

# DOKUMENTÁCIÓ

Mobil programozási alapok

Féléves feladat

Shopping List Application

Készítette: **Vékony Róbert**

Neptunkód: **H0F0SZ**

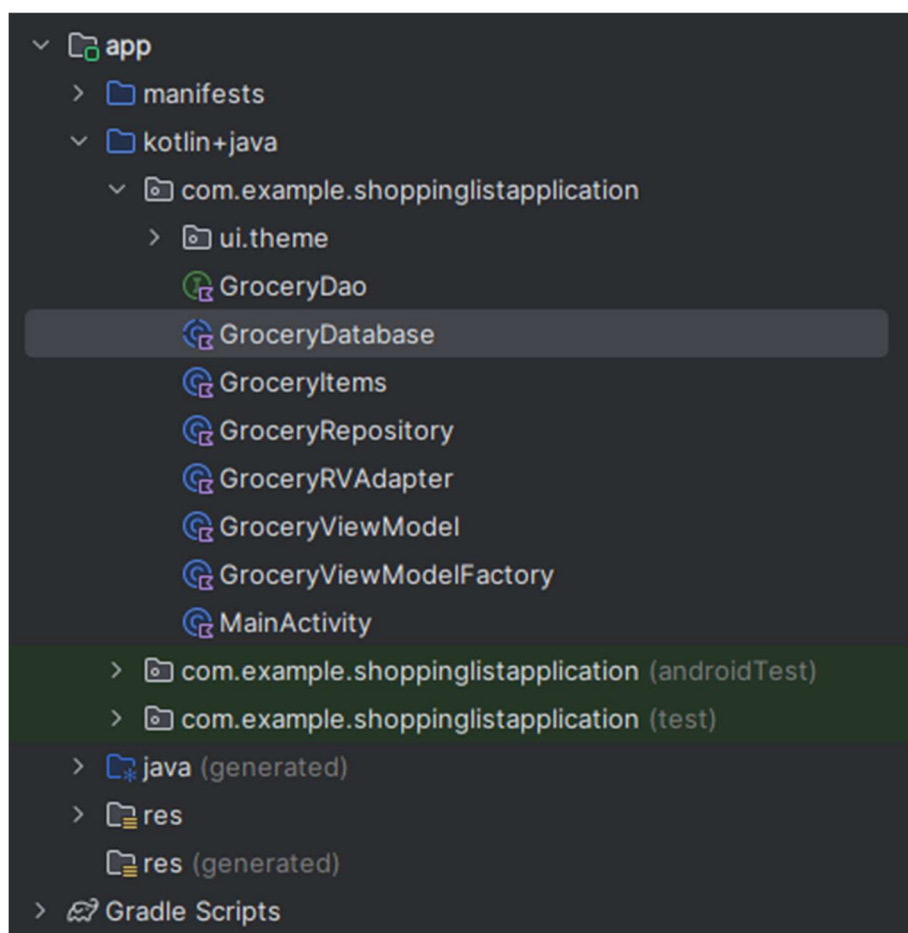
Dátum: **2024.01.28**

## Tartalom

Feladat ismertetése:.....	3
A Program felépítése:.....	4
GroceryDao.kt .....	4
Metódusok .....	4
GroceryDatabase.kt.....	5
GroceryItems.kt.....	6
GroceryRepository.kt .....	7
GroceryRVAdapter.kt .....	8
GroceryViewModel.kt .....	9
GroceryViewModelFactory.kt .....	9
MainActivity.kt .....	10
fun onCreate(savedInstanceState: Bundle?) .....	10
fun openDialog().....	11
activity_main.xml .....	12
grocery_add_dialog.xml .....	13

### Feladat ismertetése:

A Shopping List Application project célja, egy egyszerűen kezelhető bevásárló lista mobil alkalmazás fejlesztése. Android rendszerre készült készült, amihez Room adatbázis és ViewModel segítségével valósítja meg a termékek rögzítését és kezelését. Az minimalista felhasználói felületnek köszönhetően egyszerűen lehet a bevásárló listához hozzáadni, törölni, vagy módosítani termékeket.



