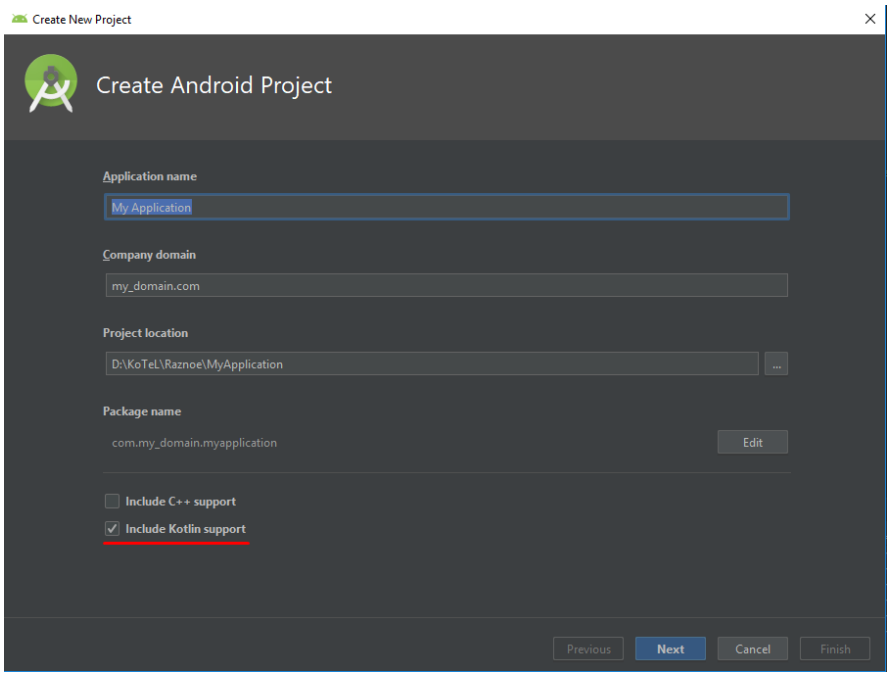
Rss - reader

Для того что бы начать писать приложение на языке Kotlin, вам нужно в самом начале при создании проекта нажать галочку «Include Kotlin support»



После того как чистый проект будет создан, для написания приложения «Заметки», вам нужно подключить следующие библиотеки в glide app

```
implementation 'com.google.code.gson:gson:2.8.5'
implementation 'com.android.support:cardview-v7:28.0.0'
implementation 'com.android.support:recyclerview-v7:28.0.0'
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.11.0'
implementation 'com.github.bumptech.glide:glide:4.7.1'
implementation 'org.sufficientlysecure:html-textview:3.6'
```

Данный список библиотек даёт нам необходимый функционал, а именно, конвертирование json, модули для отображения отдельных статей в списке и так далее, о каждой библиотеке вы можете прочитать подробно по названию, указанному в строке.

И нажать Sync Now в правом верхнем углу чтобы применились изменения в Gradle.

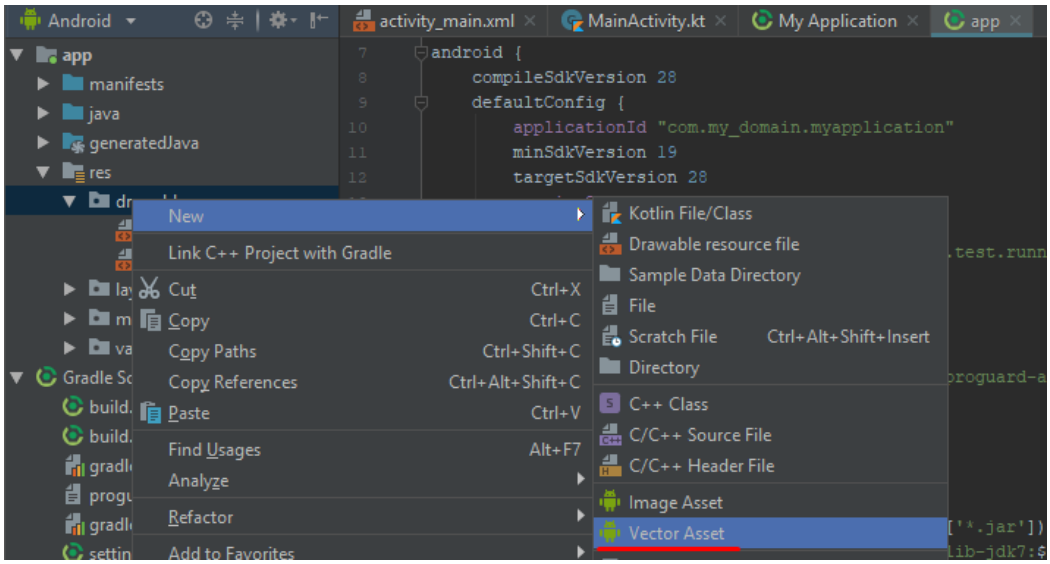
Необходимо добавить в AndroidManifest разрешение нашему приложению на использование интернета, т.к. если мы его не проставим, то приложение не сможет загрузить актуальные новости из интернета. Вообще на бедующее если вам нужно использовать какие-либо ресурсы вне вашего приложения вам необходимо будет проставить разрешения или же ваш функционал просто не будет работать.

```
    <uses-permission android:name="android.permission.INTERNET"/>
```

