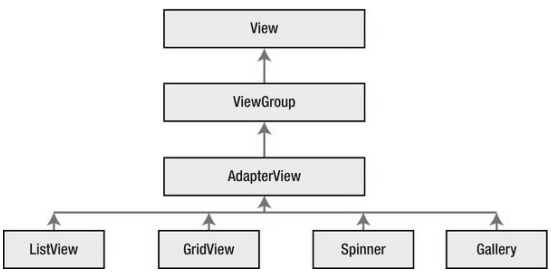
**ListView и ArrayAdapter**

Android представляет широкую палитру элементов,которые представляют списки. Все они является наследниками класса **android.widget.AdapterView**. Это такие виджеты как ListView, GridView, Spinner. Они могут выступать контейнерами для других элементов управления

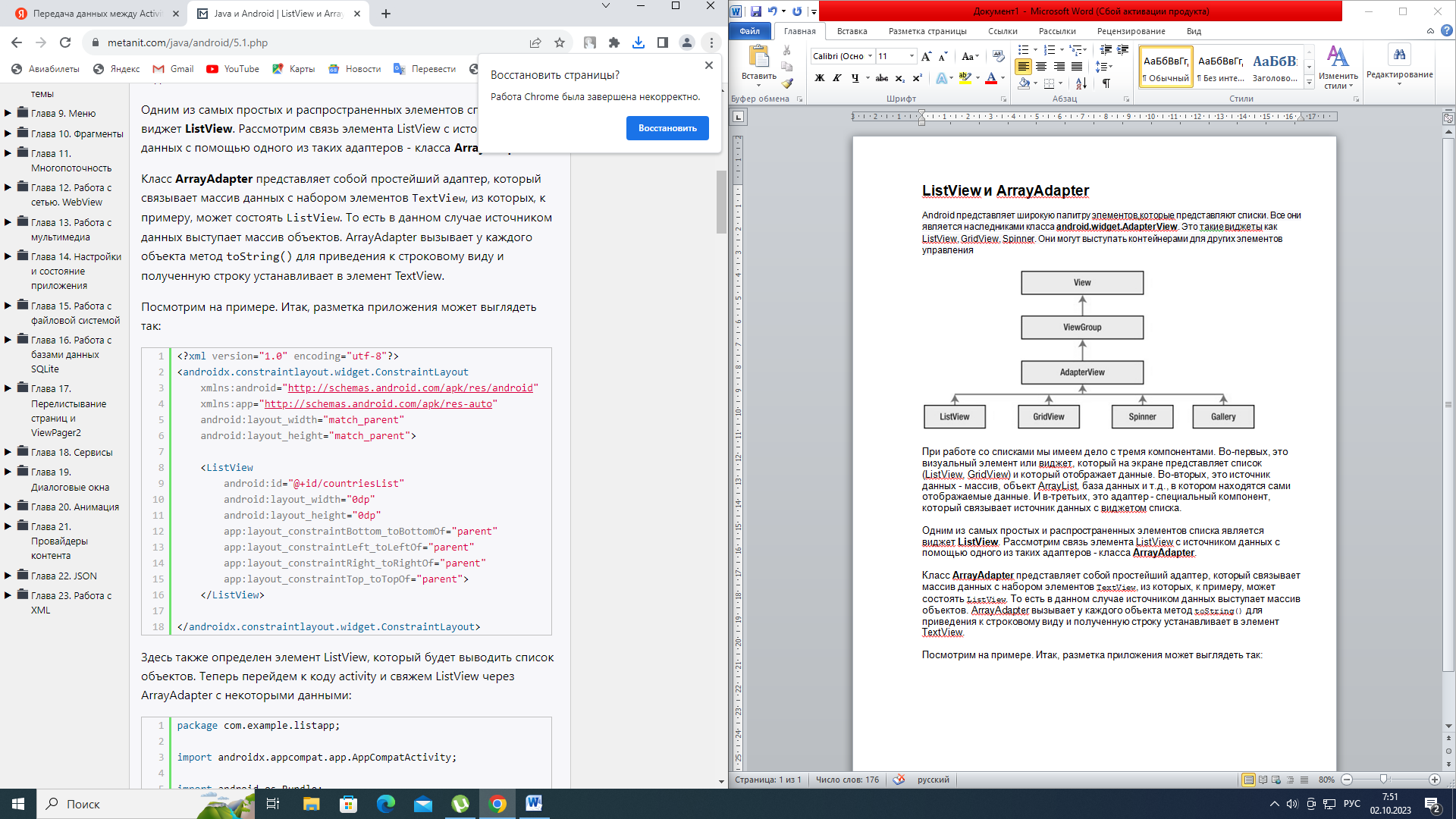


При работе со списками мы имеем дело с тремя компонентами. Во-первых, это визуальный элемент или виджет, который на экране представляет список (ListView, GridView) и который отображает данные. Во-вторых, это источник данных - массив, объект ArrayList, база данных и т.д., в котором находятся сами отображаемые данные. И в-третьих, это адаптер - специальный компонент, который связывает источник данных с виджетом списка.

Одним из самых простых и распространенных элементов списка является виджет **ListView**. Рассмотрим связь элемента ListView с источником данных с помощью одного из таких адаптеров - класса **ArrayAdapter**.

Класс **ArrayAdapter** представляет собой простейший адаптер, который связывает массив данных с набором элементов TextView, из которых, к примеру, может состоять ListView. То есть в данном случае источником данных выступает массив объектов. ArrayAdapter вызывает у каждого объекта метод toString() для приведения к строковому виду и полученную строку устанавливает в элемент TextView.

Посмотрим на примере. Итак, разметка приложения может выглядеть так:



Здесь также определен элемент ListView, который будет выводить список объектов. Теперь перейдем к коду activity и свяжем ListView через ArrayAdapter с некоторыми данными:

package com.example.listapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.widget.ArrayAdapter;

import android.widget.ListView;

public class MainActivity extends AppCompatActivity {

    // набор данных, которые свяжем со списком

    String[] countries = { "Бразилия", "Аргентина", "Колумбия", "Чили", "Уругвай"};

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // получаем элемент ListView

        ListView countriesList = findViewById(R.id.countriesList);

        // создаем адаптер

        ArrayAdapter<String> adapter = new ArrayAdapter(this,

                android.R.layout.simple\_list\_item\_1, countries);

        // устанавливаем для списка адаптер

        countriesList.setAdapter(adapter);

    }

}

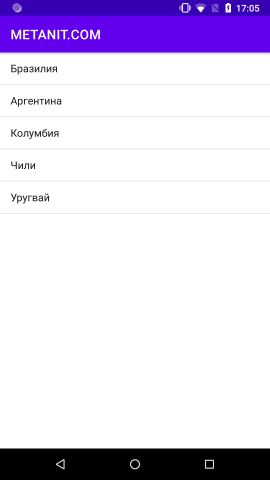
Здесь вначале получаем по id элемент ListView и затем создаем для него адаптер.