## A Program felépítése:

GroceryDao.kt

```
GroceryDao.kt x
1 package com.example.shoppinglistapplication
2
3 > import ...
4
5 @Dao
6 interface GroceryDao {
7     @Insert(onConflict = OnConflictStrategy.REPLACE)
8     suspend fun insert(item: GroceryItems)
9     @Delete
10    suspend fun delete(item: GroceryItems)
11    @Query("SELECT * FROM Grocery_items")
12    fun getAllGroceryItems(): LiveData<List<GroceryItems>>
13 }
```

Az "android.room" könyvtár által nyújtott szolgáltatásokat használja ez a DAO (Data Access Object) interface az @insert, @delete, @query műveletek végrehajtásához.

### Metódusok

**insert(item: GroceryItems):** Az adatbázisba létrehozható vele egy új elem, valamint az onConflict paraméterrel, megadható, hogy valamilyen ütközés következtében mi történjen.

**delete(item: GroceryItems):** Létrehozott tételeknek az adatbázisból való eltávolítására szolgál.

**getAllGroceryItems():LiveData<List<GroceryItems>>:** A Grocery\_items tábla összes rekordját visszaadja GroceryItems-ként.

## GroceryDatabase.kt

```
1 package com.example.shoppinglistapplication
2
3 > import ...
4
5
6
7
8 private const val DATABASE_NAME = "GroceryApp.db"
9
10 @Database(entities = [GroceryItems::class], version = 1, exportSchema = false)
11 abstract class GroceryDatabase : RoomDatabase() {
12
13     abstract fun groceryDao(): GroceryDao
14
15     companion object {
16         @Volatile
17         private var instance: GroceryDatabase? = null
18         private val LOCK = Any()
19
20         operator fun invoke(context: Context): GroceryDatabase =
21             instance ?: synchronized(LOCK) {
22                 instance ?: buildDatabase(context).also { instance = it }
23             }
24
25         private fun buildDatabase(context: Context): GroceryDatabase =
26             Room.databaseBuilder(
27                 context.applicationContext,
28                 GroceryDatabase::class.java,
29                 DATABASE_NAME
30             ).build()
31     }
32 }
```

Ez az osztály az alkalmazás Room adatbázisát reprezentálja.

A `RoomDatabase` absztrakt osztály leszármazottjaként szolgál, amely tartalmazza a DAO-hoz való hozzáférést.

Implementálja a Singleton mintát a hatékony adatbázis-kezelés érdekében.

A **`buildDatabase`** függvény a Room library segítségével építi fel az adatbázist.

```
GroceryItems.kt ×  
1 package com.example.shoppinglistapplication  
2  
3 > import ...  
6  
7  
8 @Entity(tableName = "Grocery_items")  
9 data class GroceryItems (  
10     @ColumnInfo(name = "itemName")  
11     var itemName:String,  
12  
13  
14     @ColumnInfo(name = "itemQuantity")  
15     var itemQuantity:Double,  
16  
17  
18     @ColumnInfo(name = "itemPrice")  
19     var itemPrice:Double,  
20 )  
21 {  
22     @PrimaryKey(autoGenerate = true)  
23     var id:Int?=null  
24 }  
25
```

Ez az entitás osztály reprezentálja a bevásárlási tételt az adatbázisban.

A @Entity annotációval jelöli meg, hogy az osztály egy Room adatbázis entitás.

A tábla neve "Grocery\_items", és az oszlopok a tétel nevét, mennyiségét, árát és egyedi azonosítóját (id) tartalmazzák.

