

Практическая работа 10

Сохранение данных. SharedPreferences

это справочная информация, таблицу можно будет посмотреть только после выполнения данной практической работы

SharedPreferences — постоянное хранилище на платформе Android, используемое приложениями для хранения своих настроек. Для работы с данными постоянного хранилища необходимо создать экземпляр класса (переменную) SharedPreferences. Все данные заносятся в “таблицу” - xml-файл. Для просмотра таблицы раскройте Device File Explorer в Android Studio. Раскройте папку data и в ней вновь откройте папку data, затем найдите папку с названием вашего пакета приложения, в ней выберите папку shared_prefs и откройте xml - файл (если данные не были сохранены, то он окажется пустым, если данные сохранялись - будет содержать информацию). В данной работе имя таблицы TABLEE, имя может быть любым

Device File Explorer

Emulator Pixel_3a_API_33_x86_64 Android 13, API 33

Name	Permis...	Date	Size
> acct	drwxr-xr->	2022-07-07 20:4	4 KB
> apex	drwxr-xr->	2023-01-29 20:3	1,2 KB
> bin	lrw-r--r--	2022-07-07 20:4	11 B
> cache	drwxrwx--	2022-07-07 20:4	4 KB
> config	drwxr-xr->	2023-01-29 20:3	0 B
> d	lrw-r--r--	2022-07-07 20:4	17 B
▼ data	drwxrwx--	2023-01-29 20:3	4 KB
> adb	drwx-----	2022-10-22 05:2	4 KB
> anr	drwxrwxr-	2022-10-22 05:2	4 KB
> apex	drwxr-xr->	2022-10-22 05:2	4 KB
> app	drwxrwx--	2023-01-30 19:1	4 KB
> app-asec	drwx-----	2022-10-22 05:2	4 KB
> app-ephemeral	drwxrwx--	2022-10-22 05:2	4 KB
> app-lib	drwxrwx--	2022-10-22 05:2	4 KB
> app-private	drwxrwx--	2022-10-22 05:2	4 KB
> app-staging	drwxr-x->	2022-10-22 05:2	4 KB
> backup	drwx-----	2023-01-30 19:1	4 KB
> bootanim	drwxr-xr->	2022-10-22 05:2	4 KB
> bootchart	drwxr-xr->	2022-10-22 05:2	4 KB
> cache	drwxrwx--	2022-10-22 05:2	4 KB
> dalvik-cache	drwxrwx--	2022-10-22 05:2	4 KB
▼ data	drwxrwx--	2023-01-30 18:3	12 KB
> android	drwx-----	2022-10-22 05:2	4 KB
> android.auto_generat	drwx-----	2022-10-22 05:2	4 KB
> android.auto_generat	drwx-----	2022-10-22 05:2	4 KB
> com.android.backup	drwx-----	2022-10-22 05:2	4 KB
> com.android.bips	drwx-----	2022-10-22 05:2	4 KB
> com.android.bips.aut	drwx-----	2022-10-22 05:2	4 KB