Для создания адаптера использовался следующий конструктор ArrayAdapter<String>(this,android.R.layout.simple\_list\_item\_1, countries), где

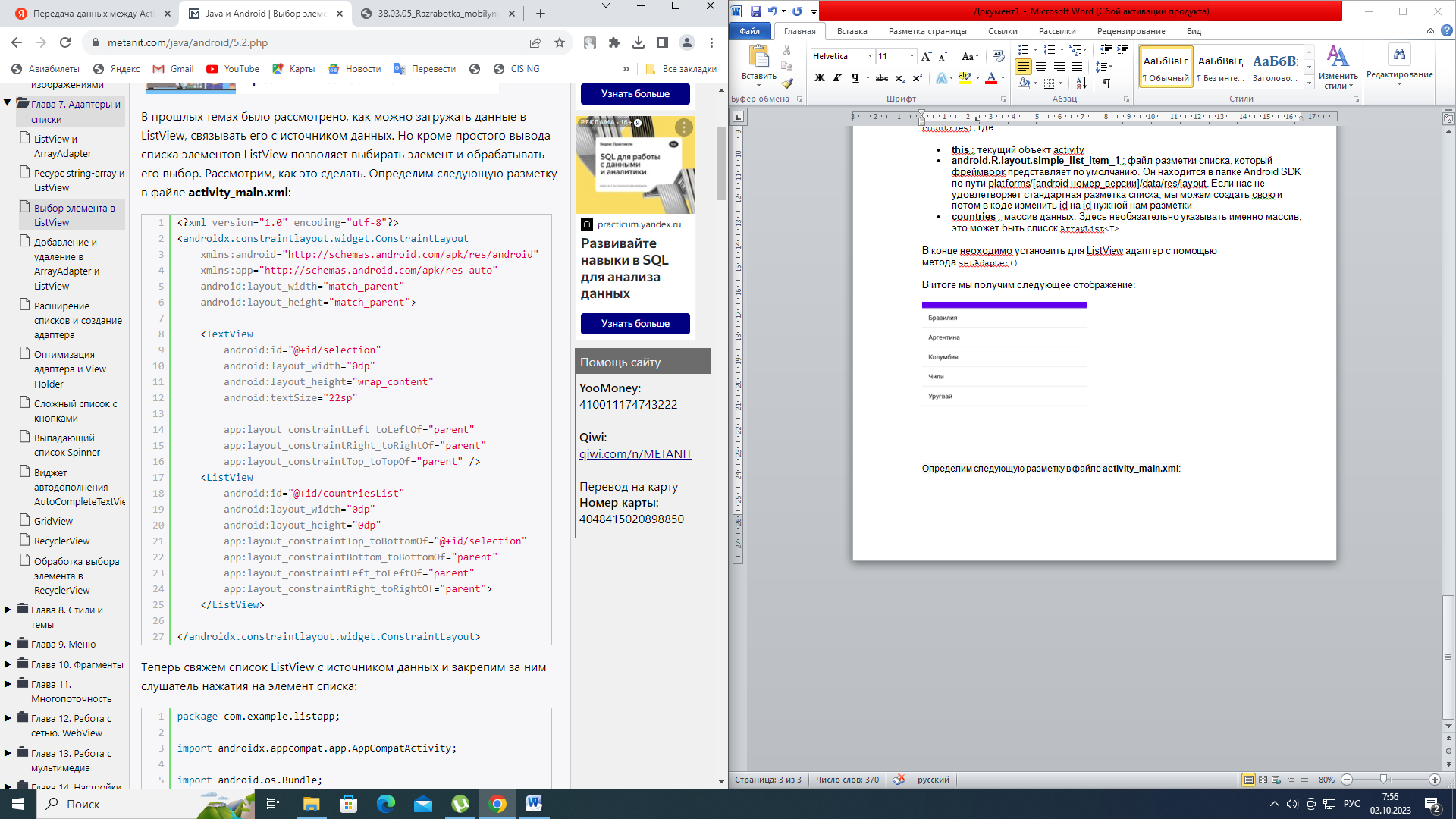
* **this** : текущий объект activity
* **android.R.layout.simple\_list\_item\_1** : файл разметки списка, который фреймворк представляет по умолчанию. Он находится в папке Android SDK по пути platforms/[android-номер\_версии]/data/res/layout. Если нас не удовлетворяет стандартная разметка списка, мы можем создать свою и потом в коде изменить id на id нужной нам разметки
* **countries** : массив данных. Здесь необязательно указывать именно массив, это может быть список ArrayList<T>.

В конце неоходимо установить для ListView адаптер с помощью метода setAdapter().

В итоге мы получим следующее отображение:



Определим следующую разметку в файле **activity\_main.xml**:



Теперь свяжем список ListView с источником данных и закрепим за ним слушатель нажатия на элемент списка:

package com.example.listapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    String[] countries = { "Бразилия", "Аргентина", "Колумбия", "Чили", "Уругвай"};

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // получаем элемент TextView

        TextView selection = findViewById(R.id.selection);

        // получаем элемент ListView

        ListView countriesList = findViewById(R.id.countriesList);

        // создаем адаптер

        ArrayAdapter<String> adapter = new ArrayAdapter(this,

                android.R.layout.simple\_list\_item\_1, countries);

        // устанавливаем для списка адаптер

        countriesList.setAdapter(adapter);

        // добавляем для списка слушатель

        countriesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

            @Override

            public void onItemClick(AdapterView<?> parent, View v, int position, long id)

            {

                // по позиции получаем выбранный элемент

                String selectedItem = countries[position];

                // установка текста элемента TextView

                selection.setText(selectedItem);

            }

        });

    }

}

Итак, метод setAdapter связывает элемент ListView с определенным адаптером. Далее для обработки выбора элемента списка устанавливается слушатель **OnItemClickListener**. Этот слушатель имеет один метод **onItemClick**, через параметры которого мы можем получить выделенный элемент и сопутствующие данные. Так, он принимает следующие параметры:

* **parent** : нажатый элемент AdapterView (в роли которого в данном случае выступает наш элемент ListView)
* **view** : нажатый виджет внутри AdapterView
* **position** : индекс нажатого виждета внутри AdapterView
* **id** : идентификатор строки нажатого элемента

Используя эти параметры, мы можем разными способами получить выделенный элемент.

