

JEGYZŐKÖNYV

Mobil Programozás

Féléves feladat

Cipő webshop elkészítése

Készítette: **Fekete Máté**

Neptunkód: **JR9KY7**

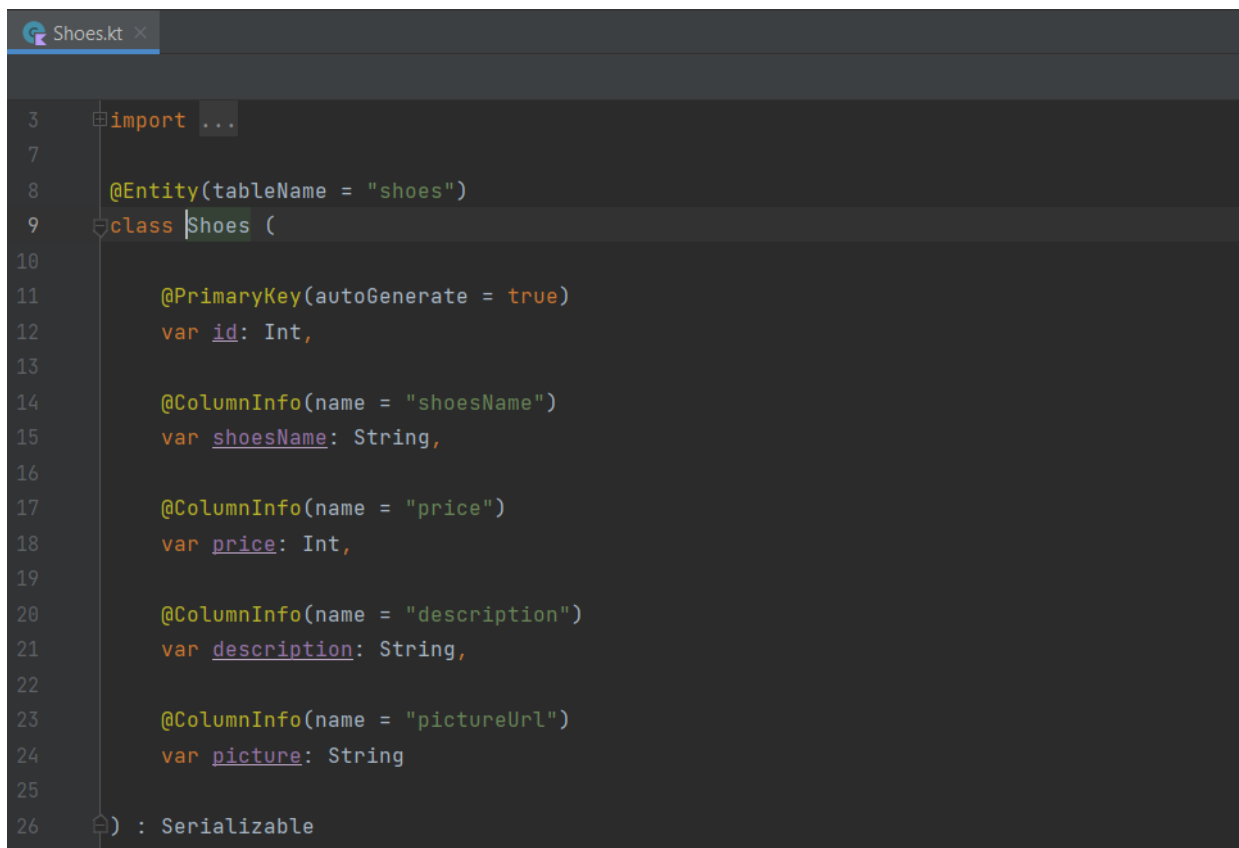
A FELADAT ISMERTETÉSE:

A feladat egy mobil alkalmazás elkészítése Kotlin-t alkalmazva, a megadott feltételek kielégítésével. A teljesítéshez szükség van egy bejelentkező, illetve egy regisztrációs felületre, és a bejelentkezés után, létre kell hozni valamilyen termékek nyilvántartására szolgáló kezelőfelületet. A feladat teljesítéséhez szükséges pontok között szerepel még, hogy szükséges egy dokumentáció, melyben be kell mutatni az elkészült alkalmazás felhasználói felületét és a megírt kódot. A feladat megoldását a backend kialakításával kezdtem, majd a frontend megalkotásával fejeztem be.