## GroceryRepository.kt

```
1 package com.example.shoppinglistapplication
2
3 class GroceryRepository(private val groceryDatabase: GroceryDatabase) {
4
5     suspend fun insert(items: GroceryItems) {
6         groceryDatabase.groceryDao().insert(items)
7     }
8
9     suspend fun delete(items: GroceryItems) {
10        groceryDatabase.groceryDao().delete(items)
11    }
12
13    fun getAllItems() = groceryDatabase.groceryDao().getAllGroceryItems()
14 }
15
```

Ez az osztály közvetíti az alkalmazás és az adatbázis közötti kommunikációt.

Az **insert** és **delete** függvények segítségével a ViewModel számára lehetővé teszi az adatok módosítását.

A **getAllItems** függvényen keresztül a ViewModel kéréseit továbbítja az adatbázis felé.

## GroceryRVAdapter.kt

```
1 package com.example.shoppinglistapplication
2
3 > import ...
4
5
6
7
8
9
10 class GroceryRVAdapter(
11     var list: List<GroceryItems>,
12     private val groceryItemClickInterface: GroceryItemClickInterface
13 )
14 : RecyclerView.Adapter<GroceryRVAdapter.GroceryViewHolder>() {
15
16
17     inner class GroceryViewHolder(itemView: View): RecyclerView.ViewHolder(itemView){
18         val nameTV: TextView = itemView.findViewById<TextView>(R.id.idtvitemname)
19         val quantityTV = itemView.findViewById<TextView>(R.id.idtvquantity)
20         val rateTV = itemView.findViewById<TextView>(R.id.idtvrate)
21         val totalTV = itemView.findViewById<TextView>(R.id.idvtotalamount)
22         val deleteIV = itemView.findViewById<ImageView>(R.id.idivdelete)
23     }
24
25
26
27
28 interface GroceryItemClickInterface{
29     fun onItemClick(groceryItems: GroceryItems)
30 }
31
32
33 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GroceryViewHolder {
34     val view = LayoutInflater.from(parent.context).inflate(R.layout.grocery_rv_item,parent, attachToRoot: false)
35     return GroceryViewHolder(view)
36 }
37
38
39 override fun onBindViewHolder(holder: GroceryViewHolder, position: Int) {
40     val currentItem = list[position]
41
42     holder.nameTV.text = currentItem.itemName
43     holder.quantityTV.text = currentItem.itemQuantity.toString()
44     holder.rateTV.text = "Ft: " + currentItem.itemPrice.toString()
45
46     val itemTotal: Double = currentItem.itemQuantity * currentItem.itemPrice
47     holder.totalTV.text = "Ft: " + itemTotal.toString()
48
49     holder.deleteIV.setOnClickListener { it: View!
50         groceryItemClickInterface.onItemClick(currentItem)
51     }
52 }
53
54 override fun getItemCount(): Int {
55     return list.size
56 }
57 }
```

RecyclerView adapter osztály, amely felelős a bevásárlólista elemeinek megjelenítéséért.

Az **onCreateViewHolder** és **onBindViewHolder** függvényekben konfigurálja a nézeteket és adataikat.

Tartalmazza az GroceryViewHolder belső osztályt, amely a nézeteket tartalmazza.



## GroceryViewModel.kt

```
GroceryViewModel.kt x
1 package com.example.shoppinglistapplication
2
3 > import ...
4
5
6
7
8 class GroceryViewModel(private val repository: GroceryRepository):ViewModel() {
9     fun insert(items: GroceryItems) = GlobalScope.launch { this: CoroutineScope
10         repository.insert(items)
11     }
12     fun delete(items: GroceryItems) = GlobalScope.launch { this: CoroutineScope
13         repository.delete(items)
14     }
15     fun getAllGroceryItems() = repository.getAllItems()
16 }
```

ViewModel osztály, amely az alkalmazás logikáját kezeli.

Az **insert** és **delete** függvényeken keresztül kommunikál a repository-val.

A **getAllGroceryItems** függvényen keresztül figyeli a változásokat az adatbázisban, és frissíti a nézetet.