Например, в данном случае получая индекс нажатого виджета, который соответствует индексу элемента в массиве строк, мы можем устанавить соответствующий элемент в массиве строк и таким образом получить его текст:

countriesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

    @Override

    public void onItemClick(AdapterView<?> parent, View v, int position, long id)

    {

        // по позиции получаем выбранный элемент

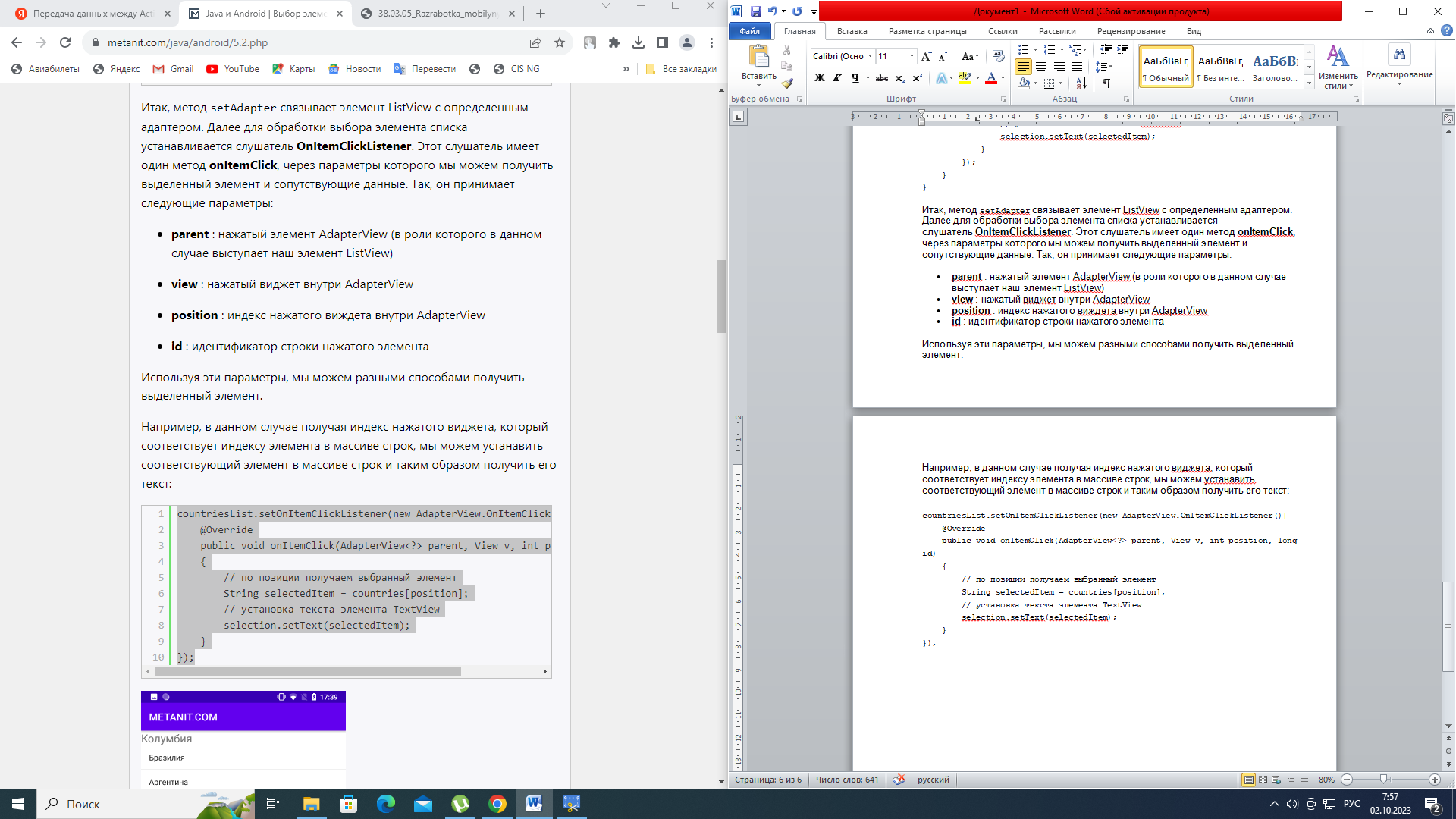
        String selectedItem = countries[position];

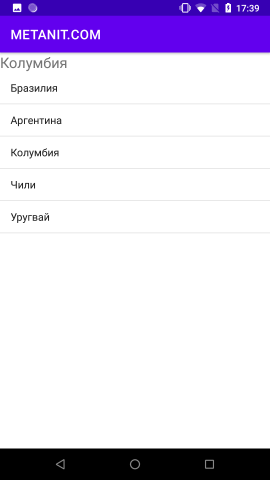
        // установка текста элемента TextView

        selection.setText(selectedItem);

    }

});





Также мы можем получить выделенный элемент из AdapterView, который передается в качестве первого параметра - AdapterView<?> parent. Так, в данном случае мы знаем, что каждый элемент в AdapterView фактически представляет строку или объект String, поэтому в данном случае можно получить выделенный элемент так:

countriesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

    @Override

    public void onItemClick(AdapterView<?> parent, View v, int position, long id)

    {

        // получаем выбранный элемент

        String selectedItem = (String)parent.getItemAtPosition(position);

        // установка текста элемента TextView

        selection.setText(selectedItem);

    }

});

Метод getItemAtPosition возвращает выделенный элемент по индексу. Это может актуально, если мы используем в качестве источника данных не массив строк, созданный в коде Java, а, например, ресурс <string-array>, заданный в файле xml.