AndroidManifest.xml

java

com.example.testproject

MainActivity3.kt

TABLEE.xml

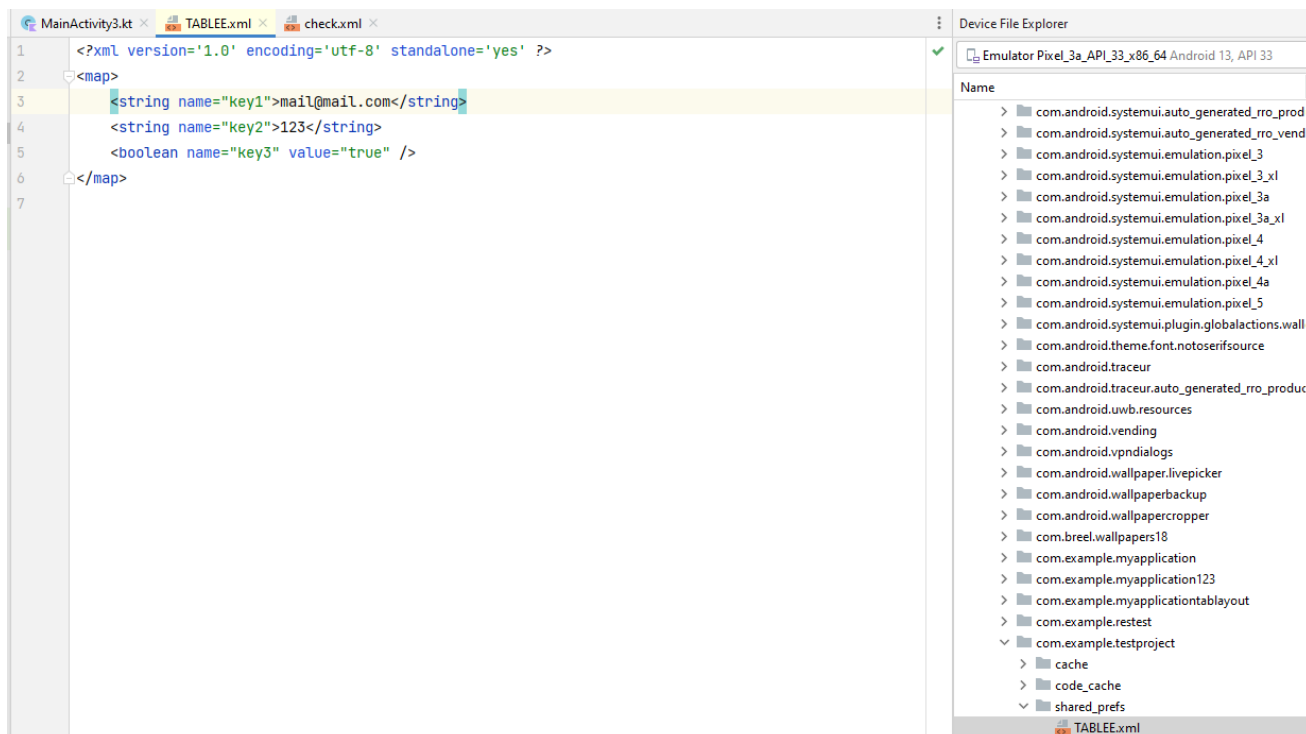
checkxml

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map />
```

Device File Explorer

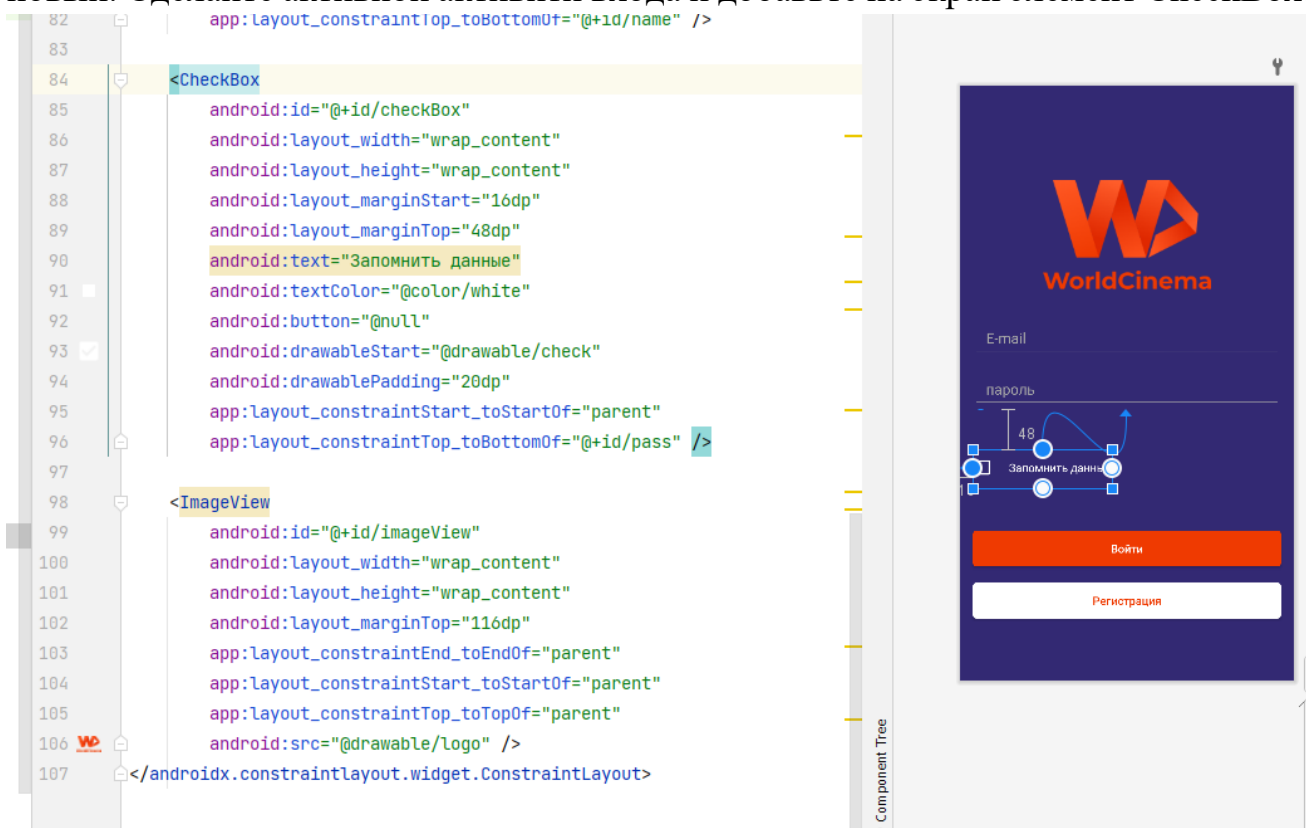
Emulator Pixel_3a_API_33_x86_64 Android 13, API 33

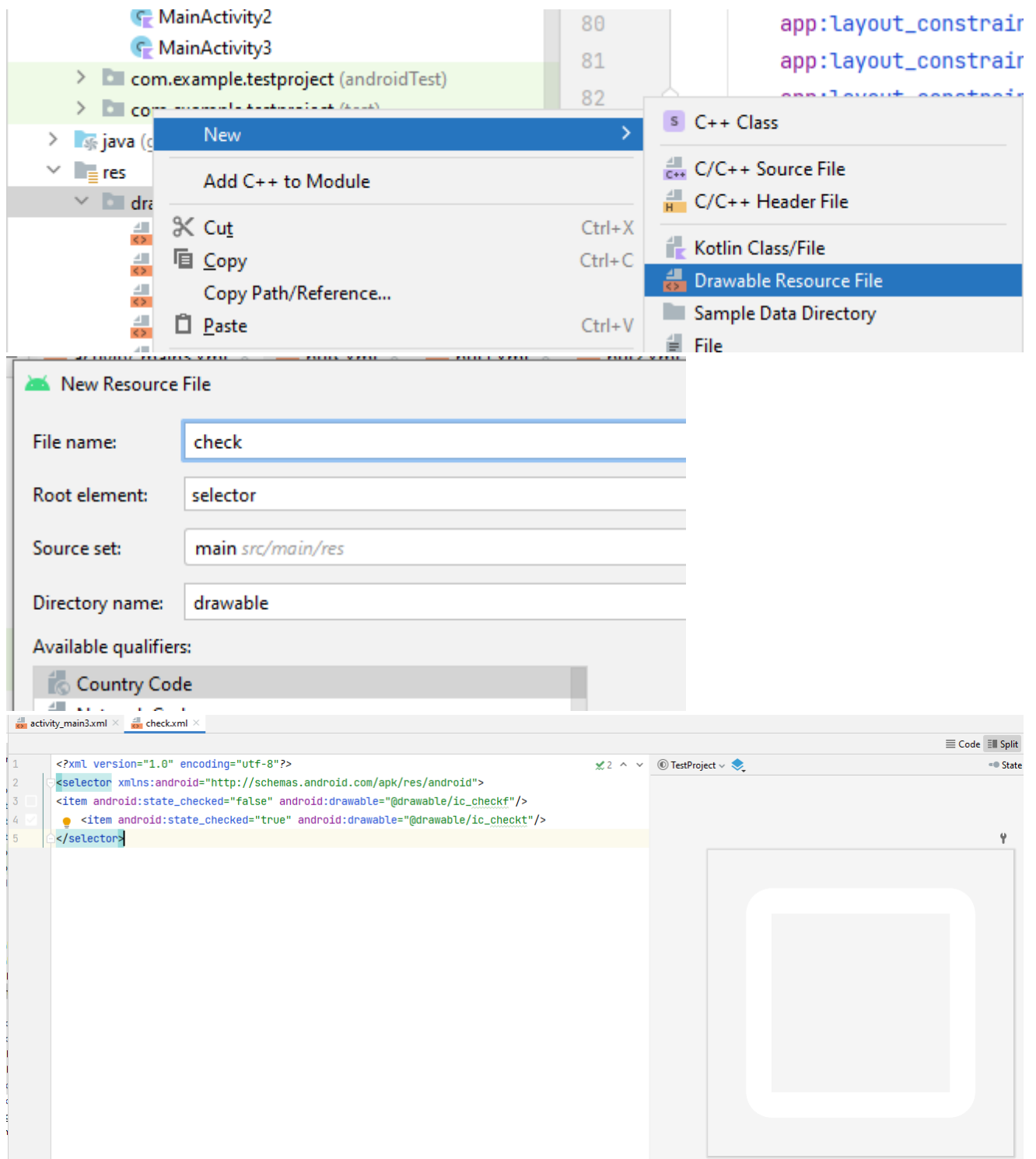
Name	Permissions	Date	Size
> com.android.systemui.auto_generated_ro_produc	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.auto_generated_ro_yend	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_3_xl	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_3a_xl	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_3a_xl	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_4_xl	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_4_xl	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_4a	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.emulation.pixel_5	drwx-----	2022-10-22 05:26	4 KB
> com.android.systemui.plugin.globalsactions.wall	drwx-----	2022-10-22 05:26	4 KB
