridView**, **Spinner**, **Gallery**, активити **ListActivity** и тп.



Назначение адаптеров

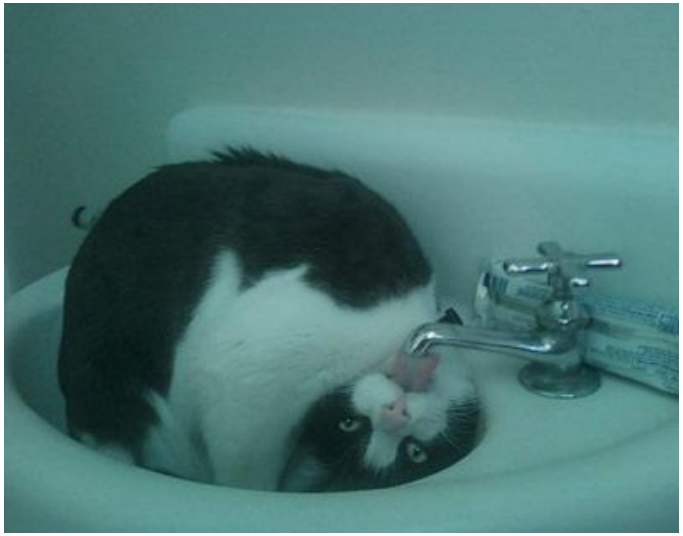
Представим, что есть коллекция объектов типа **Student** (с именами и фоточками), и какая-нибудь списковая вьюшка вроде спиннера. Назначение адаптера заключается в том, чтобы сформировать дочерние вьюшки для этого списка. Адаптер берёт необходимые данные из коллекции и строит каждую дочернюю вьюшку.



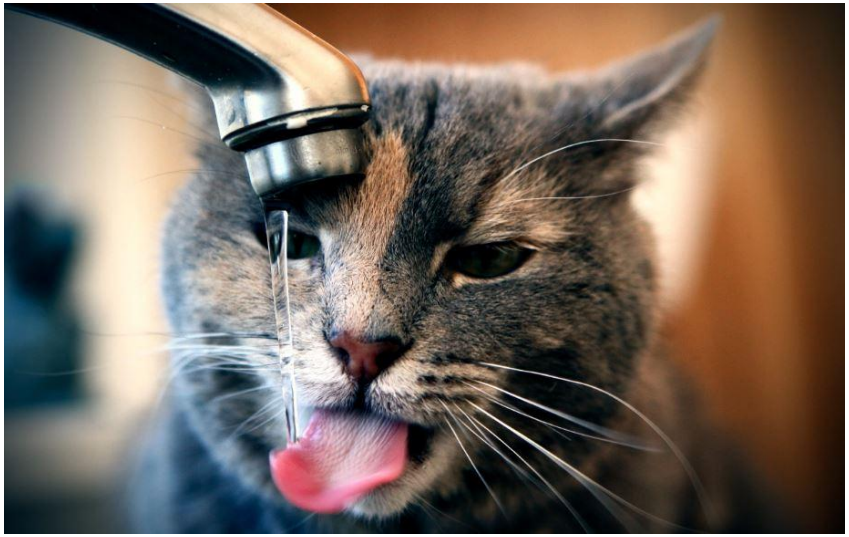


Пример адаптера в жизни

Что такое вообще адаптер? Это переходник между двумя какими-то предметами. Допустим, между питьевой водой и котом требуется адаптер в виде крана.



Плохо спроектированный адаптер



Нормальный адаптер

Адаптеры в Android

В Android часто используется список на основе **ListView**. Сам список состоит из множества элементов **TextView**, которые идут друг за другом. Но их количество будет зависеть от того, что нужно отобразить. Если это дни недели, то достаточно семи элементов, если месяцы, то уже двенадцать, ну а если нужен список продуктов в магазине, то счёт пойдёт на сотни… Короче говоря, нужно найти данные, например, в массиве или базе, а потом скормить их списку. Адаптер этим и занимается. Он берёт по порядку предоставленные данные и размещает их в списке. При этом адаптер на лету создаёт нужные компоненты **TextView** и помещает в них приготовленный текст. Ещё можно придумать свой адаптер, но существуют уже готовые адаптеры на самые распространённые случаи и их предназначение можно определить по именам. Например, **ArrayAdapter** использует массив, а **CursorAdapter** работает с объектом **Cursor**, который используется в базах данных.

Готовые адаптеры

- **ArrayAdapter<T>** - данные представлены в виде массива, и размещаются в отдельных элементах TextView
- **ListAdapter** - адаптер между ListView и данными
- **WrapperListAdapter** - ещё один адаптер для списков
- **HeaderViewListAdapter** - расширенный вариант ListAdapter, когда у ListView есть заголовки
- **SpinnerAdapter** - адаптер для связки данных с элементом Spinner
- **SimpleAdapter** - адаптер, позволяющий заполнить данными список более сложной структуры
- **CursorAdapter** - предоставляет данные для списка через курсор из БД
- **ResourceCursorAdapter** - этот адаптер дополняет CursorAdapter и может создавать вьюшки из ресурсов
- **SimpleCursorAdapter** - дополняет ResourceCursorAdapter и создаёт компоненты TextView/ImageView из столбцов, содержащихся в курсоре. Компоненты определяются в ресурсах