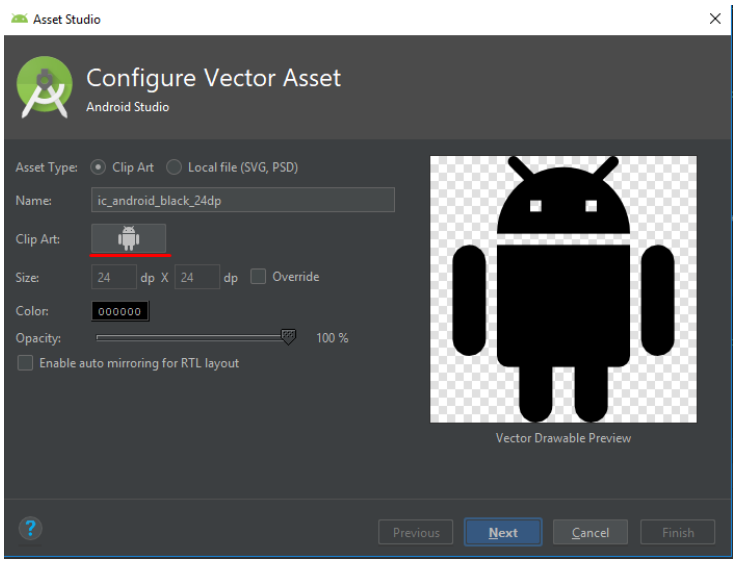
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.my_domain.rss_resder">

    <uses-permission android:name="android.permission.INTERNET"/>
```

Далее добавим иконки которые нам пригодятся в процессе написания приложения.

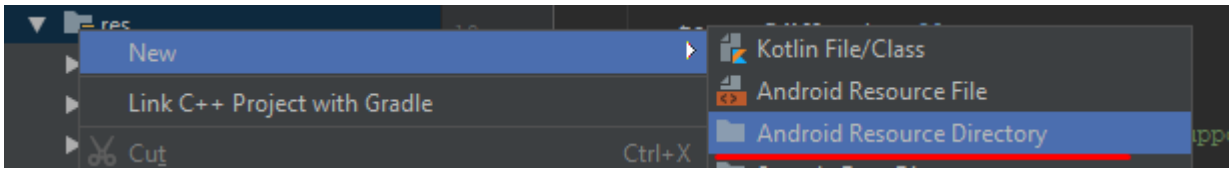


В открывшемся окне нажимаем на иконку Андроида и попадаем в каталог стандартных векторных изображений Android.

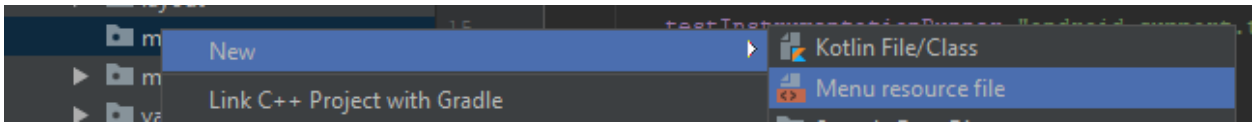


Находим нужную и нажимаем «Ok» -> «Next» -> «Finish». Вам нужно подобным образом поочерёдно добавить иконки с именами «Refresh», «Share». В каталоге присутствует поиск так что найти их не составит труда.

Далее создадим меню нашего приложения. Для этого нажмите правой кнопкой на папку «res» затем «New» и выберите «Android Resource Direcrory»,



Вверху в выпадающем меню «Resource type» выберите «menu» и нажмите «Ok». Создастся папка в которой будут храниться файлы меню. Теперь создадим сам фал нашего меню, для этого нажмите правой кнопкой на «menu» -> «New» -> «Menu resource file»



Назовём его «main_menu» и нажмём «Ok». Напишем туда следующую вёрстку компонентов меню используя картинки которые добавляли ранее.

main_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    >
    <item
        android:id="@+id/menu_refresh"
        android:icon="@drawable/ic_refresh_black_24dp"
        android:title="Refresh"
        app:showAsAction="always"/>

</menu>
```