## GroceryViewModelFactory.kt

```
GroceryViewModelFactory.kt x
1 package com.example.shoppinglistapplication
2
3 > import ...
4
5
6
7 class GroceryViewModelFactory(private val repository: GroceryRepository):ViewModelProvider.NewInstanceFactory() {
8     override fun <T : ViewModel> create(modelClass: Class<T>): T {
9         return GroceryViewModel(repository) as T
10     }
11 }
```

Segédosztály, amely lehetővé teszi a ViewModelProvider számára, hogy létrehozza a GroceryViewModel példányt.

## MainActivity.kt

fun onCreate(savedInstanceState: Bundle?)

```
MainActivity.kt x
1 package com.example.shoppinglistapplication
2
3 > import ...
15
16 </> class MainActivity : AppCompatActivity(), GroceryRVAdapter.GroceryItemClickInterface {
17     lateinit var itemRV: RecyclerView
18     lateinit var addFAB: FloatingActionButton
19     lateinit var list: List<GroceryItems>
20     lateinit var groceryRVAdapter: GroceryRVAdapter
21     lateinit var groceryViewModel: GroceryViewModel
22     override fun onCreate(savedInstanceState: Bundle?) {
23         super.onCreate(savedInstanceState)
24         setContentView(R.layout.activity_main)
25         itemRV = findViewById(R.id.rvitems)
26         addFAB = findViewById(R.id.fabAdd)
27         list = ArrayList<GroceryItems>()
28         groceryRVAdapter = GroceryRVAdapter(list, groceryItemClickInterface: this)
29         itemRV.layoutManager = LinearLayoutManager(context: this)
30         itemRV.adapter = groceryRVAdapter
31         val groceryRepository = GroceryRepository(GroceryDatabase(context: this))
32         val factory = GroceryViewModelFactory(groceryRepository)
33         groceryViewModel = ViewModelProvider(owner: this, factory).get(GroceryViewModel::class.java)
34         groceryViewModel.getAllGroceryItems().observe(owner: this, Observer { it: List<GroceryItems>!
35             groceryRVAdapter.list = it
36             groceryRVAdapter.notifyDataSetChanged()
37         })
38         addFAB.setOnClickListener { it: View!
39             openDialog()
40         }
41     }
42 }
```

Az Activity életciklusának kezdetén hívódik meg.

Beállítja az Activity tartalmát a activity\_main.xml elrendezési fájl alapján. Inicializálja a RecyclerView-t, a FloatingActionButton-ot, a ViewModel-t és az Adapter-t.

Regisztrálja az Observer-t a ViewModel által visszaadott LiveData-vel, hogy frissítse a felületet az adatok változása esetén.

Beállítja a FloatingActionButton kattintási eseményét, hogy megnyissa a dialógusablakot az új bevásárlási tételek hozzáadásához.

fun openDialog()

```
MainActivity.kt x
42
43 fun openDialog(){
44     val dialog = Dialog(context: this)
45     dialog setContentView(R.layout.grocery_add_dialog)
46     val btnCancel = dialog.findViewById<AppCompatButton>(R.id.idbtncancel)
47     val addbtn = dialog.findViewById<AppCompatButton>(R.id.idbtnadd)
48     val itemEdt = dialog.findViewById<EditText>(R.id.idEdititemname)
49     val itemPriceEdt = dialog.findViewById<EditText>(R.id.idEdititemprice)
50     val itemQuantityEdt = dialog.findViewById<EditText>(R.id.idEdititemquantity)
51     btnCancel.setOnClickListener { it: View!
52         dialog.dismiss()
53     }
54     addbtn.setOnClickListener { it: View!
55         val itemname:String = itemEdt.text.toString()
56         val itemprice:String = itemPriceEdt.text.toString()
57         val itemquantity:String = itemQuantityEdt.text.toString()
58         val qty : Double = itemquantity.toDouble()
59         val pr : Double = itemprice.toDouble()
60         if (itemname.isNotEmpty() && itemprice.isNotEmpty() && itemquantity.isNotEmpty()){
61             val items = GroceryItems(itemname,qty,pr)
62             groceryViewModel.insert(items)
63             Toast.makeText(applicationContext, text: "Elem hozzáadva", Toast.LENGTH_SHORT).show()
64             groceryRVAdapter.notifyDataSetChanged()
65             dialog.dismiss()
66         }
67         else{
68             Toast.makeText(applicationContext, text: "Kérlek mindent tölts ki!", Toast.LENGTH_SHORT).show()
69         }
70     }
71     dialog.show()
72 }
```

Megnyit egy dialógusablakot az új bevásárlási tételek hozzáadásához.