A KÓD RÉSZLETEZÉSE ÉS A FELHASZNÁLÓI FELÜLET:

Backend:

Elsősorban létrehozam a megfelelő fájlszerkezetet és kezdeti identitásokat. Ezeknek fogom később az adatait felhasználni. A helyes működéshez elengedhetetlen a DAO fájlok létrehozása, ami a front- és backend közötti kommunikációt segíti. Ezek a fájlok betekintést nyújtanak az adatstruktúrába.



```
3  import ...
7
8  @Entity(tableName = "shoes")
9  class Shoes (
10
11      @PrimaryKey(autoGenerate = true)
12      var id: Int,
13
14      @ColumnInfo(name = "shoesName")
15      var shoesName: String,
16
17      @ColumnInfo(name = "price")
18      var price: Int,
19
20      @ColumnInfo(name = "description")
21      var description: String,
22
23      @ColumnInfo(name = "pictureUrl")
24      var picture: String
25
26  ) : Serializable
```

Az adapterek létrehozásával folytattam, amelyek a fentebb szereplő “entitások” kezelőmoduljai. Feladatuk az identitásokkal történő alapvető, illetve táblaspecifikus műveletek kezelése. Az ilyen műveletek közé tartozik a listázás, számolás, módosítás, létrehozás, stb.

```
ShoesAdapter.kt
21
22 class ShoesAdapter : RecyclerView.Adapter<ShoesAdapter.RecipeViewHolder>() {
23
24     var listener: OnItemClickListener? = null
25     var ctx: Context? = null
26     var arrShoes = ArrayList<Shoes>()
27
28     class RecipeViewHolder(view: View): RecyclerView.ViewHolder(view){
29
30     }
31
32     fun setData(arrData : List<Shoes>){
33         arrShoes = arrData as ArrayList<Shoes>
34     }
35
36     fun setClickListener(listener1: OnItemClickListener){
37         listener = listener1
38     }
39
40     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): RecipeViewHolder {
41         ctx = parent.context
42         return RecipeViewHolder(LayoutInflater.from(parent.context).inflate(R.layout.cardelement_layout,parent, attachToRoot false))
43     }
}
```

Frontend:

Amint készen lett a backend rész, folytattam a frontend-el. Ez már látványosabb eredményeket hozott. Ez lesz az a felület, amit a felhasználó lát és kezelni tud. Aktivitásokat hoztam létre oldalanként. Itt a komponensek műveleteit írtam meg, melyeket a felhasználó fog majd tudni használni, ezekkel kezelheti az appot. Szükségem volt több importra is és később elvégeztem a kellő példányosításokat és inicializáltam a változókat.

```

FirstScreenActivity.kt
1  package com.example.myapplication
2
3  import ...
13
14  class FirstScreenActivity : AppCompatActivity() , View.OnClickListener {
15
16      var arrShoes = ArrayList<Shoes>()
17      var shoesAdapter = ShoesAdapter()
18
19      override fun onCreate(savedInstanceState: Bundle?) {
20          super.onCreate(savedInstanceState)
21          setContentView(R.layout.main_layout)
22
23          arrShoes.add(Shoes( id: 1, shoesName: "Nike Dunk Low", price: 38000, description: "Leírás_1!", picture: "https://www
24          arrShoes.add(Shoes( id: 2, shoesName: "Adidas Samba", price: 40000, description: "Leírás_2", picture: "https://stati
25          shoesAdapter.setData(arrShoes)
26
27          val recyclerView = findViewById<RecyclerView>(R.id.shoesCardElement)
28          recyclerView.adapter = shoesAdapter
29          recyclerView.layoutManager = LinearLayoutManager( context: this)
30

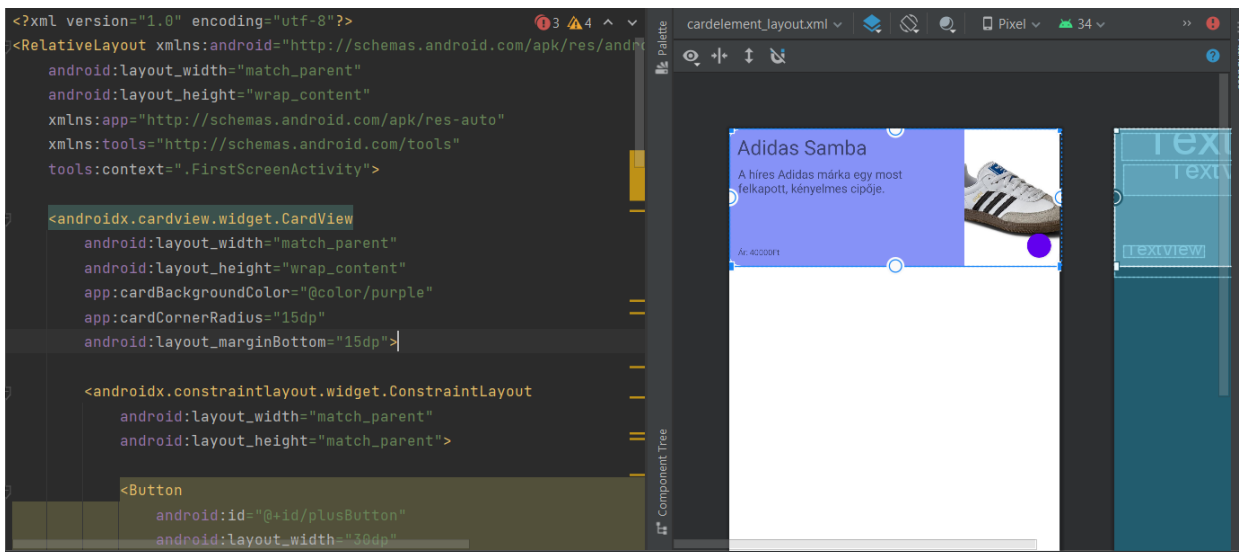
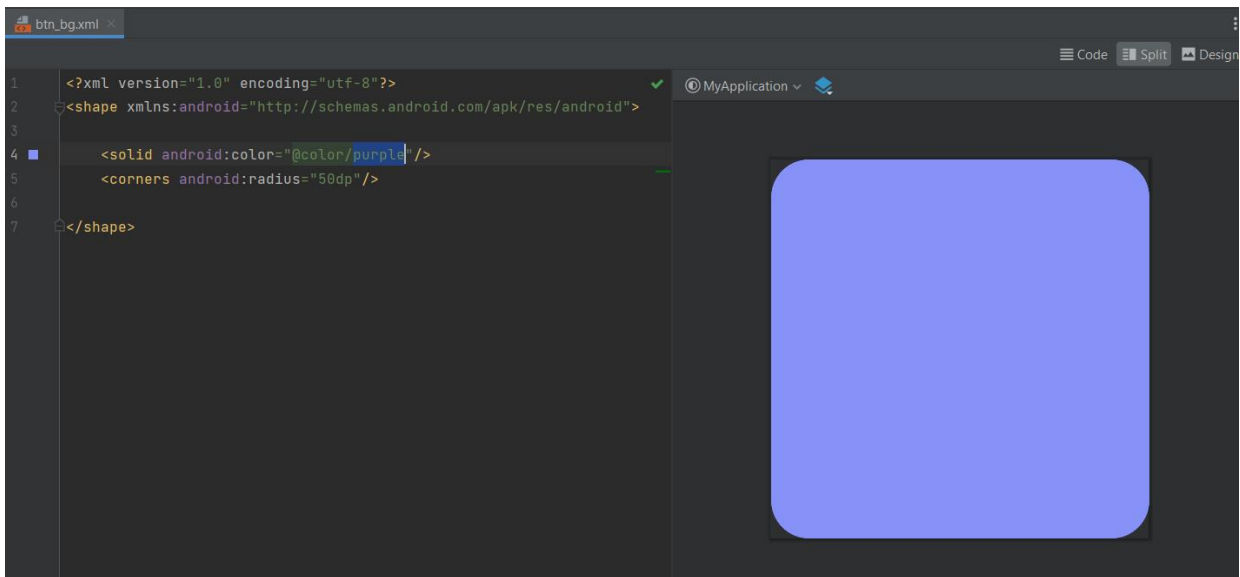
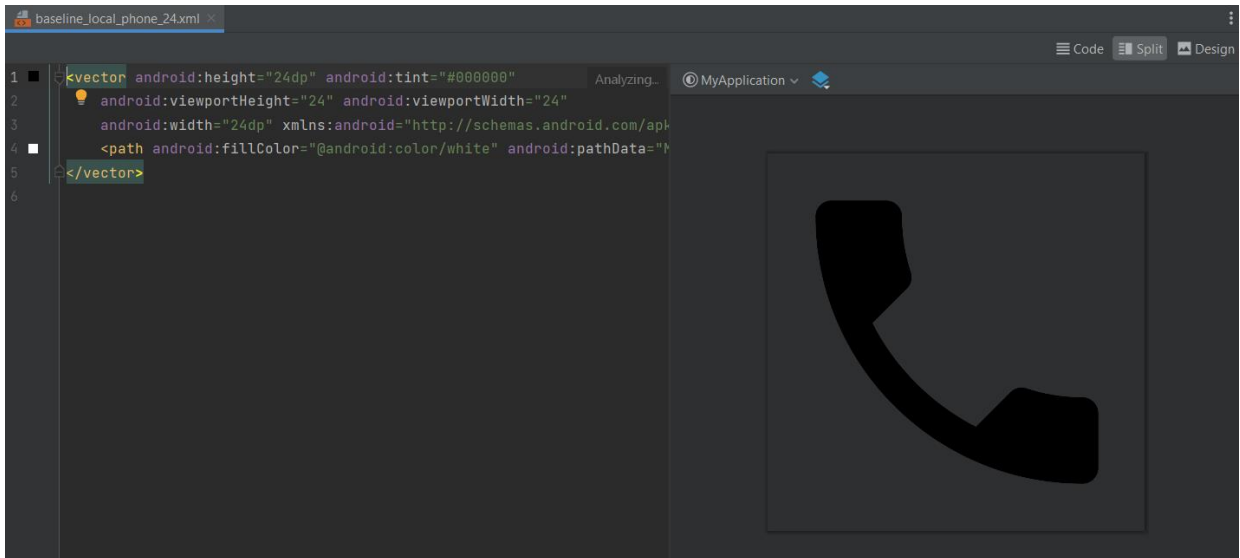
```