> com.android.theme.font.notoserifsource	drwx-----	2022-10-22 05:26	4 KB
> com.android.traceur	drwx-----	2022-11-14 17:21	4 KB
> com.android.traceur.auto_generated_ro_produc	drwx-----	2022-10-22 05:26	4 KB
> com.android.uwb.resources	drwx-----	2022-10-22 05:26	4 KB
> com.android.vending	drwx-----	2022-10-22 05:26	4 KB
> com.android.vpndialogs	drwx-----	2022-10-22 05:26	4 KB
> com.android.wallpaper.livepicker	drwx-----	2022-10-22 05:26	4 KB
> com.android.wallpaperbackup	drwx-----	2022-10-22 05:26	4 KB
> com.android.wallpapercropper	drwx-----	2022-10-22 05:26	4 KB
> com.breel.wallpapers18	drwx-----	2022-10-22 05:26	4 KB
> com.example.myapplication	drwx-----	2022-12-13 18:22	4 KB
> com.example.myapplication123	drwx-----	2022-11-21 19:50	4 KB
> com.example.myapplicationtablayout	drwx-----	2023-01-22 14:52	4 KB
> com.example.rested	drwx-----	2023-01-28 05:25	4 KB
▼ com.example.testproject	drwx-----	2023-01-30 18:38	4 KB
> cache	drwxrws--x	2023-01-30 18:38	4 KB
> code_cache	drwxrws--x	2023-01-30 19:14	4 KB