Beállítja a dialógusablak nézetét a grocery\_add\_dialog.xml elrendezési fájl alapján.

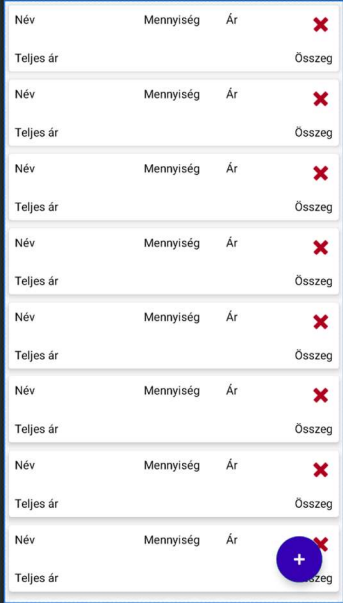
Rögzíti a felhasználó által megadott adatokat, és hozzáadja az új tételt az adatbázishoz a ViewModel segítségével.

## activity\_main.xml

activity\_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:background="#F9F9F9"
7     android:layout_height="match_parent"
8     tools:context=".MainActivity">
9
10
11
12
13     <androidx.recyclerview.widget.RecyclerView
14         android:id="@+id/rvitems"
15         android:layout_width="match_parent"
16         android:layout_height="match_parent"
17         tools:listitem="@layout/grocery_rv_item">
18
19
20     </androidx.recyclerview.widget.RecyclerView>
21
22
23     <com.google.android.material.floatingactionbutton.FloatingActionButton
24         android:id="@+id/fabAdd"
25         android:layout_width="wrap_content"
26         android:layout_height="wrap_content"
27         android:layout_alignParentEnd="true"
28         android:layout_alignParentBottom="true"
29         android:layout_marginStart="25dp"
30         android:layout_marginTop="25dp"
31         android:layout_marginEnd="25dp"
32         android:layout_marginBottom="25dp"
33         android:backgroundTint="@color/purple_700"
34         android:elevation="5dp"
35         android:src="@android:drawable/ic_input_add"
36         app:tint="@color/white" />
37 </RelativeLayout>
```

activity\_main.xml



## grocery\_add\_dialog.xml

The image shows the Android Studio interface with the XML code for `grocery_add_dialog.xml` on the left and a visual preview on the right.


**XML Code:**

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"
3   android:layout_width="match_parent"
4   android:layout_height="wrap_content"
5   xmlns:app="http://schemas.android.com/apk/res-auto"
6   android:layout_gravity="center"
7   android:backgroundTint="#FEF8F6"
8   android:background="@drawable/btn_bg"
9   app:cardCornerRadius="10dp">
10
11   <RelativeLayout
12     android:layout_width="match_parent"
13     android:layout_height="491dp">
14
15     <TextView
16       android:id="@+id/idtvHeading"
17       android:layout_width="match_parent"
18       android:layout_height="wrap_content"
19       android:layout_marginStart="4dp"
20       android:layout_marginTop="4dp"
21       android:layout_marginEnd="4dp"
22       android:layout_marginBottom="4dp"
23       android:background="@color/teal_700"
24       android:gravity="center"
25       android:padding="4dp"
26       android:text="Elem hozzáadása"
27       android:textAlignment="center"
28       android:textAllCaps="false"
29       android:textColor="@color/white"
30       android:textSize="20sp"
31       android:textStyle="bold" />
32
33     <LinearLayout
34       android:layout_width="match_parent"
35       android:layout_height="wrap_content"
36       android:layout_below="@id/idtvHeading"
37       android:layout_alignParentBottom="true"
38       android:layout_marginBottom="91dp"
39       android:orientation="vertical">
40
41       <com.google.android.material.textfield.TextInputLayout
42         android:layout_width="match_parent"
43         android:layout_height="wrap_content"
44         android:layout_margin="10dp"
45         android:padding="5dp"
46         app:counterTextColor="@color/black"
47         app:hintTextColor="@color/black">
```