В-третьих, мы можем использовать выделенный элемент, который передается в качестве второго параметра - View v. Так, в данном случае адаптер использует в качестве типа разметки ресурс - **android.R.layout.simple\_list\_item\_1**, а это значит, что выделенный элемент представляет элемент **TextView**, в котором выводится данный текст. Поэтому в данном случае мы также могли бы получить выделенный элемент так:

countriesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

    @Override

    public void onItemClick(AdapterView<?> parent, View v, int position, long id)

    {

        // получаем выбранный элемент

        TextView textView = (TextView) v;

        String selectedItem = (String)textView.getText();

        // установка текста элемента TextView

        selection.setText(selectedItem);

        // или так

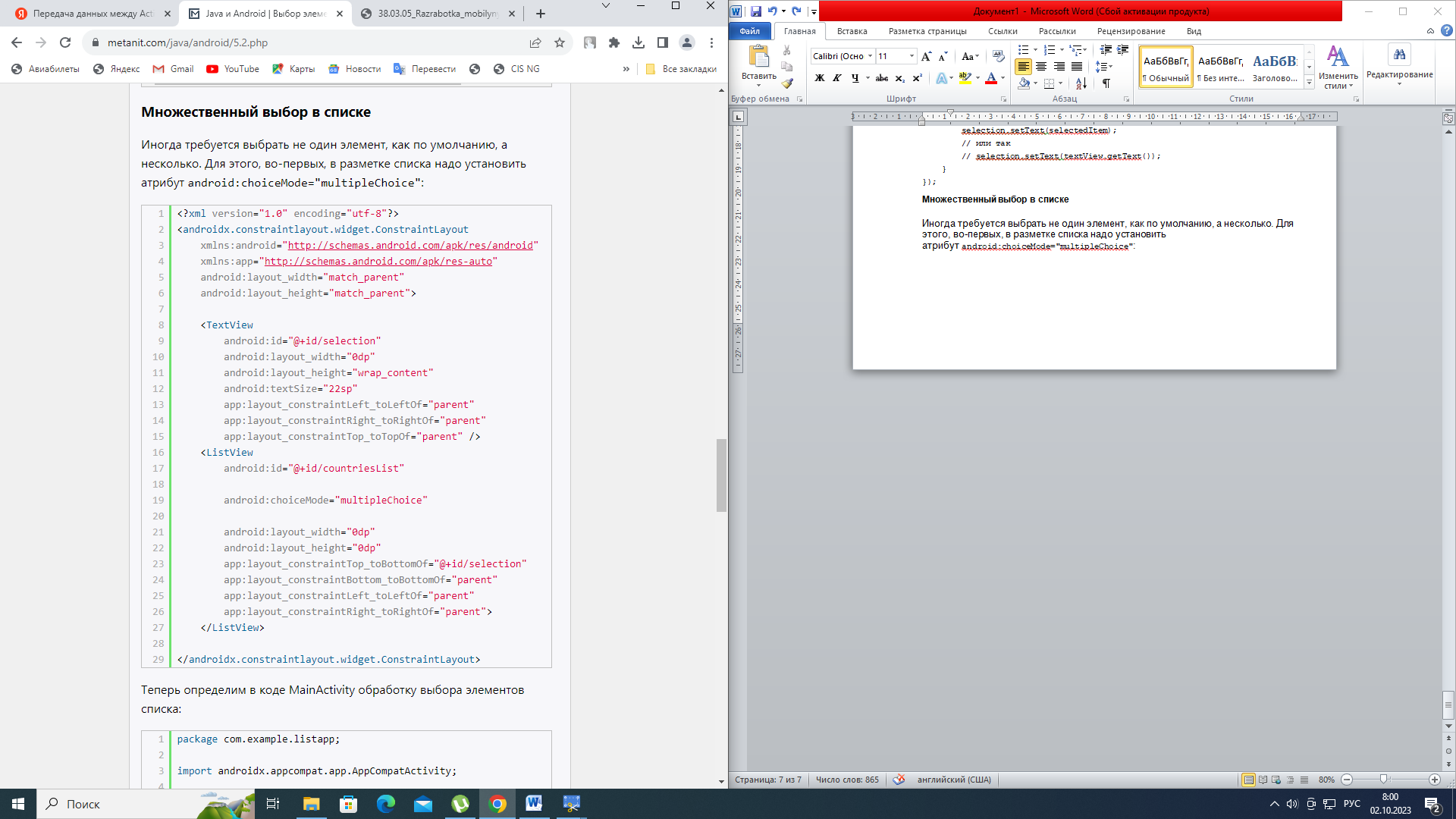
        // selection.setText(textView.getText());

    }

});

### Множественный выбор в списке

Иногда требуется выбрать не один элемент, как по умолчанию, а несколько. Для этого, во-первых, в разметке списка надо установить атрибут android:choiceMode="multipleChoice":



Теперь определим в коде MainActivity обработку выбора элементов списка:

package com.example.listapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.util.SparseBooleanArray;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    String[] countries = { "Бразилия", "Аргентина", "Колумбия", "Чили", "Уругвай"};

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // получаем элемент TextView

        TextView selection = findViewById(R.id.selection);

        // получаем элемент ListView

        ListView countriesList = findViewById(R.id.countriesList);

        // создаем адаптер

        // создаем адаптер

        ArrayAdapter<String> adapter = new ArrayAdapter(this,

                android.R.layout.simple\_list\_item\_multiple\_choice, countries);

        // устанавливаем для списка адаптер

        countriesList.setAdapter(adapter);

        // добавляем для списка слушатель

        countriesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

            @Override

            public void onItemClick(AdapterView<?> parent, View v, int position, long id)

            {

                SparseBooleanArray selected=countriesList.getCheckedItemPositions();

                String selectedItems="";

                for(int i=0;i < countries.length;i++)

                {

                    if(selected.get(i))

                        selectedItems+=countries[i]+",";

                }

                // установка текста элемента TextView

                selection.setText("Выбрано: " + selectedItems);

            }

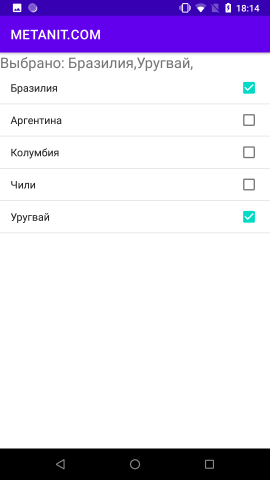
        });

    }

}

Ресурс android.R.layout.simple\_list\_item\_multiple\_choice представляет стандартную разметку, предоставляемую фреймворком, для создания списка с множественным выбором.

А при выборе элементов мы получаем все выбранные позиции в объект **SparseBooleanArray**, затем пробегаемся по всему массиву, и если позиция элемента в массиве есть в SparseBooleanArray, то есть она отмечена, то добавляем отмеченный элемент в строку.



## Добавление и удаление в ArrayAdapter и ListView

После привязки ListView к источнику данных через адаптер мы можем работать с данными - добавлять, удалять, изменять только через адаптер. ListView служит только для отображения данных.