one_page_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
            android:id="@+id/shareBtn"
            android:icon="@drawable/ic_share"
            android:title="Refresh"
            app:showAsAction="always"/>

</menu>
```

В файле styles.xml находящийся в папке «values» изменим тему приложения на следующую:

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

Создадим файл «layout» в котором мы напишем вёрстку отдельных компонентов нашего списка заметок, правой кнопкой по папке «layout»->«New»-> «Layout resource file» назовём его «row» :

Напишем код вёрстки:

row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:cardView="http://schemas.android.com/apk/res-auto"
        android:padding="8dp"
        android:layout_marginBottom="8dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        cardView:cardCornerRadius="8dp"
        cardView:cardElevation="8dp">

    <LinearLayout
            android:layout_margin="8dp"
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
        <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal"
                android:weightSum="100">
            <LinearLayout
                    android:layout_width="0dp"
                    android:layout_height="match_parent"
                    android:orientation="vertical"
                    android:layout_weight="99">
                <TextView
                        android:id="@+id/textTitle"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:text="Title"
                        android:textColor="@color/colorPrimary"
                        android:textSize="21sp"
                        android:textStyle="bold"/>
                <TextView
                        android:id="@+id/txtPubdate"
                        android:textSize="12sp"
                        android:textStyle="italic"
                        android:text="2017-05-13 12:30:00"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"/>
            </LinearLayout>
            <LinearLayout
                    android:layout_width="wrap_content"
                    android:layout_height="match_parent"
                    android:gravity="end">
                <ImageView
                        android:id="@+id/imageNews"
                        android:layout_width="100dp"
```

```
                        android:layout_height="100dp"/>

            </LinearLayout>
        </LinearLayout>
    </LinearLayout>

</android.support.v7.widget.CardView>
```

Нажав Preview можете посмотреть, как будет выглядеть каждый компонент списка:

Затем создадим Activity которая будет отвечать за просмотр краткого содержания статьи «OneNewsActivity», если вы создадите Activity правильно, то помимо «row.xml» в папке «layout» будет ещё 2 фала «activity_one_news» и «activity_main» это файлы в которых буде вёрстка экранов для просмотра краткого содержания статьи и главного экрана со списком всех статей.

activity_one_news.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        tools:context=".OneNewsActivity">

    <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar_one"
            android:minHeight="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:titleTextColor="@android:color/white"
```

```
    />
    <LinearLayout
            android:layout_margin="8dp"
            android:layout_below="@+id/toolbar_one"
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

        <TextView
                android:id="@+id/oneTextTitle"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Title"
                android:textColor="@color/colorPrimary"
                android:textSize="25sp"
                android:textStyle="bold"/>

        <TextView
                android:id="@+id/oneTextPubdate"
                android:textSize="14sp"
                android:textStyle="italic"
                android:text="2017-05-13 12:30:00"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"/>

        <org.sufficientlysecure.htmltextview.HtmlTextView
                android:id="@+id/html_text"
                android:layout_gravity="bottom"
                android:textSize="21dp"
                android:text="Content"
                android:textAppearance="@android:style/TextAppearance.Medium"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
        />
    </LinearLayout>
</RelativeLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:minHeight="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            app:titleTextColor="@android:color/white"
    />

    <android.support.v7.widget.RecyclerView
            android:id="@+id/recyclerView"
            android:layout_below="@+id/toolbar"
            android:padding="8dp"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
    />
```

```
</RelativeLayout>
```

Теперь создадим мини архитектуру нашего приложения. Создадим папки в которых будут лежать файлы соответственно их назначению, а именно «Adapter», «Common», «Interface», «Model».

Изложу общий принцип работы программы. Мы берём новостные rss данные которые предоставляет сервер, в данном случае это Onliner.by, после чего переводим их в json формат с помощью ресурса rss2json.com, это реализовано в файле «RetrofitServiceGenerator». После того как приходит ответ, данные обрабатываются в адаптере «FeedViewHolder», из всего ответа выбираются нужные нам данные по каждой статье, в чём помогают модели (файлы в папке «Model») в которых прописана вся структура приходящего ответа. Отформатированные данные загружаются в список «RecyclerView» после чего мы получаем новостную ленту, которую предоставил сервер.

Adapter

FeedViewHolder.kt