▼ shared_prefs	drwxrws--x	2023-01-30 19:14	4 KB
TABLEE.xml	-rw-rw----	2023-01-30 19:14	140 B



Сохранение почты и пароля пользователя

Продолжите работу над проектом Сinема, если он отсутствует - создайте новый. Сделайте активной активити входа и добавьте на экран элемент CheckBox:



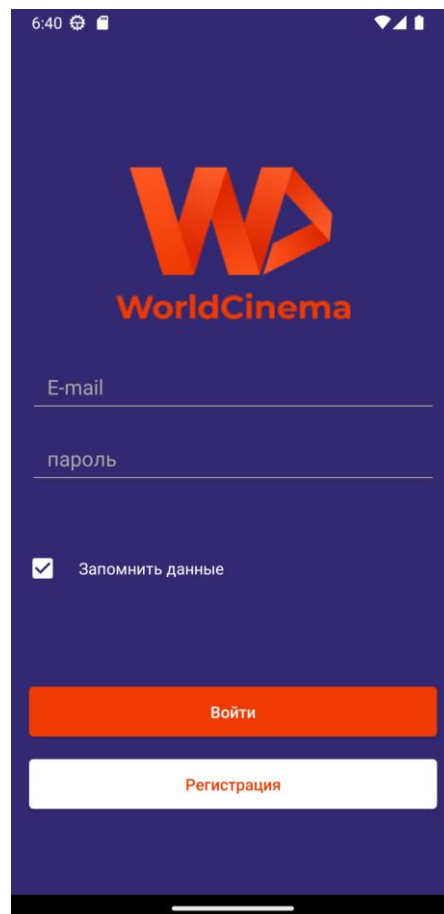
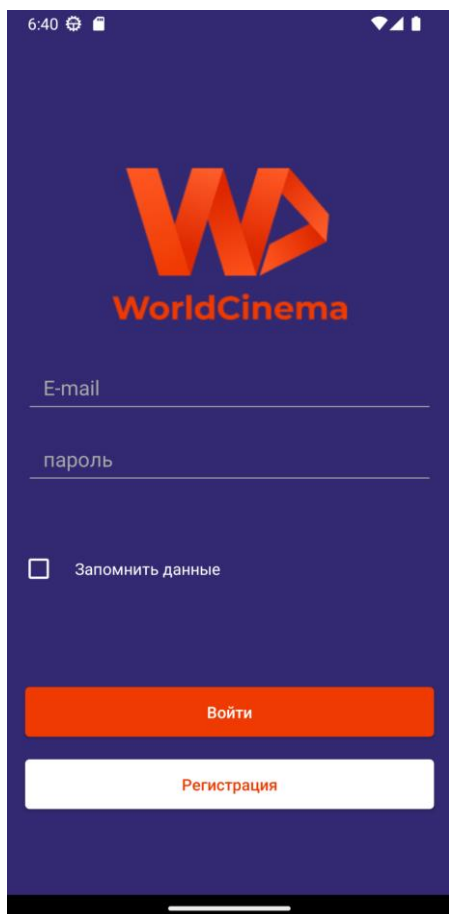


92 строка - отключение рамки чекбокса по умолчанию

93 строка - назначение пользовательских изображений для состояний чекбокс нажат и не нажат, подключение файла drawable selector

При создании файла-селектора назначаются пользовательские иконки для двух состояний чекбокса: checked - true нажат и checked - false не нажат.

Запустите приложение:



Настройте сохранение (запоминание) введенных пользователем почты, пароля и состояния чекбокса (нажат или нет) при нажатии на кнопку **Войти**, но только в том случае, если будет установлена галочка “Запомнить данные”.

Откройте файл настройки логики экрана и дополните код следующими строками (внимательно прочитайте пояснения к строкам кода):