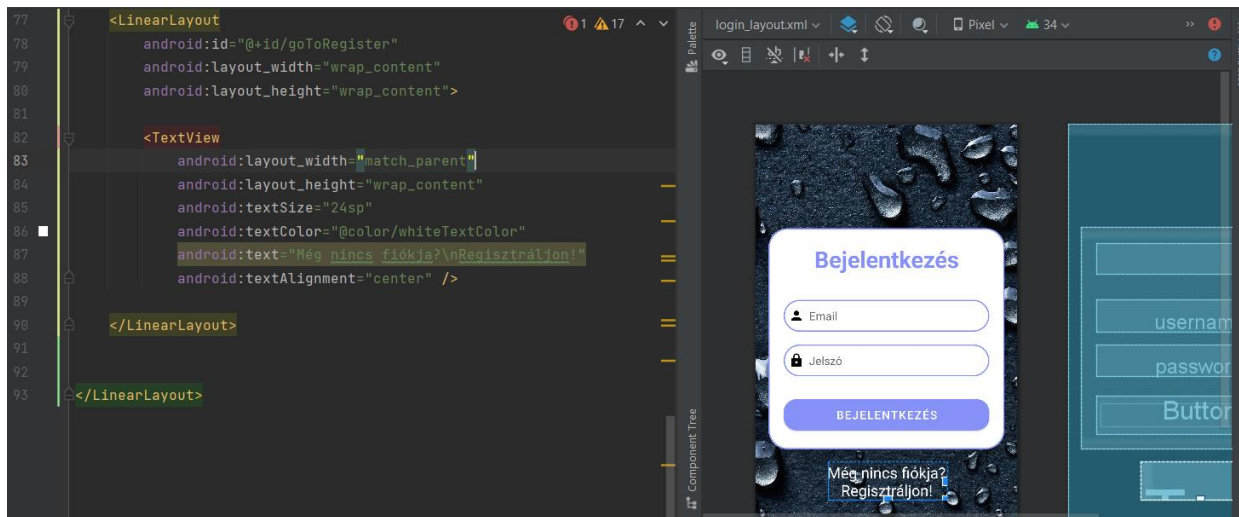
```

RegisterActivity.kt
1  package com.example.myapplication
2
3  import ...
16
17  class RegisterActivity : AppCompatActivity() {
18
19      private lateinit var btnRegister: Button
20      private lateinit var username: EditText
21      private lateinit var password: EditText
22      private lateinit var passwordAgain: EditText
23      private lateinit var fullName: EditText
24      private lateinit var phoneNumber: EditText
25
26      private lateinit var goToLogin: LinearLayout
27
28      private var isAllowed: Boolean = false
29
30      private lateinit var myDb : ShoesDB
31      private lateinit var userDao: UserDao
32
33      override fun onCreate(savedInstanceState: Bundle?) {
34          super.onCreate(savedInstanceState)

```

Ennek befejeztével a fizikális kezelőfelülettel foglalkoztam. Behúztam a kellő resource fájlokat, amikkel úgymond “díszítettem” az appot. Ezek között találhatóak ikonok, színek, elrendezések, formák, csak hogy említsek egy párat. Szerencsére az Andoid Studio segítségemre volt ebben, mert több előre definiált vector-t vagy layout-ot lehetett használni az alkalmazáshoz, sok időt meg lehetett takarítani vele.





AZ ALKALMAZÁS FUNKCIÓI KÖZÉ TARTOZNAK AZ ALÁBBIK:

- Regisztráció
- Bejelentkezés
- Kijelentkezés
- Új felhasználó létrehozása
- Cipők listájának megjelenítése
- Cipők adatainak módosítása
- Cipők és rögzített címek eltávolítása
- Cipők vásárlása

Pár kép a futó alkalmazásról:

