Mobil megoldások

Kocsis Gergely

2019.01.11.

LiveData

Folytassuk az előző órán elkezdett projektet.

Első lépésként helyettesítsük a MainActivity-ben használt `refreshScrollView` metódust egy fejlettebb megoldással.

Használjuk a LiveData osztályt, mely az Android SDK eszköze a dinamikusan változó adatok követésére. A LiveData tulajdonképpen az Observeer (Figyelő) minta megvalósítása.

LiveData

Folytassuk az előző órán elkezdett projektet.

Első lépésként helyettesítsük a MainActivity-ben használt `refreshScrollView` metódust egy fejlettebb megoldással.

Használjuk a LiveData osztályt, mely az Android SDK eszköze a dinamikusan változó adatok követésére. A LiveData tulajdonképpen az Observeer (Figyelő) minta megvalósítása.

Alakítsuk át a DAO osztályunkban a lekérdező metódust úgy, hogy az LiveData típusú eredményt adjon:

```
@Query("SELECT * FROM ShoppingList")
LiveData<List<ShoppingListItem>> getAllItems();
```

Ezzel az adatbázis lekérésünk “figyelhetővé” vált.

LiveData

Most a MainActivity-ben szedjük ki a `refreshScrollView` mindkét hívását. Ezután állítsunk be egy figyelőt az OnCreate metódusban.

```
shoppingListDatabase.shoppingListDAO().getAllItems().observe(this,
new Observer<List<ShoppingListItem>>() {
    @Override
    public void onChanged(List<ShoppingListItem> shoppingListItems) {
        itemsView.setText(shoppingListItems.toString());
    }
});
```

Ha mindent jól csináltunk, akkor innentől kezdve már működik is.

RecyclerView

Második feladatként alakítsuk át a felhasználói felületet.

A könnyebb tesztelhetőségért adjunk egy törlő gombot is a felülethez.

```
//MainActivity
public void clearDB(View view) {
    new Thread(new Runnable() {
        @Override
        public void run() {
            shoppingListDatabase.shoppingListDAO().clearDB();
        }
    }).start();
}

//ShoppingListDAO
@Query("DELETE FROM ShoppingList")
void clearDB();
```

RecyclerView

Cseréljük ki a ScrollView-t egy RecyclerView-val. Figyeljünk oda, hogy a MainActivity-ben ezután hiba lesz a korábban használt referenciák miatt.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/my_recycler_view"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="16dp"
    android:layout_marginLeft="16dp"
    android:layout_marginTop="16dp"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="16dp"
    android:layout_marginBottom="16dp"
    app:layout_constraintBottom_toTopOf="@+id/editText"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

RecyclerView

Az iméntelkészített RecyclerView lesz a lista elemeinek tárolója. Szükség van egy újabb XML megadására, ami azt írja le, hogy egy elem hogy nézzen ki.

Hozzuk ezt lééétre a res→layout alatt item_layout.xml néven

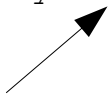
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="wrap_content">
            <TextView
                android:id="@+id/textView2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Item name: " />

            <TextView
                android:id="@+id/textView"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:textAppearance="@style/TextAppearance.AppCompat.Large"/>
</LinearLayout>
```

RecyclerView

Készítsünk egy Adapter osztályt, mely a listánk elemeit beteszi a RecyclerView-ba

```
public class ViewAdatpter extends RecyclerView.Adapter<ViewAdatpter.MyViewHolder>
```



Belső osztály, amin keresztül kapcsoljuk az elemeket leíró xml-t az Adapterhez

A belső osztály definíciója:

```
public static class MyViewHolder extends RecyclerView.ViewHolder {
    TextView tv;
    public MyViewHolder(View view){
        super(view);
        MyViewHolder.this.tv=view.findViewById(R.id.textView);
    }
}
```

RecyclerView

Készítsünk egy Adapter osztályt, mely a listánk elemeit beteszi a RecyclerView-ba

Az adatok tárolását egy belső változóval oldjuk meg. Ezt a példányosításkor inicializáljuk. A getItemCount metódusban is ennek a méretét adjuk vissza.

```
private List<ShoppingListItem> data;

public ViewAdatpter(List<ShoppingListItem> data){
    this.data=data;
}

@Override
public int getItemCount() {
    return data.size();
}
```

RecyclerView

Készítsünk egy Adapter osztályt, mely a listánk elemeit beteszi a RecyclerView-ba

A maradék két metódus arra szolgál hogy az egyes adatokat a lista elemekhez kössük.

```
@NonNull
@Override
public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_layout,
parent, false);
    return new MyViewHolder(tv);
}

@Override
public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {
    holder.tv.setText(data.get(position).getItemName());
}
```

RecyclerView

Készítsünk egy Adapter osztályt, mely a listánk elemeit beteszi a RecyclerView-ba

Az onCreate metódus módosítását a következőképp végezzük el:

```
mRecyclerView = findViewById(R.id.my_recycler_view);
mRecyclerView.setLayoutManager(new LinearLayoutManager(MainActivity.this));

shoppingListDatabase.shoppingListDAO().getAllItems().observe(this, new
Observer<List<ShoppingListItem>>() {
    @Override
    public void onChanged(List<ShoppingListItem> shoppingListItems) {
        list.clear();
        list.addAll(shoppingListItems);

        mRecyclerView.setAdapter(new ViewAdatpter(list));
    }
});
```

RecyclerView

Szépítsük az alkalmazás megjelenését CardView használatával.

Interaktivitás hozzáadása:

```
public static class MyViewHolder extends RecyclerView.ViewHolder implements
View.OnClickListener{
    TextView tv;
    final ViewAdatpter mAdapter;
    public MyViewHolder(View tv, ViewAdatpter va){
        super(tv);
        MyViewHolder.this.tv=tv.findViewById(R.id.textView);
        itemView.setOnClickListener(this);
        this.mAdapter=va;
    }

    @Override
    public void onClick(View view) {
        int mPosition = getLayoutPosition();

        ShoppingListItem element = data.get(mPosition);
        element.setItemName("done");
        data.set(mPosition, element);

        mAdapter.notifyDataSetChanged();
    }
}
```