```
1 package com.example.testproject
2
3 import android.content.Intent
4 import android.content.SharedPreferences
5 import androidx.appcompat.app.AppCompatActivity
6 import android.os.Bundle
7 import android.view.View
8 import android.widget.CheckBox
9 import android.widget.EditText
10 import android.widget.TextView
11
12 class MainActivity3 : AppCompatActivity() {
13
14     var preff: SharedPreferences?=null
15     lateinit var email: EditText
16     lateinit var password:EditText
17     lateinit var check:CheckBox
18     override fun onCreate(savedInstanceState: Bundle?) {
19         super.onCreate(savedInstanceState)
20         setContentView(R.layout.activity_main3)
21         email=findViewById(R.id.name)
22         password=findViewById(R.id.pass)
23         check=findViewById(R.id.checkBox)
24         preff=getSharedPreferences( name: "TABLEE", MODE_PRIVATE)
25         check.isChecked=preff?.getBoolean( key: "key3", defValue: false)?:false
26         email.setText(preff?.getString( key: "key1", defValue: ""))
27         password.setText(preff?.getString( key: "key2", defValue: ""))
28     }
29     fun savestate(check:Boolean)
30     {
31         val editor=preff?.edit()
32         editor?.putBoolean("key3", check)
33         editor?.apply()
34     }
```

14 строка - для работы с данными постоянного хранилища необходимо создать экземпляр класса (переменную) SharedPreferences и присвоить начальное значение **null**

ключевое слово **null** представляет специальный литерал, который указывает, что **переменная не имеет** как такового значения. То есть у нее по сути **отсутствует значение**. Подобное значение может быть полезно в ряде ситуациях, когда необходимо использовать данные, но при этом точно неизвестно, а есть ли в реальности эти данные. Например, мы получаем данные по сети, данные могут прийти или не прийти. Либо может быть ситуация, когда нам надо явным образом указать, что данные не установлены

24 строка - для инициализации переменной `preff` используется метод `getSharedPreferences`. В скобках указаны 2 параметра: **name** — **название** файла настроек (название таблицы, где будут сохраняться под ключевым словом данные, если такой таблицы еще не существует, она будет создана автоматически при вызове метода `edit()`) и **mode** — **режим работы** - `MODE_PRIVATE` (доступ к настройкам будет только у этого приложения). Все другие режимы работы являются устаревшими и не используются

25 строка - для чекбокса присваивается значение состояния: нажат или нет. Значение состояния чекбокса считывается из памяти (истина - нажат, ложь - не нажат). Значение записано в таблицу (см.24 строку) под ключом “key3”. Если ничего не было записано - значение по умолчанию установлено `false`.

?: – оператор "элвис"

Оператор, обозначаемый вопросительным знаком с двоеточием (Elvis operator), подобен проверке на null в варианте if-else. Он возвращает значение слева от себя, если оно не null.

И возвращает значение справа от себя, если то, что слева, – null.

Другими словами, после elvis-оператора находится значение по-умолчанию, которое возвращается только в том случае, если выражение до ?: вернуло null.

26 и 27 строки - считываем значения текстовых полей, ранее занесенные в таблицу под ключевыми слова `key1` (почта) и `key2` (пароль) и устанавливаем их в качестве значения для параметра `text` (заполняем ими текстовое поле), если в таблицу ничего не было записано - текстовые поля останутся пустыми и в них будут отображаться подсказки - `hint`