Для управления данными мы можем использовать методы адаптера или напрямую источника данных. Например, класс ArrayAdapter предоставляет следующие методы для управления данными:

* **void add(T object)**: добавляет элемент object в конец массива
* **void addAll(T... items)**: добавляет все элементы items в конец массива
* **void addAll(Collection<? extends T> collection)**: добавляет коллекцию элементов collection в конец массива
* **void clear()**: удаляет все элементы из списка
* **void insert(T object, int index)**: добавляет элемент object в массив по индексу index
* **void remove(T object)**: удаляет элемент object из массива

Однако после применения вышеуказанных методов изменения коснутся только массива, выступающего источником данных. Чтобы синхронизировать изменения с элементом ListView, надо вызвать у адаптера метод notifyDataSetChanged().

Например, определим в файле **activity\_main.xml** следующие элементы:

<?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>"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent">

        <EditText

            android:id="@+id/userName"

            android:layout\_width="0dp"

            android:layout\_height="wrap\_content"

            app:layout\_constraintHorizontal\_weight="4"

            app:layout\_constraintLeft\_toLeftOf="parent"

            app:layout\_constraintRight\_toLeftOf="@+id/add"

            app:layout\_constraintTop\_toTopOf="parent" />

        <Button

            android:id="@+id/add"

            android:layout\_width="0dp"

            android:layout\_height="wrap\_content"

            app:layout\_constraintHorizontal\_weight="1"

            android:text="+"

            android:onClick="add"

            app:layout\_constraintRight\_toLeftOf="@+id/remove"

            app:layout\_constraintLeft\_toRightOf="@+id/userName"

            app:layout\_constraintTop\_toTopOf="parent"/>

        <Button

            android:id="@+id/remove"

            android:layout\_width="0dp"

            android:layout\_height="wrap\_content"

            app:layout\_constraintHorizontal\_weight="1"

            android:text="-"

            android:onClick="remove"

            app:layout\_constraintTop\_toTopOf="parent"

            app:layout\_constraintLeft\_toRightOf="@+id/add"

            app:layout\_constraintRight\_toRightOf="parent" />

    <ListView

        android:id="@+id/usersList"

        android:layout\_width="0dp"

        android:layout\_height="0dp"

        android:choiceMode="multipleChoice"

        app:layout\_constraintTop\_toBottomOf="@+id/userName"

        app:layout\_constraintBottom\_toBottomOf="parent"

        app:layout\_constraintLeft\_toLeftOf="parent"

        app:layout\_constraintRight\_toRightOf="parent">

    </ListView>

</androidx.constraintlayout.widget.ConstraintLayout>

Для вывода списка предназначен ListView с возможностью множественного выбора элементов. Для добавления и удаления определены две кнопки. Для ввода нового объекта в список предназначено поле EditText.

Теперь изменим класс **MainActivity**:

package com.example.listapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.EditText;

import android.widget.ListView;

import java.util.ArrayList;

import java.util.Collections;

public class MainActivity extends AppCompatActivity {

    ArrayList<String> users = new ArrayList<String>();

    ArrayList<String> selectedUsers = new ArrayList<String>();

    ArrayAdapter<String> adapter;

    ListView usersList;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // добавляем начальные элементы

        Collections.addAll(users, "Tom", "Bob", "Sam", "Alice");

        // получаем элемент ListView

        usersList = findViewById(R.id.usersList);

        // создаем адаптер

        adapter = new ArrayAdapter(this, android.R.layout.simple\_list\_item\_multiple\_choice, users);

        // устанавливаем для списка адаптер

        usersList.setAdapter(adapter);

        // обработка установки и снятия отметки в списке

        usersList.setOnItemClickListener(new AdapterView.OnItemClickListener(){

            @Override

            public void onItemClick(AdapterView<?> parent, View v, int position, long id)

            {

                // получаем нажатый элемент

                String user = adapter.getItem(position);

                if(usersList.isItemChecked(position))

                    selectedUsers.add(user);

                else

                    selectedUsers.remove(user);

            }

        });

    }

    public void add(View view){

        EditText userName = findViewById(R.id.userName);

        String user = userName.getText().toString();

        if(!user.isEmpty()){

            adapter.add(user);

            userName.setText("");

            adapter.notifyDataSetChanged();

        }

    }

    public void remove(View view){

        // получаем и удаляем выделенные элементы

        for(int i=0; i< selectedUsers.size();i++){

            adapter.remove(selectedUsers.get(i));

        }

        // снимаем все ранее установленные отметки

        usersList.clearChoices();

        // очищаем массив выбраных объектов

        selectedUsers.clear();

        adapter.notifyDataSetChanged();

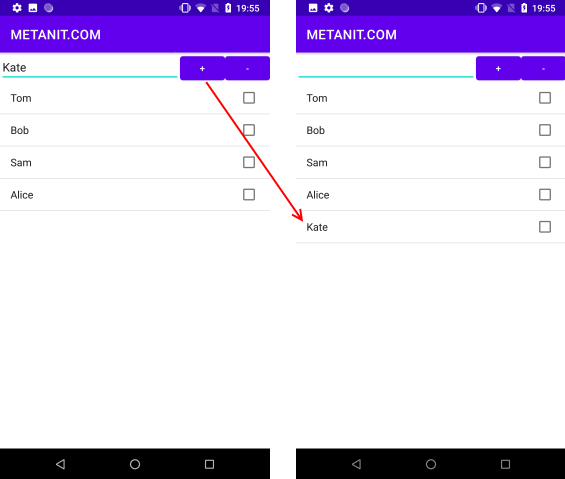
    }

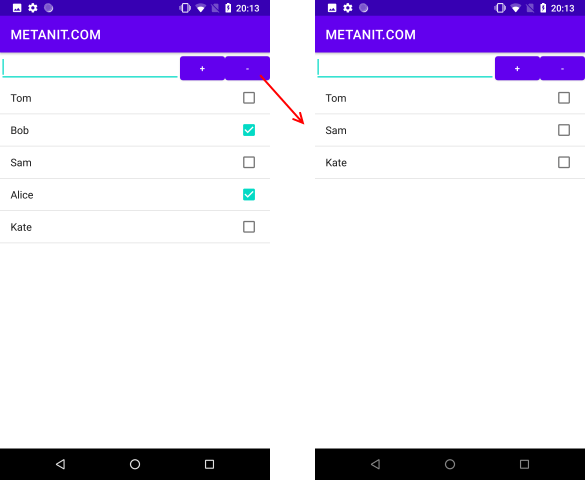
}

С добавлением все относительно просто: получаем введенную строку и добавляем в список с помощью метода **adapter.add()**. Чтобы обновить ListView после добавления вызывается метод **adapter.notifyDataSetChanged()**.