**Visual Preview:**

The preview shows a dialog titled "Elem hozzáadása" (Add Item) with a teal header. It contains three input fields: "Név" (Name), "Mennyiség" (Quantity), and "Ár" (Price). At the bottom, there are two buttons: "Bezárás" (Close) and "Hozzáadás" (Add).

```
47 ■      app:hintTextColor="@color/black">
48
49      <com.google.android.material.textfield.TextInputEditText
50          android:id="@+id/idEditItemname"
51          android:layout_width="match_parent"
52          android:layout_height="57dp"
53          android:hint="Név"
54          android:inputType="text"
55          android:textSize="14sp"
56          app:counterTextColor="@color/black"
57          app:hintTextColor="@color/black" />
58  </com.google.android.material.textfield.TextInputLayout>
59
60  <com.google.android.material.textfield.TextInputLayout
61      android:layout_width="match_parent"
62      android:layout_height="83dp"
63      android:layout_margin="10dp"
64      android:padding="5dp"
65      app:counterTextColor="@color/black"
66      app:hintTextColor="@color/black">
67
68      <com.google.android.material.textfield.TextInputEditText
69          android:id="@+id/idEditItemquantity"
70          android:layout_width="match_parent"
71          android:layout_height="wrap_content"
72          android:hint="Mennyiség"
73          android:inputType="numberDecimal"
74          android:textSize="14sp"
75          app:counterTextColor="@color/black"
76          app:hintTextColor="@color/black" />
77  </com.google.android.material.textfield.TextInputLayout>
78
79  <com.google.android.material.textfield.TextInputLayout
80      android:layout_width="match_parent"
81      android:layout_height="wrap_content"
82      android:layout_margin="10dp"
83      android:padding="5dp"
84      app:counterTextColor="@color/black"
85      app:hintTextColor="@color/black">
86
87      <com.google.android.material.textfield.TextInputEditText
88          android:id="@+id/idEditItemprice"
89          android:layout_width="match_parent"
90          android:layout_height="wrap_content"
91          android:hint="Ár"
92          android:inputType="numberDecimal"
93          android:textSize="14sp"
94          app:counterTextColor="@color/black"
95          app:hintTextColor="@color/black" />
96  </com.google.android.material.textfield.TextInputLayout>
97
```



```
97
98
99  <LinearLayout
100      android:layout_width="match_parent"
101      android:layout_height="wrap_content"
102      android:layout_margin="5dp"
103      android:orientation="horizontal"
104      android:weightSum="2">
105
106      <androidx.appcompat.widget.AppCompatButton
107          android:id="@+id/idbtncancel"
108          android:layout_width="0dp"
109          android:layout_height="wrap_content"
110          android:layout_margin="8dp"
111          android:layout_weight="1"
112          android:background="@color/material_dynamic_secondary20"
113          android:padding="5dp"
114          android:text="Bezárás"
115          android:textAllCaps="false"
116          android:textColor="@color/white" />
117
118      <androidx.appcompat.widget.AppCompatButton
119          android:id="@+id/idbtnadd"
120          android:layout_width="0dp"
121          android:layout_height="wrap_content"
122          android:layout_margin="8dp"
123          android:layout_weight="1"
124          android:background="@color/teal_700"
125          android:padding="5dp"
126          android:text="Hozzáadás"
127          android:textAllCaps="false"
128          android:textColor="@color/white" />
129  </LinearLayout>
130 </RelativeLayout>
131 </androidx.cardview.widget.CardView>
```