29-34 строки - прописываем функцию записи состояния чекбокса (нажат или нет) и сохраняем значение под ключом “key 3” в таблице

35-40 строки - прописываем функцию сохранения введенных почты (ключ “key1”) и пароля (ключ “key2”) в таблицу. Необходимо сохранить два значения, поэтому у функции при ее объявлении указывается два параметра. Тип параметра соответствует значению, которое необходимо сохранить

Через параметры в функцию можно передать различные значения.

Параметры перечисляются в скобках после имени функции. При вызове функции для этих параметров необходимо передать значения. Значения, передаваемые параметрам функции при ее вызове, называются **аргументами**. Аргументы передаются по позиции, то есть первый аргумент передается первому

параметру, второй аргумент - второму параметру и так далее. При этом аргументы должны соответствовать параметрам по типу

41-46 строки - прописываем функцию удаления всех записанных данных из таблицы

47-61 строки - прописываем функцию, срабатывающую при нажатии на кнопку “Войти”:

48 и 49 строки - объявляем переменные и присваиваем им значения параметра text текстовых полей

50 строка - логической переменной присваиваем состояние чекбокса (isChecked может быть в двух состояниях - ложь (не нажат) и истина (нажат))

51 строка - если чекбокс был нажат, вызвать (52 строка) и выполнить функцию сохранения данных в память (в качестве аргументов указать переменные, значения которых нужно сохранить) и функцию сохранения состояния чекбокса (53 строка)

55 строка - иначе, если значение false, вызываем функцию очистки таблицы (57 строка)

59 строка - переходим на следующий экран (имя активити прописываете в соответствии с вашим проектом), переход будет осуществлен независимо от проверки и выполнения условий операторов if-else (обратите внимание, он прописан ниже всех проверок)

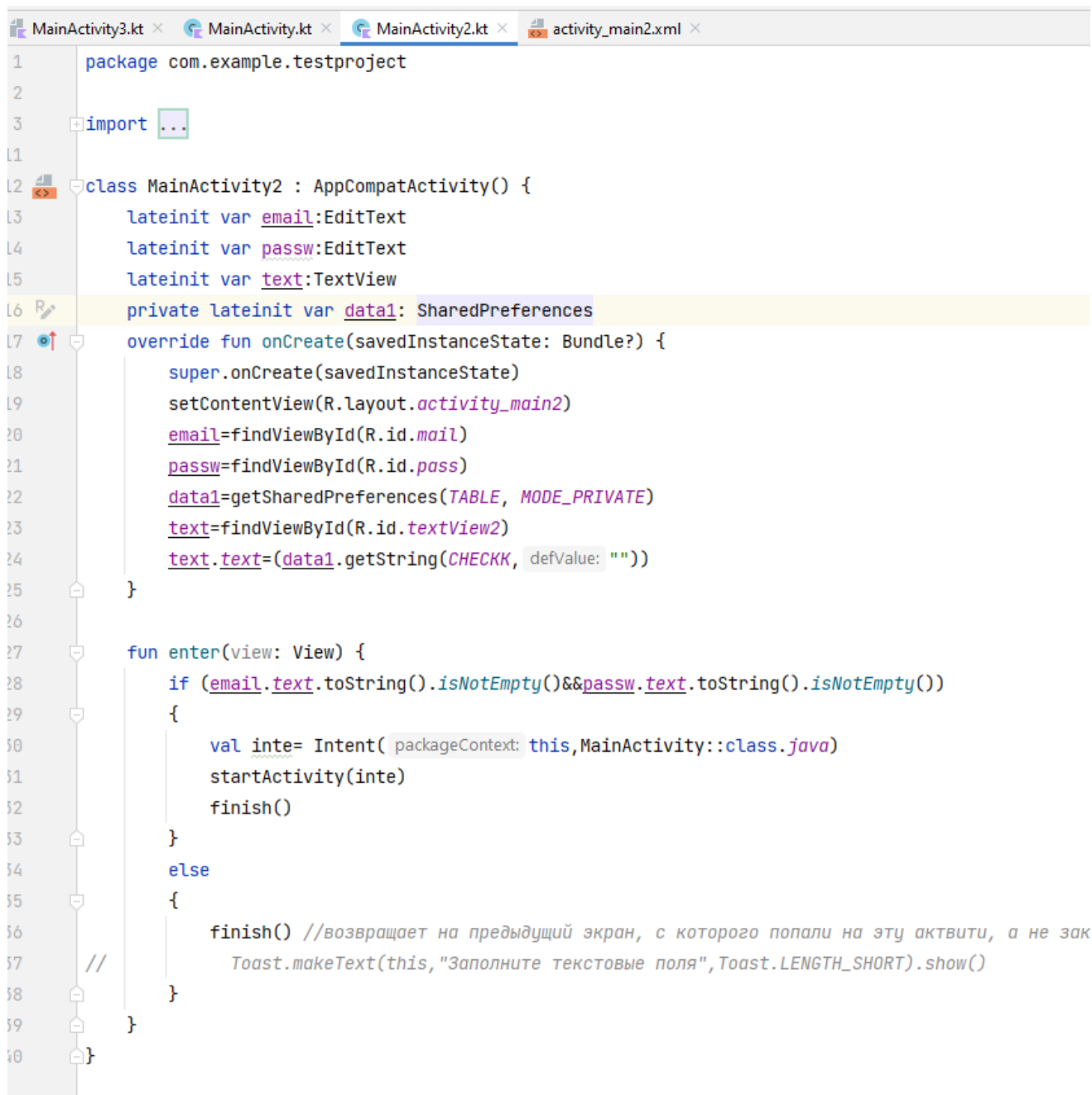
*ключа объявить константами вне класса MainActivity : AppCompatActivity()
(см.11 и 12 строки кода на изображении ниже). Таким образом у вас будет
возможность обращаться к ним из разных активити*

Переключитесь на экран Регистрации (3 экран вашего приложения). Вам необходимо реализовать следующий функционал: пользователь запускает первый раз приложение, он не зарегистрирован. Нажимает на кнопку Регистрация и попадает на соответствующий экран. Заполняет текстовые поля, нажимает на кнопку Зарегистрироваться и попадает на экран Вход. Вводит свою почту и пароль и если пароль не совпадает с тем, что был введен на экране регистрации ему высвечивается сообщение об ошибке. Если пользователь введет другую почту, а не ту, что указал при регистрации, ему также должно быть показано сообщение об ошибке (через всплывающее или диалоговое окно реализовать сообщения системы - решайте самостоятельно). У пользователя должна быть возможность сохранить данные, указанные на экране входа. При нажатии на кнопку Войти пользователь должен попасть на экран своего профиля (он был создан в ранее выполненной практической работе). На экране Профиль пользователя должно отображаться имя пользователя, которое он указал при регистрации в приложении. Часть реализации функционала будет показана далее, **дописать** недостающий код будет вашей **4 контрольной работой**. Логике экрана “Вход” необходимо переделать, иначе будет ошибка в работе приложения.

Проанализируйте код, представленный ниже. Обратите внимание, сохранение данных происходит на одной активити, а обращение на другой.