А для удаления создается дополнительный список selectedUsers, который будет содержать выделенные элементы. Для получения выделенных элементов и добавления их в список используется слушатель **AdapterView.OnItemClickListener**, метод onItemClick() которого вызывается при установке или снятия отметки с элемента, то есть при любом нажатии на элемент.

По нажатию на кнопку удаления пробегаемся по списку выделенных элементов и вызываем для каждого из них метод **adapter.remove()**.

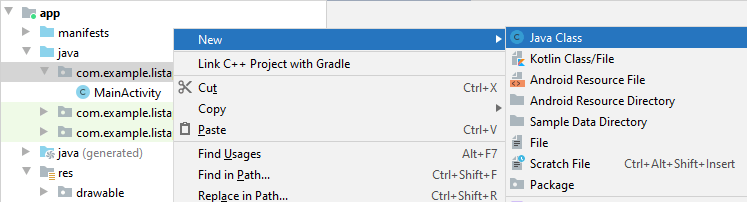




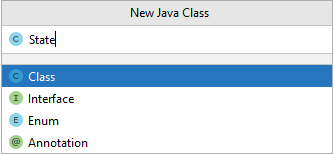
Традиционные списки ListView, использующие стандартные адаптеры ArrayAdapter, прекрасно работают с массивами строк. Однако чаще мы будем сталкиваться с более сложными по структуре списками, где один элемент представляет не одну строку, а несколько строк, картинок и других компонентов.

Для создания сложного списка нам надо переопределить один из используемых адаптеров. Поскольку, как правило, используется ArrayAdapter, то именно его мы и переопределим.

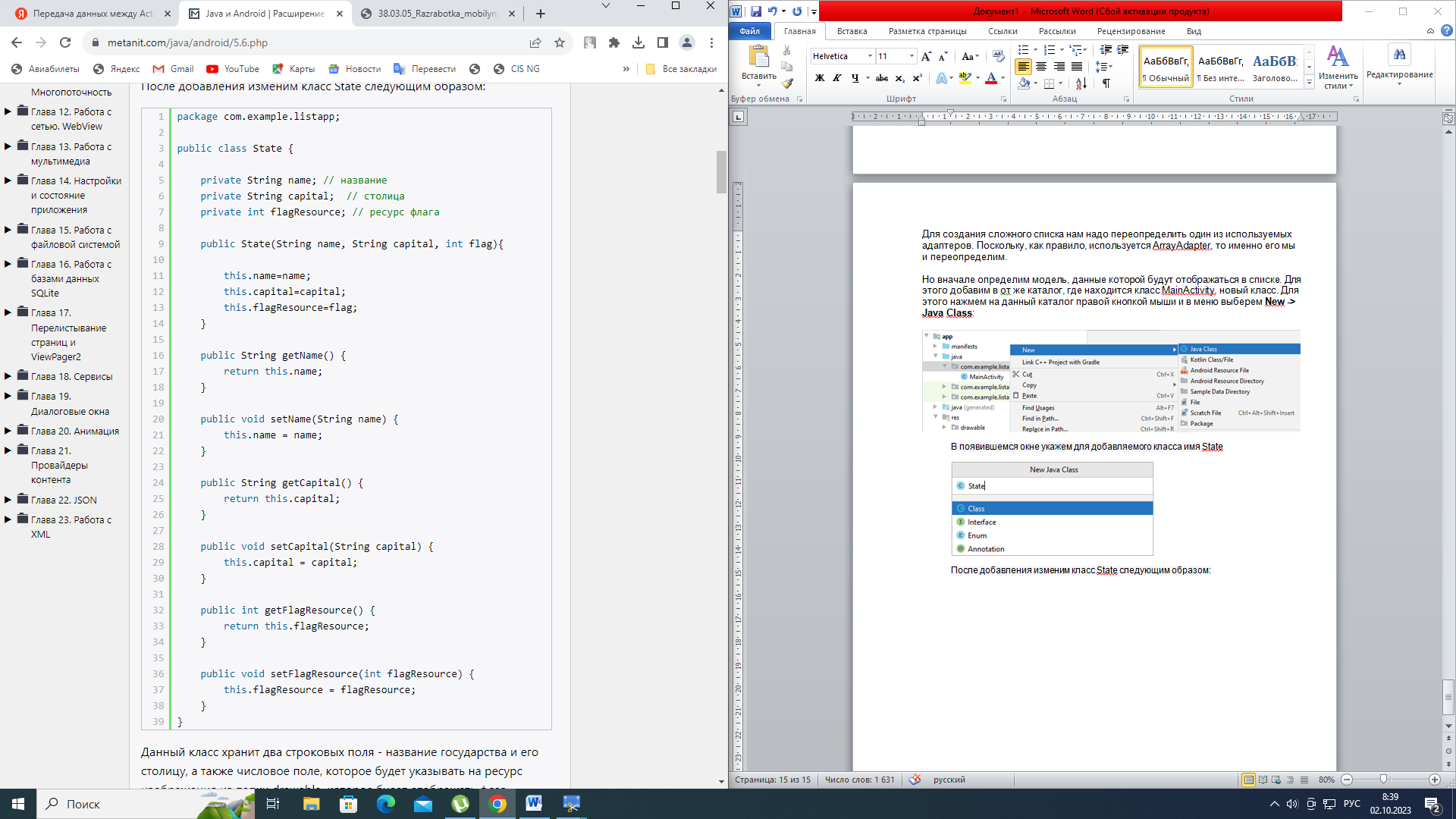
Но вначале определим модель, данные которой будут отображаться в списке. Для этого добавим в от же каталог, где находится класс MainActivity, новый класс. Для этого нажмем на данный каталог правой кнопкой мыши и в меню выберем **New -> Java Class**:



В появившемся окне укажем для добавляемого класса имя State

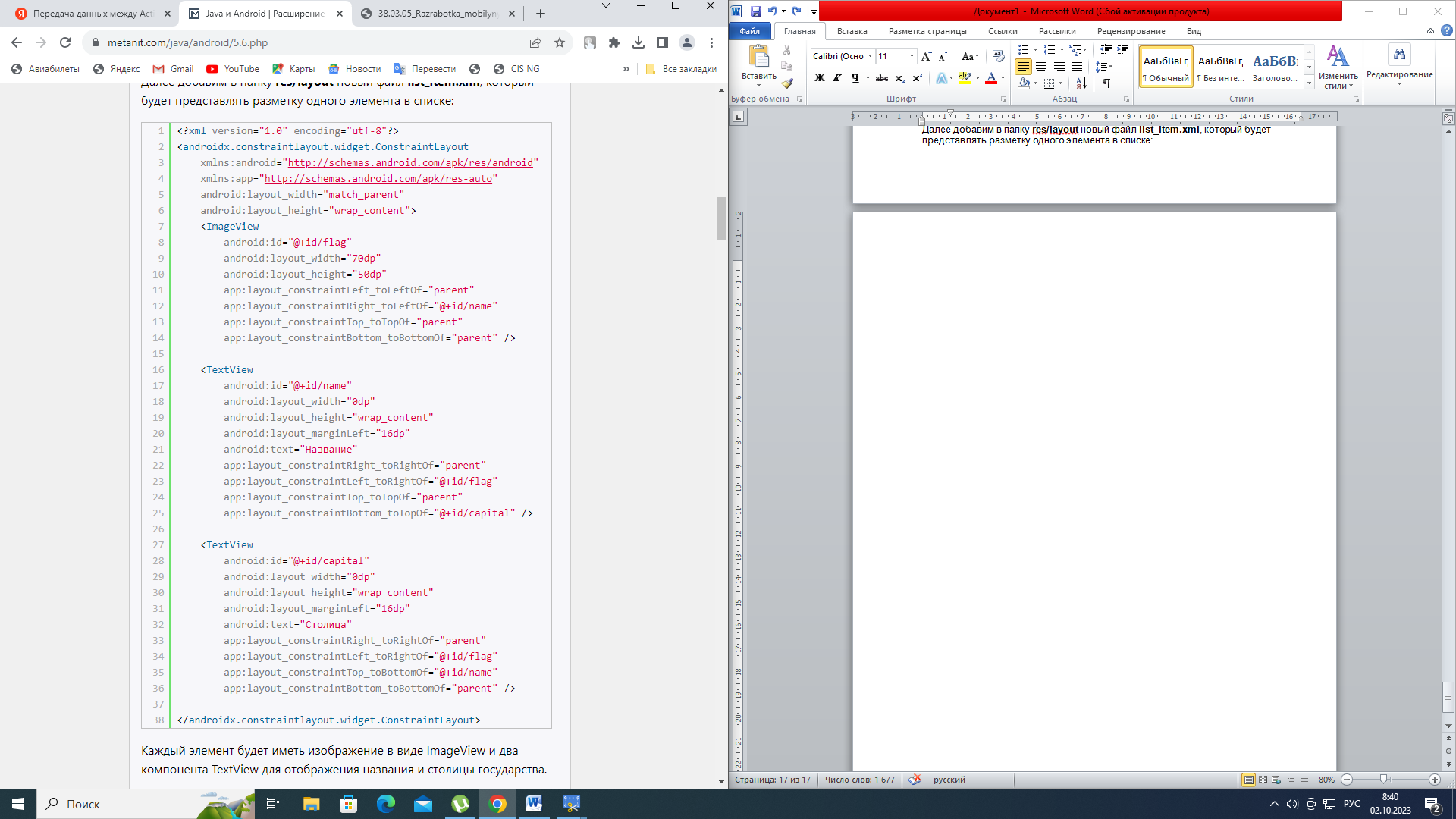


После добавления изменим класс State следующим образом:



Данный класс хранит два строковых поля - название государства и его столицу, а также числовое поле, которое будет указывать на ресурс изображения из папки drawable, которое будет отображать флаг государства.

Далее добавим в папку **res/layout** новый файл **list\_item.xml**, который будет представлять разметку одного элемента в списке:



Каждый элемент будет иметь изображение в виде ImageView и два компонента TextView для отображения названия и столицы государства.

После этого добавим в каталог, где находятся классы MainActivity и State, новый класс, который назовем **StateAdapter**:

package com.example.listapp;

import android.content.Context;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.TextView;

import java.util.List;

public class StateAdapter extends ArrayAdapter<State> {

    private LayoutInflater inflater;

    private int layout;

    private List<State> states;

    public StateAdapter(Context context, int resource, List<State> states) {

        super(context, resource, states);

        this.states = states;

        this.layout = resource;

        this.inflater = LayoutInflater.from(context);

    }

    public View getView(int position, View convertView, ViewGroup parent) {

        View view=inflater.inflate(this.layout, parent, false);

        ImageView flagView = view.findViewById(R.id.flag);

        TextView nameView = view.findViewById(R.id.name);

        TextView capitalView = view.findViewById(R.id.capital);

        State state = states.get(position);

        flagView.setImageResource(state.getFlagResource());

        nameView.setText(state.getName());

        capitalView.setText(state.getCapital());

        return view;

    }

}

Все взаимодействие со списком здесь будет идти через класс StateAdapter. В конструкторе StateAdapter нам надо передать в конструктор базового класса три параметра:

* контекст, в котором используется класс. В его роли кк правило выступает класс Activity
* ресурс разметки интерфейса, который будет использоваться для создания одного элемента в ListView
* набор объектов, которые будут выводиться в ListView

В конструкторе StateAdapter мы получаем ресурс разметки и набор объекто и сохраняем их в отдельные переменные. Кроме того, для создания объекта View по полученному ресурсу разметки потребуется объект LayoutInflater, который также сохраняется в переменную.

В методе getView() устанавливается отображение элемента списка. Данный метод принимает три параметра:

* position: передает позицию элемента внутри адаптера, для которого создается представление
* convertView: старое представление элемента, которое при наличии используется ListView в целях оптимизации
* parent: родительский компонент для представления элемента

В данном случае с помощью объекта LayoutInflater создаем объект View для каждого отдельного элемента в списке:

View view=inflater.inflate(this.layout, parent, false);

Из созданного объекта View получаем элементы ImageView и TextView по id

ImageView flagView = (ImageView) view.findViewById(R.id.flag);

TextView nameView = (TextView) view.findViewById(R.id.name);

TextView capitalView = (TextView) view.findViewById(R.id.capital);

Это те элементы, которые определены в файле list\_item.xml. Здесь же мы их получаем.