BaseAdapter

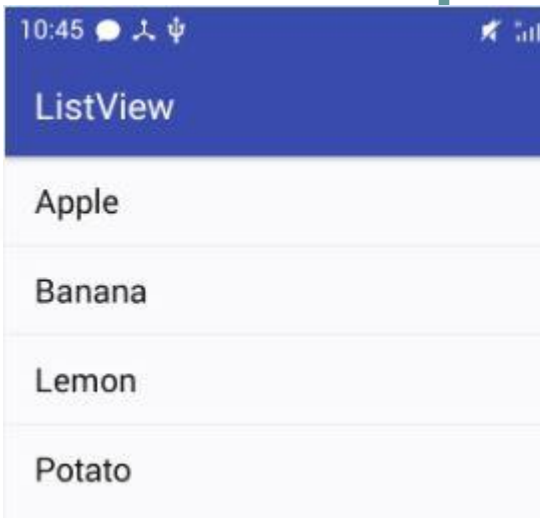
Стандартные адаптеры не всегда покрывают потребности программиста. Если нужен особый специальный адаптер, то в Android есть абстрактный класс **BaseAdapter**, который можно расширить. Собственный адаптер необходим в тех случаях, когда требуется специальное управление данными или дополнительный контроль над отображением дочерних вьюшек. Кроме того, можно предусмотреть в адаптере элементы кэширования для повышения производительности работы.

http://developer.alexanderklimov.ru/android/theory/adapters.php

ListView – пример 1

- Добавляем в activity_main.xml

```
<ListView
    android:id="@+id/list1"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```



- В методе onCreate пишем

```
ListView list = (ListView) findViewById(R.id.list1);
String[] data = {"Apple", "Banana", "Lemon", "Potato"};
ArrayAdapter<String> adapter =
        new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, data);
list.setAdapter(adapter);
```

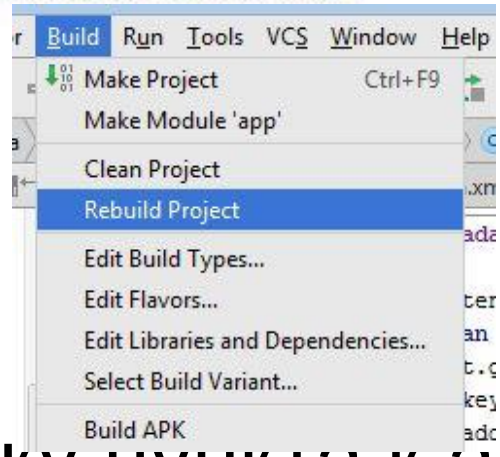
http://www.vogella.com/tutorials/AndroidListView/article.html

ListView – пример 2

- Добавляем файл my_item.xml

MainActivity.java | my_item.xml | activity_main.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textColor="@color/colorAccent"
    android:textSize="25sp"
    android:padding="10dp"
    android:fontFamily="casual"
    android:textStyle="bold" >
</TextView>
```



Apple

Banana

Lemon

Potato

- Подключаем разметку пункта к адаптеру

```
ArrayAdapter<String> adapter =
        new ArrayAdapter<>(this, R.layout.my_item, data);
```