```
import android.content.Context
import android.content.Intent
import android.support.v7.widget.RecyclerView
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import com.bumptech.glide.Glide
import com.my_domain.rss_reader.Interface.ItemClickListener
import com.my_domain.rss_reader.Model.RSSObject
import com.my_domain.rss_reader.OneNewsActivity
import com.my_domain.rss_reader.R

class FeedViewHolder(itemView: View): RecyclerView.ViewHolder(itemView),
View.OnClickListener, View.OnLongClickListener {

    var txtTitle: TextView
    var txtPubDate: TextView
    var srcImage: ImageView

    private var itemClickListener: ItemClickListener? = null

    init {
        txtTitle = itemView.findViewById(R.id.textTitle) as TextView
        txtPubDate = itemView.findViewById(R.id.txtPubdate) as TextView
        srcImage = itemView.findViewById(R.id.imageNews) as ImageView

        itemView.setOnClickListener(this)
        itemView.setOnLongClickListener(this)
    }

    fun setItemClickListener(itemClickListener: ItemClickListener) {
        this.itemClickListener = itemClickListener
    }
```

```
    override fun onClick(v: View?) {
        itemClickListener!!.onClick(v, adapterPosition, false)
    }

    override fun onLongClick(v: View?): Boolean {
        itemClickListener!!.onClick(v, adapterPosition, true)

        return true
    }
}


class FeedAdapter(private val rssObject: RSSObject, private val mContext:
Context): RecyclerView.Adapter<FeedViewHolder>() {

    private val inflater = LayoutInflater.from(mContext)

    override fun onBindViewHolder(holder: FeedViewHolder, position: Int) {

        holder.txtTitle.text = rssObject.items[position].title
        holder.txtPubDate.text = rssObject.items[position].pubDate

Glide.with(mContext).load(rssObject.items[position].thumbnail).into(holder.srcImag
e)

        holder.setItemClickListener(ItemClickListener { _, pos, isLongClick ->
            if (!isLongClick) {
                val intent = Intent(mContext, OneNewsActivity::class.java)
                intent.putExtra("TITLE", rssObject.items[pos].title)
                intent.putExtra("DATE", rssObject.items[pos].pubDate)
                intent.putExtra("CONTENT", rssObject.items[pos].content)
                intent.putExtra("AUTHOR", rssObject.items[pos].author)
                mContext.startActivity(intent)
            }
        })
    }

    override fun getItemCount(): Int = rssObject.items.size

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
FeedViewHolder {
        val itemView = inflater.inflate(R.layout.row, parent, false)

        return FeedViewHolder(itemView)
    }
}
```

Common

RetrofitServiceGenerator.kt

```
import com.my_domain.rss_reader.BuildConfig
import com.my_domain.rss_reader.Interface.FeedService
import okhttp3.OkHttpClient
import okhttp3.logging.HttpLoggingInterceptor
import retrofit2.Retrofit
import retrofit2.converter.gson.GsonConverterFactory

class RetrofitServiceGenerator {

    companion object {
```

```
        fun createService(): FeedService {
            val apiUrl = "https://api.rss2json.com/v1/"
            val retrofit = Retrofit.Builder()
                .baseUrl(apiUrl)
                .client(builderHttpClient())
                .addConverterFactory(GsonConverterFactory.create())
                .build()

            return retrofit.create(FeedService::class.java)
        }


        private fun builderHttpClient(): OkHttpClient {

            val client = OkHttpClient.Builder()

            if (BuildConfig.DEBUG) {
                val logging = HttpLoggingInterceptor()
                logging.level = HttpLoggingInterceptor.Level.BODY
                client.addInterceptor(logging)
            }

            return client.build()
        }
    }
}
```

Interface

FeedService.kt

```
import com.my_domain.rss_reader.Model.RSSObject
import retrofit2.Call
import retrofit2.http.GET
import retrofit2.http.Query

interface FeedService {

    @GET("api.json")
    fun getFeed(@Query("rss_url") url: String): Call<RSSObject>
}
```

ItemClickListener .java