```
MainActivity3.kt x MainActivity.kt x MainActivity2.kt x activity_main2.xml x
4      import android.content.SharedPreferences
5      import androidx.appcompat.app.AppCompatActivity
6      import android.os.Bundle
7      import android.view.View
8      import android.widget.CheckBox
9      import android.widget.EditText
10     import android.widget.TextView
11     const val TABLE="TABLE"
12     const val CHECKK="KEY4"
13     class MainActivity3 : AppCompatActivity() {
14
15         var preff: SharedPreferences?=null
16         lateinit var email: EditText
17         lateinit var password:EditText
18         lateinit var check:CheckBox
19         override fun onCreate(savedInstanceState: Bundle?) {
20             super.onCreate(savedInstanceState)
21             setContentView(R.layout.activity_main3)
22             email=findViewById(R.id.name)
23             password=findViewById(R.id.pass)
24             check=findViewById(R.id.checkBox)
25             preff=getSharedPreferences(TABLE, MODE_PRIVATE)
26             check.isChecked=preff?.getBoolean( key: "key3", defValue: false)?:false
27             email.setText(preff?.getString(CHECKK, defValue: ""))
28             password.setText(preff?.getString( key: "key2", defValue: ""))
29         }
30         fun savestate(check:Boolean)
31         {
32             val editor=preff?.edit()
33             editor?.putBoolean("key3", check)
34             editor?.apply()
35         }
36         fun saveData(mail:String,pass:String){
37             val editor=preff?.edit()
38             editor?.putString(CHECKK, mail)
39             editor?.putString("key2", pass)
40             editor?.apply()

```



```

1 package com.example.testproject
2
3 import ...
4
5 class MainActivity2 : AppCompatActivity() {
6     lateinit var email: EditText
7     lateinit var passw: EditText
8     lateinit var text: TextView
9     private lateinit var data1: SharedPreferences
10
11     override fun onCreate(savedInstanceState: Bundle?) {
12         super.onCreate(savedInstanceState)
13         setContentView(R.layout.activity_main2)
14         email=findViewById(R.id.mail)
15         passw=findViewById(R.id.pass)
16         data1=getSharedPreferences(TABLE, MODE_PRIVATE)
17         text=findViewById(R.id.textView2)
18         text.text=(data1.getString(CHECKK, defValue: ""))
19     }
20
21     fun enter(view: View) {
22         if (email.text.toString().isEmpty() && passw.text.toString().isEmpty())
23         {
24             val inte= Intent( packageContext: this, MainActivity::class.java)
25             startActivity(inte)
26             finish()
27         }
28         else
29         {
30             finish() //возвращает на предыдущий экран, с которого попали на эту активити, а не за
31             // Toast.makeText(this, "Заполните текстовые поля", Toast.LENGTH_SHORT).show()
32         }
33     }
34 }

```

Если вам необходимо присвоить значение из таблицы данных переменной, используйте следующую конструкцию:

```
var conter=0
```

```
conter=pref?.getInt( key: "key1", defValue: 0)!!
```

Переменная должна быть такого же типа, как и данные, указанного ключа.
Для проверки условий, используйте if...else