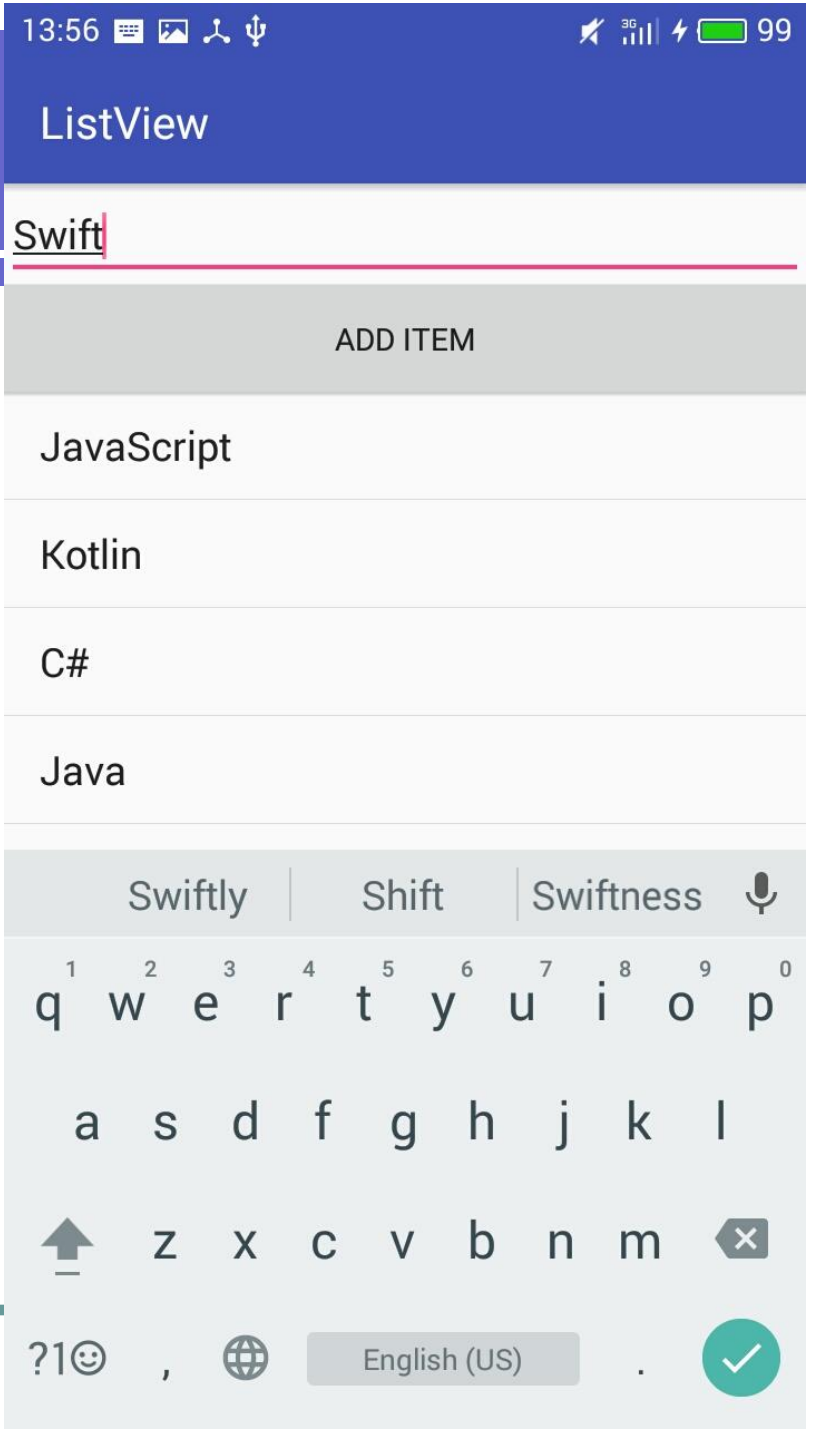
ListView – пример 3

- XML:

https://git.io/vi49l

- Java:

https://git.io/vi49z



Практика. Список контактов

- Добавить в манифест **<uses-permission android:name="android.permission.READ_CONTACTS" />**
- В activity_main.xml сделать любой макет, и добавить в него ListView с **android:id="@+id/list"**
- Код файла MainActivity.java:

  https://git.io/viulL

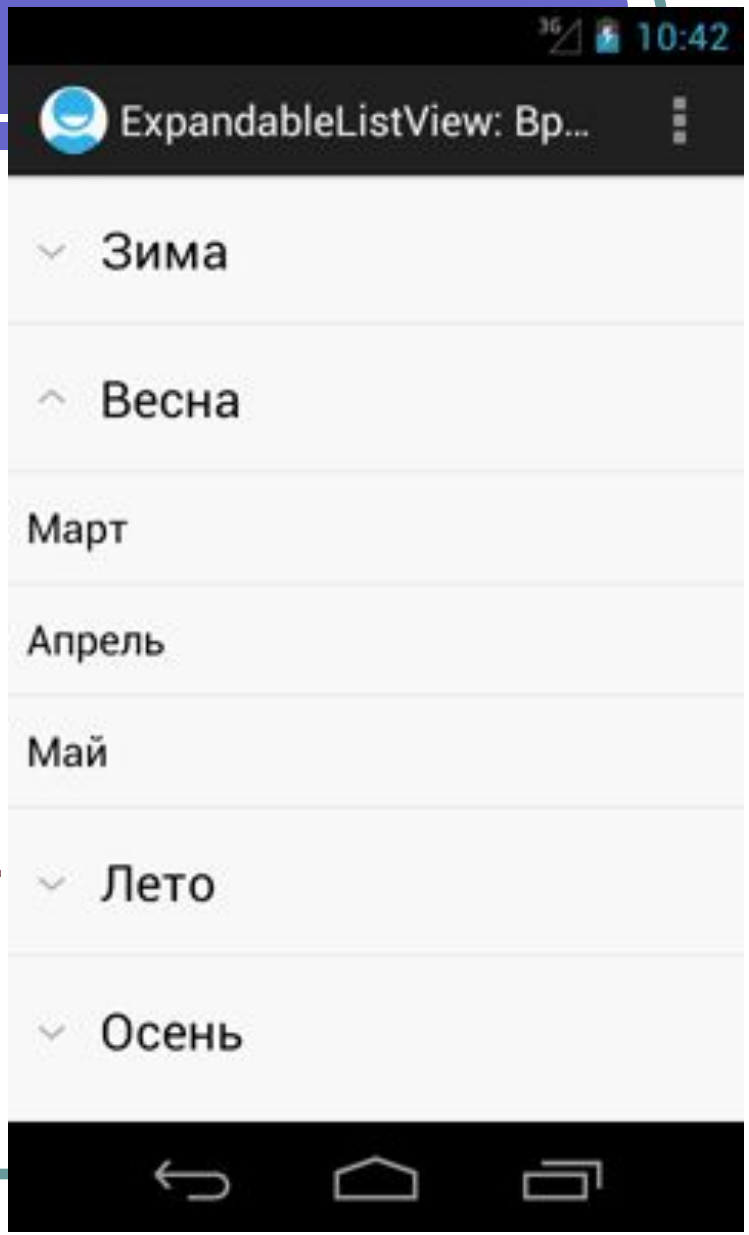
Настройки ListView

- **android:divider** (разделительная полоска)
- **android:listSelecto**r (фон пункта)
- **android:choiceMode** (множественный выбор)
- Кнопка под списком
- Плавная прокрутка
- http://developer.alexanderklimov.ru/android/views/listview.php