```
import android.view.View;

public interface ItemClickListener {

    void onClick(View view, int position, boolean isLongClick);

}
```

Model

Feed.kt

```
data class Feed(
    val url: String,
```

```
    val title: String,
    val link: String,
    val author: String,
    val description: String,
    val image: String
)
```

Item.kt

```
data class Item(
    val title: String,
    val pubDate: String,
    val link: String,
    val guid: String,
    val author: String,
    val thumbnail: String,
    val description: String,
    val content: String,
    val enclosure: Object,
    val categories: List<String>
)
```

RSSObject.kt

```
data class RSSObject(
    val status: String,
    val feed: Feed,
    val items: List<Item>
)
```

Activities

MainActivity.kt

```
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.support.v7.widget.LinearLayoutManager
import android.util.Log
import android.view.Menu
import android.view.MenuItem
import com.my_domain.rss_reader.Adapter.FeedAdapter
import com.my_domain.rss_reader.Common.RetrofitServiceGenerator
import com.my_domain.rss_reader.Model.RSSObject
import kotlinx.android.synthetic.main.activity_main.*
import retrofit2.Call
import retrofit2.Callback
import retrofit2.Response

class MainActivity : AppCompatActivity() {

    private val RSS_link = "https://people.onliner.by/feed"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        toolbar.title = "NEWS by Onliner"

        setSupportActionBar(toolbar)
```

```
        val linearLayoutManager = LinearLayoutManager(baseContext,
LinearLayoutManager.VERTICAL, false)
        recyclerView.layoutManager = linearLayoutManager

        loadRSS()
    }

    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.main_menu, menu)

        return true
    }

    override fun onOptionsItemSelected(item: MenuItem?): Boolean {

        if (item?.itemId == R.id.menu_refresh) {
            loadRSS()
        }

        return true
    }

    private fun loadRSS() {

        val call = RetrofitServiceGenerator.createService().getFeed(RSS_link)

        call.enqueue(object : Callback<RSSObject> {

            override fun onFailure(call: Call<RSSObject>?, t: Throwable?) {

                Log.d("ResponseError", "failed")
            }

            override fun onResponse(call: Call<RSSObject>?, response:
Response<RSSObject>?) {

                if (response?.isSuccessful == true) {
                    response.body()?.let { rssObject ->
                        Log.d("Response", "${rssObject.toString()}")
                        val adapter = FeedAdapter(rssObject, baseContext)
                        recyclerView.adapter = adapter
                        adapter.notifyDataSetChanged()
                    }
                }
            }
        })

    }
}
```

OneNewsActivity.kt

```
import android.content.Intent
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.view.Menu
import android.view.MenuItem
import android.view.View
import kotlinx.android.synthetic.main.activity_one_news.*
import org.sufficientlysecure.htmltextview.HtmlHttpImageGetter
import org.sufficientlysecure.htmltextview.HtmlTextView
```

```
class OneNewsActivity : AppCompatActivity() {

    var title = ""
    var date = ""

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_one_news)

        val author  = intent.getStringExtra("AUTHOR")

        this.toolbar_one.title = "Author - \n$author"

        setSupportActionBar(toolbar_one)

        title = intent.getStringExtra("TITLE")
        date = intent.getStringExtra("DATE")
        val content = intent.getStringExtra("CONTENT")


        this.oneTextTitle.text = title
        this.oneTextPubdate.text = date

        val textView = findViewById<View>(R.id.html_text) as HtmlTextView
        textView.setHtml(content, HtmlHttpImageGetter(textView))

    }

    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.one_page_menu, menu)

        return true
    }

    override fun onOptionsItemSelected(item: MenuItem?): Boolean {

        if (item?.itemId == R.id.shareBtn) {
            shareData()
        }

        return true
    }

    private fun shareData() {
        //concatenate
        val s = title + "\n" + date
        //share intent
        val shareIntent = Intent()
        shareIntent.action = Intent.ACTION_SEND
        shareIntent.type = "text/plain"
        shareIntent.putExtra(Intent.EXTRA_TEXT, s)
        startActivity(Intent.createChooser(shareIntent, s))

    }
}
```

Как вы уже могли заметить в коде классов встречаются комментарии, они написаны для вас что бы вам было проще разобраться в коде приложения. Также не забывайте в процессе написания подключать библиотеки («Alt» + «Enter» -> Import), это так же будет подсказывать вам сама студия.

Если вы всё сделали правильно, то у вас получится приложение как на скриншоте.