Далее используя параметр position, получаем объект State, для которого создается разметка:

State state = states.get(position);

Затем полученные элементы ImageView и TextView наполняем из полученного по позиции объекта State:

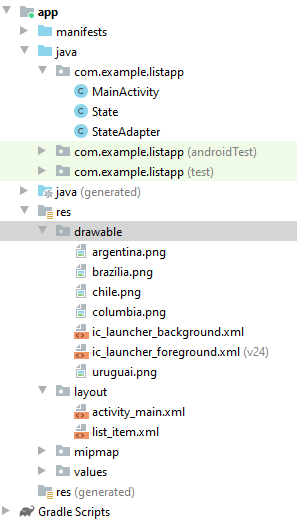
flagView.setImageResource(state.getFlagResource());

nameView.setText(state.getName());

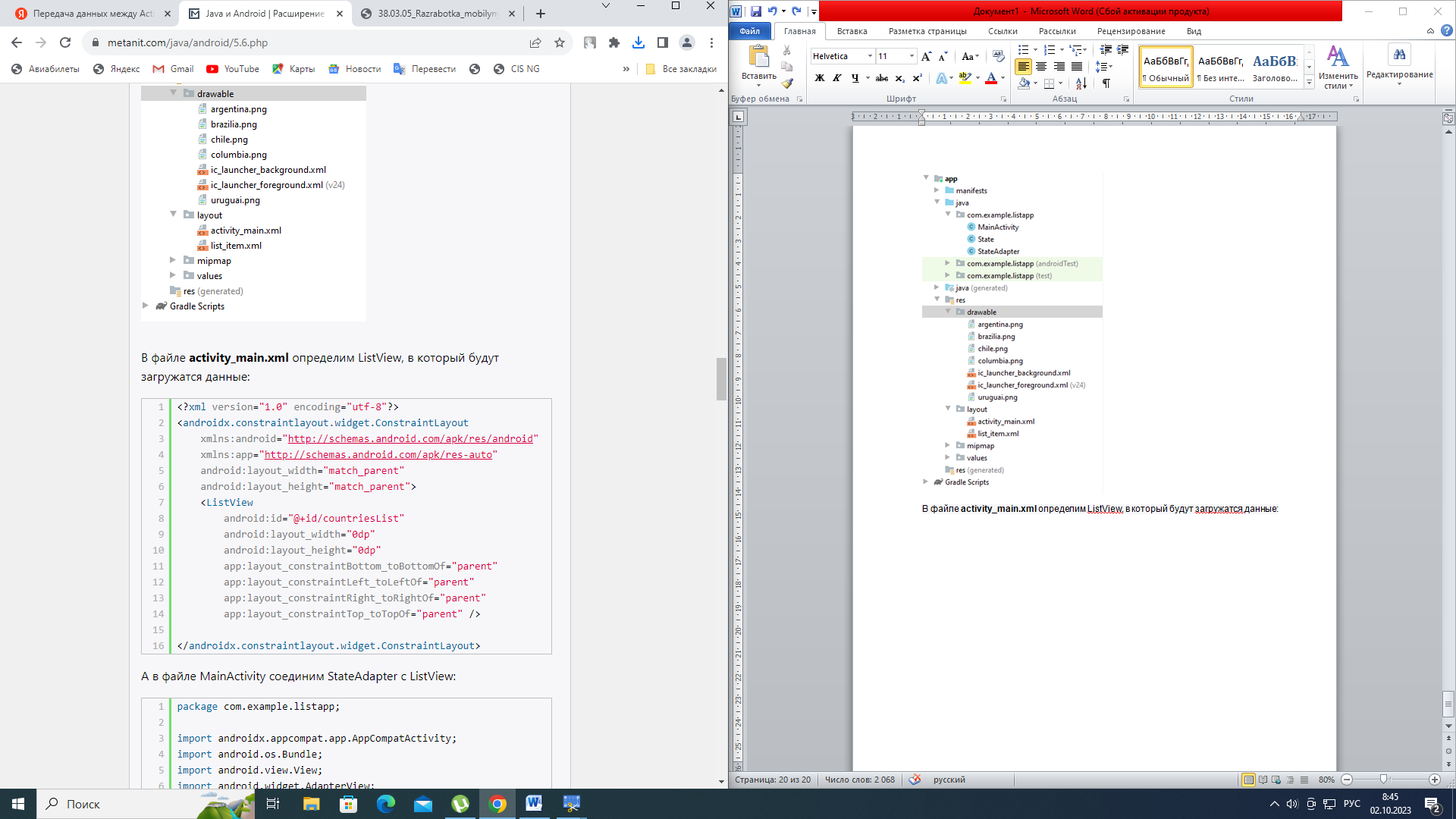
capitalView.setText(state.getCapital());

И в конце созданный для отображения объекта State элемент View возвращается из метода:  
return view;

Для использования изображений добавим в папку *res/drawable* несколько изображений, в моем случае это пять изображений флагов государств. В итоге проект будет выглядеть следующим образом:



В файле **activity\_main.xml** определим ListView, в который будут загружатся данные:



А в файле MainActivity соединим StateAdapter с ListView:

package com.example.listapp;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ListView;

import android.widget.Toast;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    ArrayList<State> states = new ArrayList<State>();

    ListView countriesList;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // начальная инициализация списка

        setInitialData();

        // получаем элемент ListView

        countriesList = findViewById(R.id.countriesList);

        // создаем адаптер

        StateAdapter stateAdapter = new StateAdapter(this, R.layout.list\_item, states);

        // устанавливаем адаптер

        countriesList.setAdapter(stateAdapter);

        // слушатель выбора в списке

        AdapterView.OnItemClickListener itemListener = new AdapterView.OnItemClickListener() {

            @Override

            public void onItemClick(AdapterView<?> parent, View v, int position, long id) {

                // получаем выбранный пункт

                State selectedState = (State)parent.getItemAtPosition(position);

                Toast.makeText(getApplicationContext(), "Был выбран пункт " + selectedState.getName(),

                        Toast.LENGTH\_SHORT).show();

            }

        };

        countriesList.setOnItemClickListener(itemListener);

    }

    private void setInitialData(){

        states.add(new State ("Бразилия", "Бразилиа", R.drawable.brazilia));

        states.add(new State ("Аргентина", "Буэнос-Айрес", R.drawable.argentina));

        states.add(new State ("Колумбия", "Богота", R.drawable.columbia));

        states.add(new State ("Уругвай", "Монтевидео", R.drawable.uruguai));

        states.add(new State ("Чили", "Сантьяго", R.drawable.chile));

    }

}

В качестве источника данных здесь выступает класс ArrayList, который получает данные в методе setInitialData. Каждому добавляемому объекту State в списке передается название государства, его столица и ресурс изображения из папки res/drawable, который представляет флаг государства.

При создании адаптера ему передается ранее созданный ресурс разметки list\_item.xml и список states:

StateAdapter stateAdapter = new StateAdapter(this, R.layout.list\_item, states);