ExpandableListView

Эта вьюшка является расширенным вариантом компонента **ListView**. Основное отличие - разворачивающий список второго уровня: список в списке

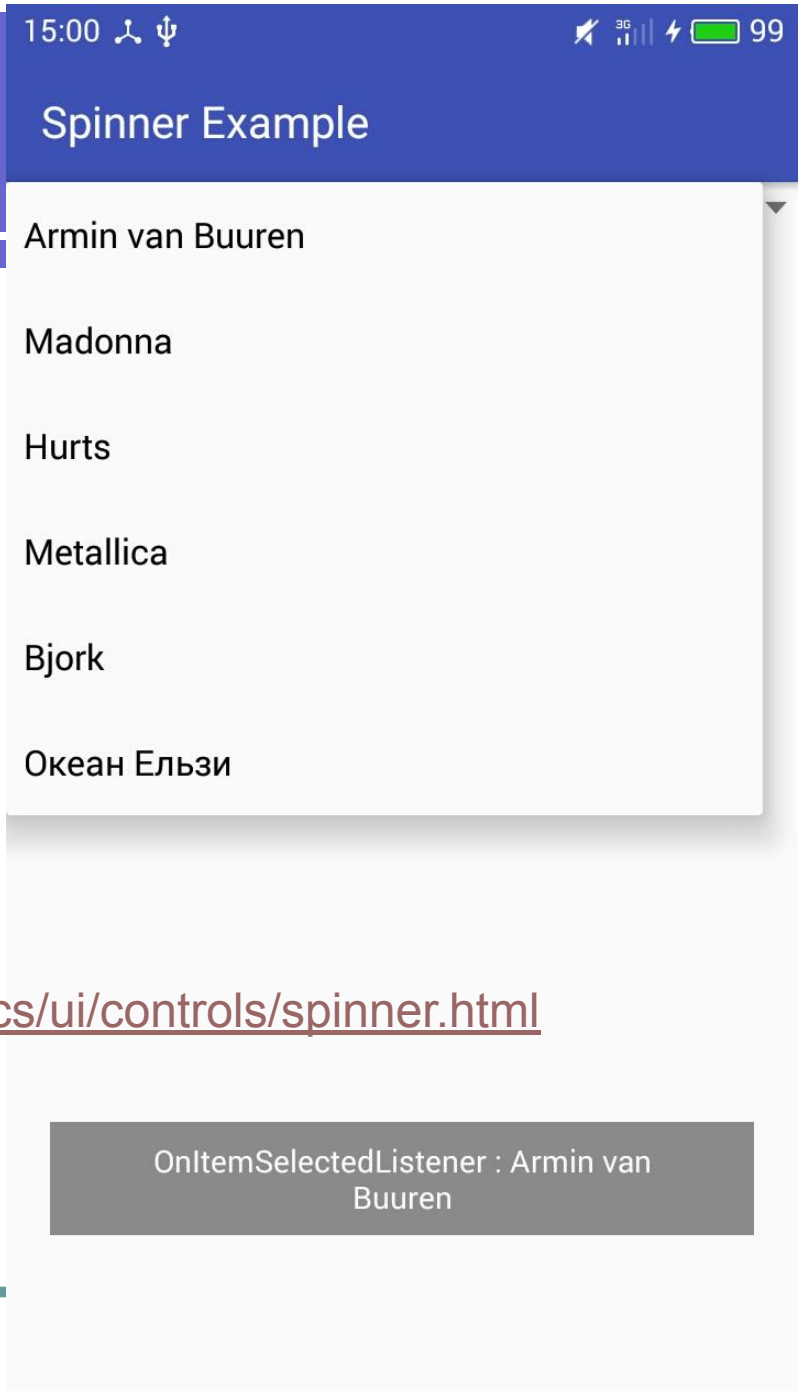
http://developer.alexanderklimov.ru/android/views/expandablelistview.php



Spinner

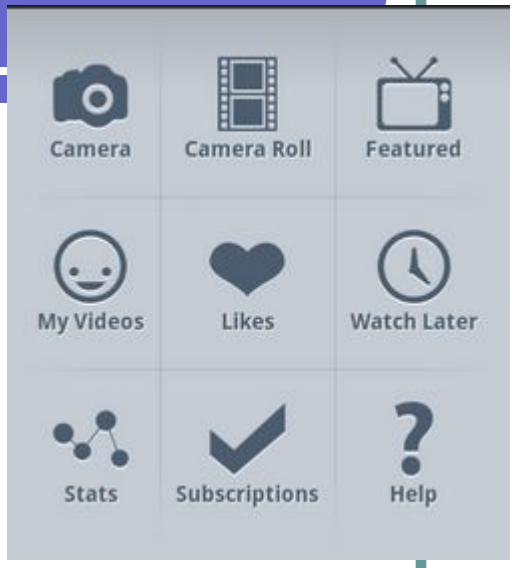
https://git.io/vi4dQ

https://developer.android.com/guide/topics/ui/controls/spinner.html

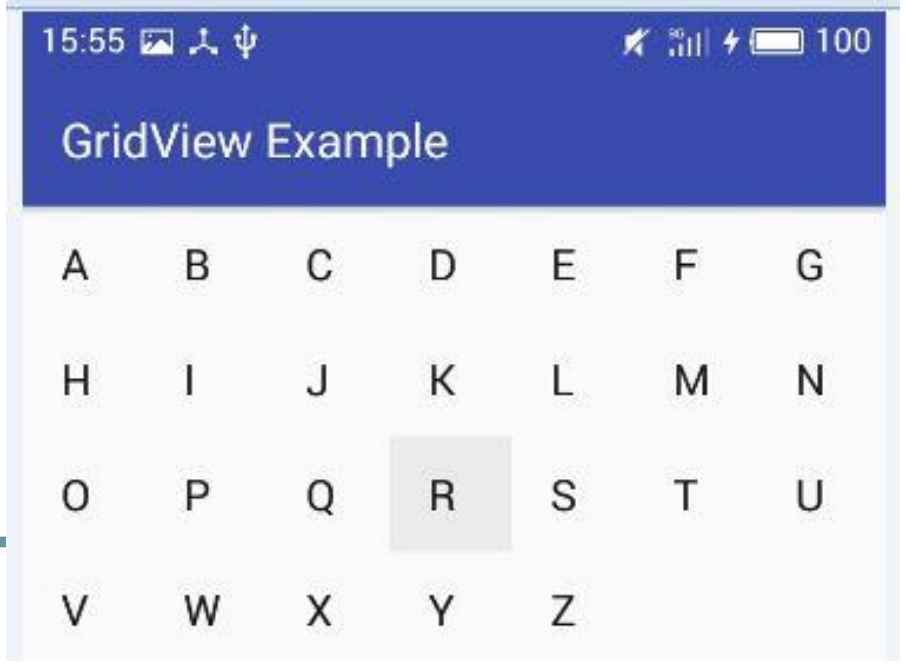


GridView

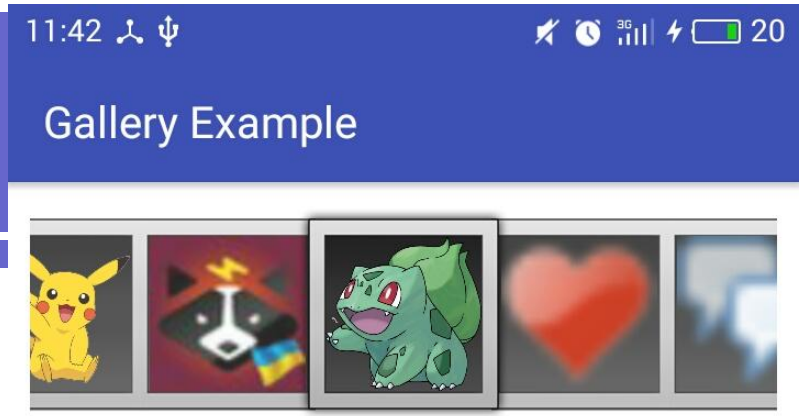
```xml
<?xml version="1.0" encoding="utf-8"?>
<GridView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gridView1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:columnWidth="50dp"
    android:gravity="center"
    android:numColumns="auto_fit"
    android:stretchMode="columnWidth"></GridView>
```



https://git.io/vi4Aq



Gallery



- Создать файл res/values/attrs.xml

```
<resources>
    <declare-styleable name="MyGallery">
        <attr name="android:galleryItemBackground" />
    </declare-styleable>
</resources>
```

- Activity_main.xml:
- **https://git.io/viEQQ**
- MainActivity.java:
- **https://git.io/viEQx**

http://www.androidinterview.com/android-gallery-view-example-displaying-a-list-of-images/

ViewPager

- Добавить в build.gradle (dependencies)

compile 'com.android.support:support-v4:24.0.0'

- **Код activity_main.xml**

```
<android.support.v4.view.ViewPager
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/view_pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

- **Код MainActivity.java**

https://git.io/viEAB

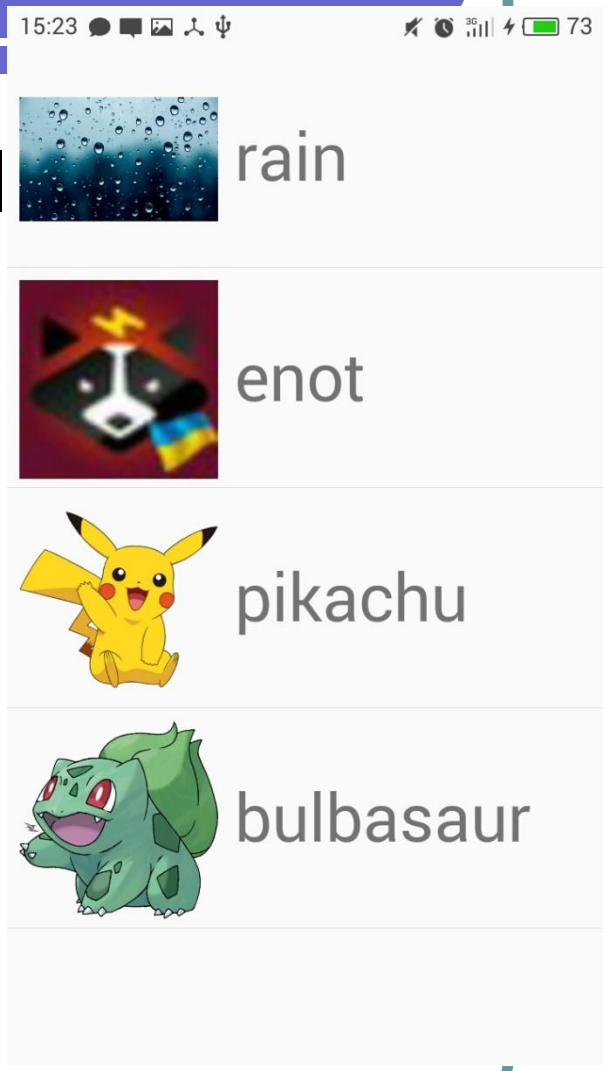
ListActivity + MyArrayAdapter

- Файл res/layout/my_item.xml

https://git.io/viueF

- Файл MainActivity.java

https://git.io/viuve



Simple Adapter

Конструктор этого адаптера выглядит так:

```
SimpleAdapter(Context context,
    List<? extends Map<String, ?>> data,
    int resource,
    String[] from,
    int[] to)
```

Название адаптера несколько обманчиво ☺

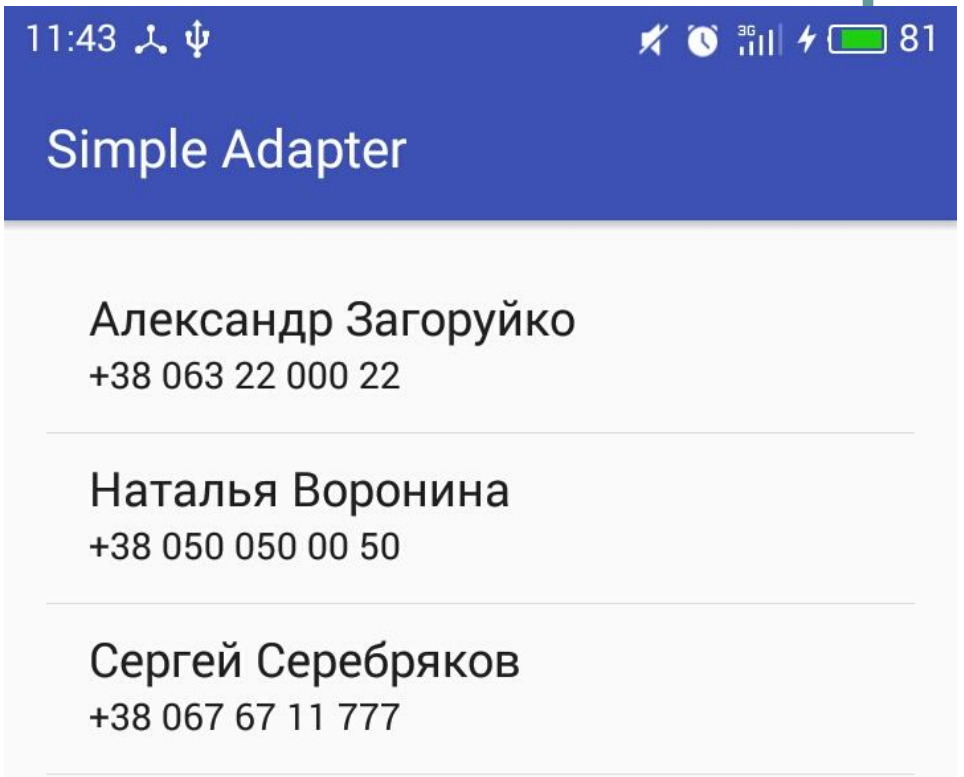
Описание параметров

В параметре **data** используется коллекция Map-объектов или её наследников, например, **HashMap**. Каждый **Map** содержит данные для отдельного элемента списка. Чтобы адаптер понимал, какие данные нужно вставлять во вьюшки каждого пункта списка, указывается два массива **from** и **to**. В массиве **from** используются ключи из **Map**, а в массиве **to** – айди компонентов. Адаптер последовательно перебирает все компоненты из массива **to** и сопоставляет им соответствующие значения из **from**. Важно, чтобы в массивах **to** и **from** было одинаковое количество компонентов!

Пример на SimpleAdapter

- Java-код:

https://git.io/viVcD



- Разметка:

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

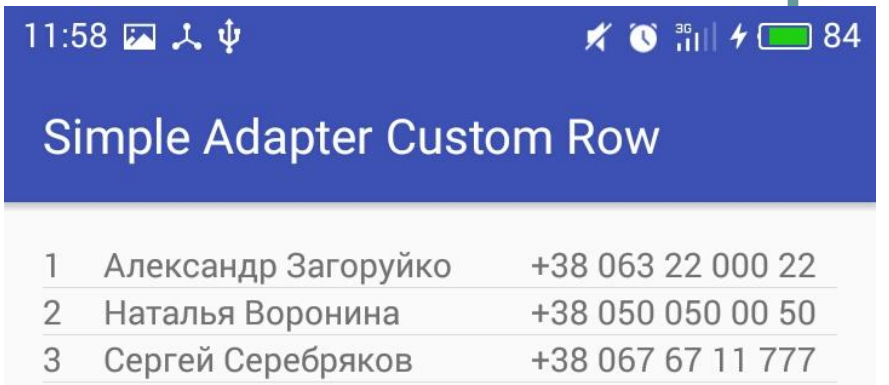
SimpleAdapter Custom Row Layout

- my_item.xml:

https://git.io/viVcQ

- MainActivity.java

https://git.io/viVcb



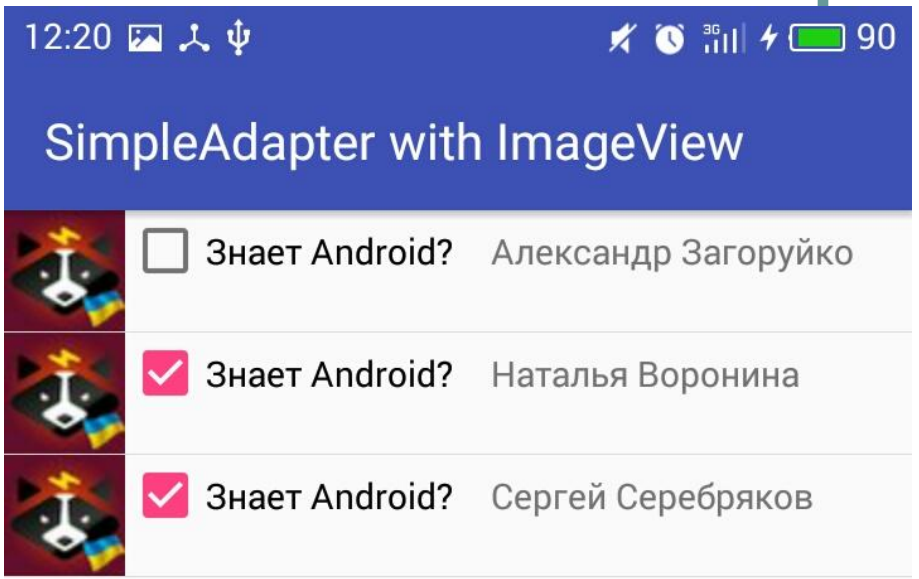
SimpleAdapter with ImageView

- my_item.xml:

https://git.io/viVCv

- MainActivity.java:

https://git.io/viVCT



BaseAdapter

- **BaseAdapter** is a common base class of a general implementation of an Adapter that can be used in ListView, GridView, Spinner etc.
- Whenever you need a customized list in a ListView or customized grids in a GridView you create your own adapter and extend base adapter
- Base Adapter can be extended to create a custom Adapter for displaying a custom list item.
- ArrayAdapter is also an implementation of BaseAdapter!

CustomAdapter minimal code

```
public class CustomAdapter extends BaseAdapter {
    @Override public int getCount() {
        return 0;
    }
    @Override public Object getItem(int i) {
        return null;
    }
    @Override public long getItemId(int i) {
        return 0;
    }
    @Override public View getView(int i, View view,
                                  ViewGroup viewGroup) {
        return null;
    }
}
```

getCount()

The getCount() function returns the total number of items to be displayed in a list.

```
@Override public int getCount() {
    int count = arrayList.size();
    return count;
}
```

getItem()

This function is used to get the data item associated with the specified position in the data set to obtain the corresponding data of the specific location in the collection of data items.

```
@Override public int getItem(int i) {
    return arrayList.get(i);
}
```

getItemId()

As for the getItemId (int position), it returns the corresponding to the position item ID. The function returns a long value of item position to the adapter.

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

getView()

This function is automatically called when the list item view is ready to be displayed. In this function we set the layout for list items using LayoutInflater class and then add the data to the views like ImageView, TextView etc.

```
@Override
public View getView(int i, View view, ViewGroup vg) {
    view = inf.inflate(R.layout.activity_main, null);
ImageView icon = (ImageView)
view.findViewById(R.id.icon);
    icon.setImageResource(flags[i]);
}
```

Extends BaseAdapter

- activity_main.xml

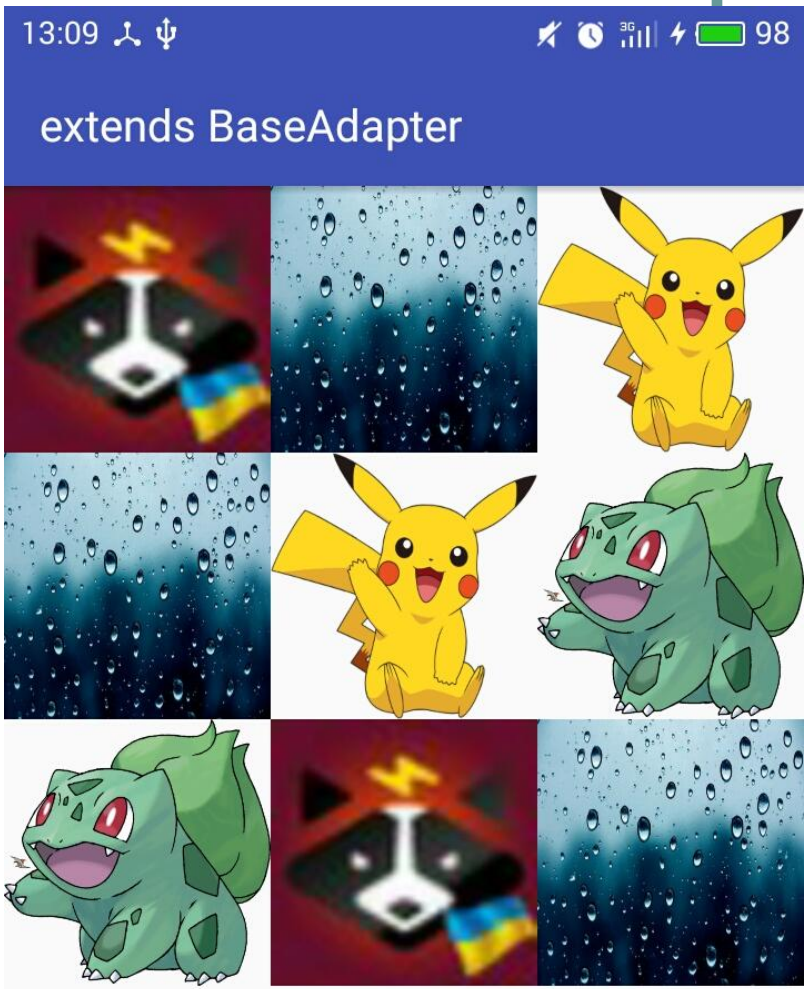
https://git.io/viVCq

- my_item.xml

https://git.io/viVCc

- MainActivity.java

https://git.io/viVCl

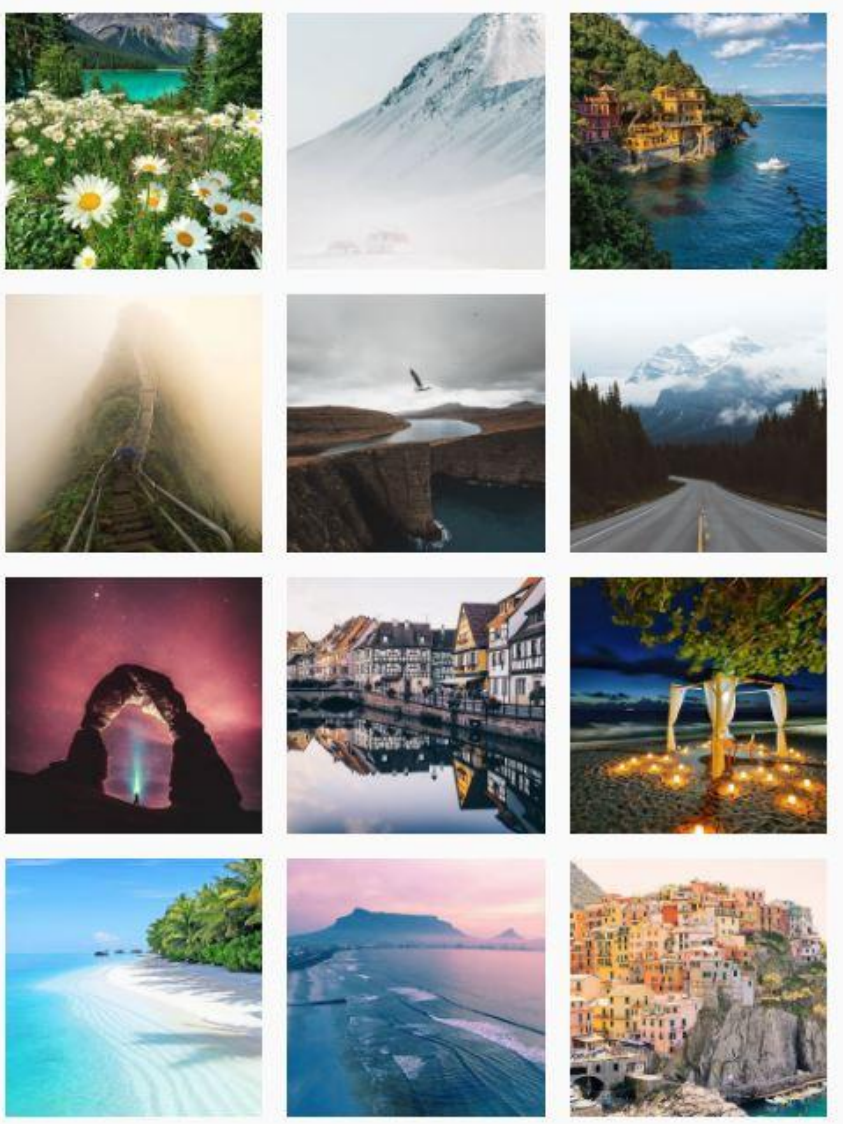


Домашнее задание №1 и 2

- Загрузить список телефонных контактов в SimpleAdapter с кастомной разметкой (в идеале, пункт списка – это аватарка контакта, имя и фамилия крупным шрифтом, номер телефона шрифтом помельче)
- Загрузить в ListActivity набор карточек (создать что-то вроде новостной ленты)

Домашнее задание №3 и 4

- Сделать GridView, в который загружается 30-40 картинок из интернета (инструкция, например, такая: http://stacktips.com/tutorials/android/download-and-display-image-in-android-gridview)
- Реализовать игру «Пятнашки»



Домашнее задание №5

Реализовать приложение «**Список желаний**».

На экране появляется список предметов, которые вы хотели бы получить в подарок.

Пункт списка выглядит как:

картинка – название товара – цена в долларах – чекбокс

Предметы хранятся в листе, количество элементов может меняться (все элементы создаются программно). Под списком есть текстовое поле, которое показывает общую стоимость всех подарков. Также, есть 3 радио-кнопки, которые позволяют пересчитать стоимость в долларах / евро / гривнах по текущему курсу (курс берётся с сайта национального банка Украины)