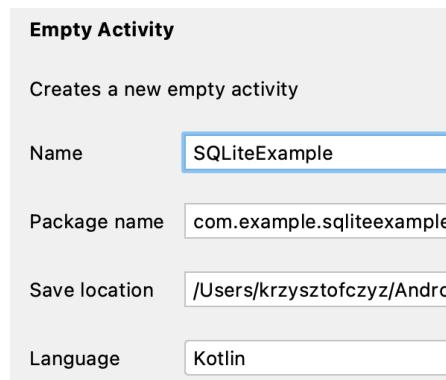


Obsługa własnej bazy danych

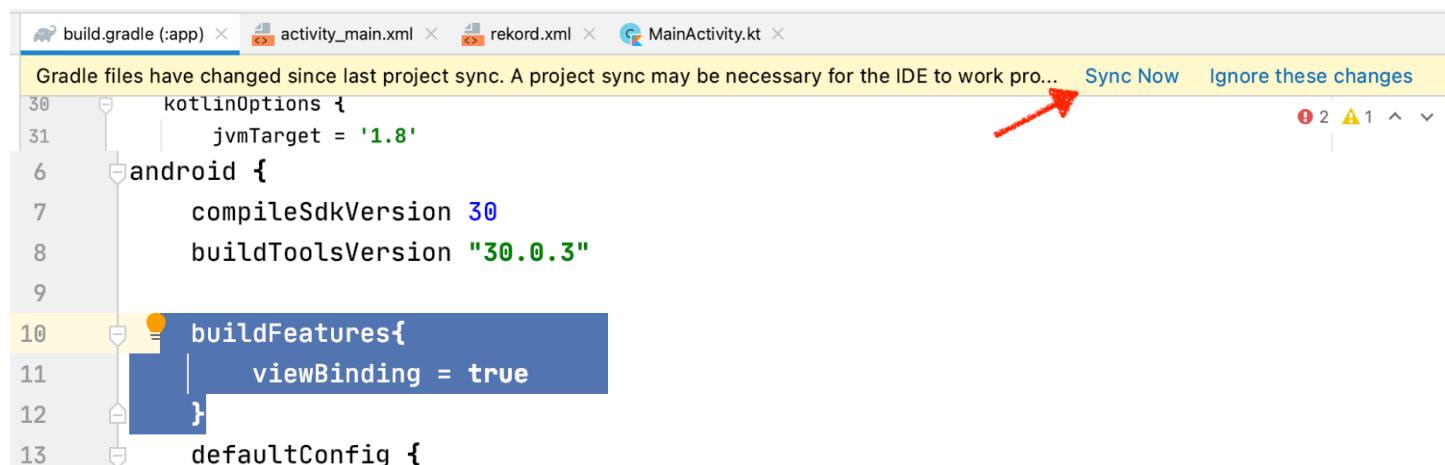
Utworzenie bazy danych w AS ma na celu zapis danych z aplikacji oraz ich odczyt. Przydatne to jest w szczególności w trakcie wyłączenia i ponownego włączenia aplikacji, która ma działać tak jak w momencie wyłączenia a nie podczas pierwszego startu aplikacji. Jednym słowem przechowujemy dane aplikacji w lokalnej bazie danych. Lokalne dane mogą być przechowywane na karcie SD lub w bazie danych.

Tworzymy aplikację – nowy kotlinowy projekt:

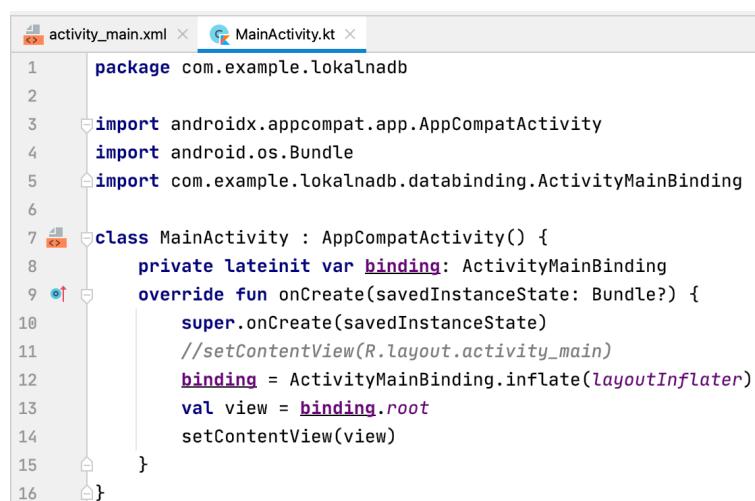


Wybieramy pustą aktywność.

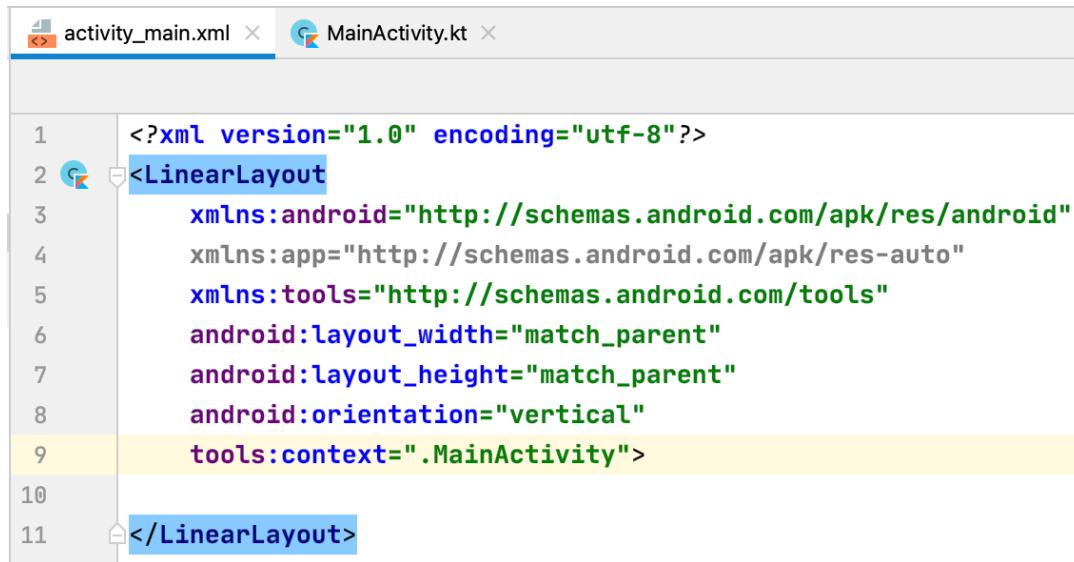
W pliku build.gradle (dla Modules) należy dodać linie i zsynchronizować z projektem:



Tradycyjnie w MainActivity.kt należy zmodyfikować zawartość



Tworzymy LinearLayout typu vertical (oczywiście w pliku activity_main.xml):



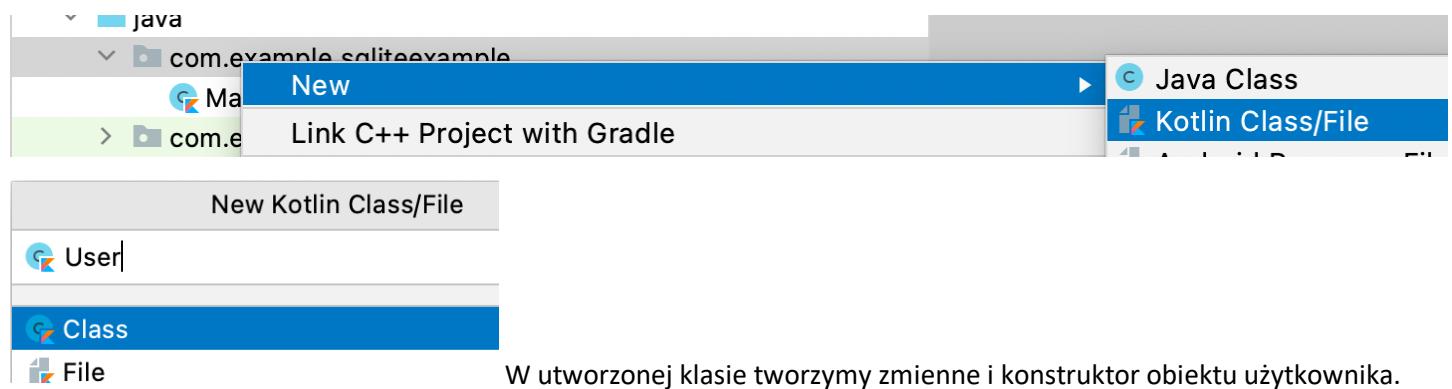
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:orientation="vertical"
9     tools:context=".MainActivity">
10
11 </LinearLayout>
```

W layoutie tworzymy kolejny LinearLayout: z polami edycyjnymi oraz przyciskiem zapisu.

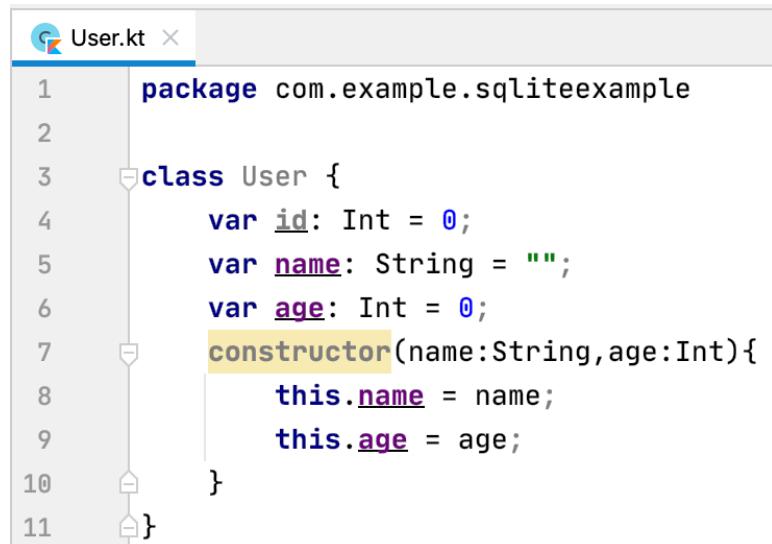


```
10
11 <LinearLayout
12     android:layout_width="match_parent"
13     android:layout_height="wrap_content"
14     android:orientation="vertical">
15
16     <EditText
17         android:id="@+id/etName"
18         android:layout_width="match_parent"
19         android:layout_height="wrap_content"
20         android:layout_margin="10dp"
21         android:padding="10dp"/>
22
23     <EditText
24         android:id="@+id/etAge"
25         android:layout_width="match_parent"
26         android:layout_height="wrap_content"
27         android:layout_margin="10dp"
28         android:padding="10dp"
29         android:inputType="number"/>
30
31     <Button
32         android:id="@+id/btnZapisz"
33         android:layout_width="match_parent"
34         android:layout_height="wrap_content"
35         android:layout_margin="10dp"
36         android:padding="10dp"
37         android:text="@string/zapisz"/>
38
39 </LinearLayout>
```

Tworzymy klasę **User** reprezentującą dane użytkownika:



W utworzonej klasie tworzymy zmienne i konstruktor obiektu użytkownika.

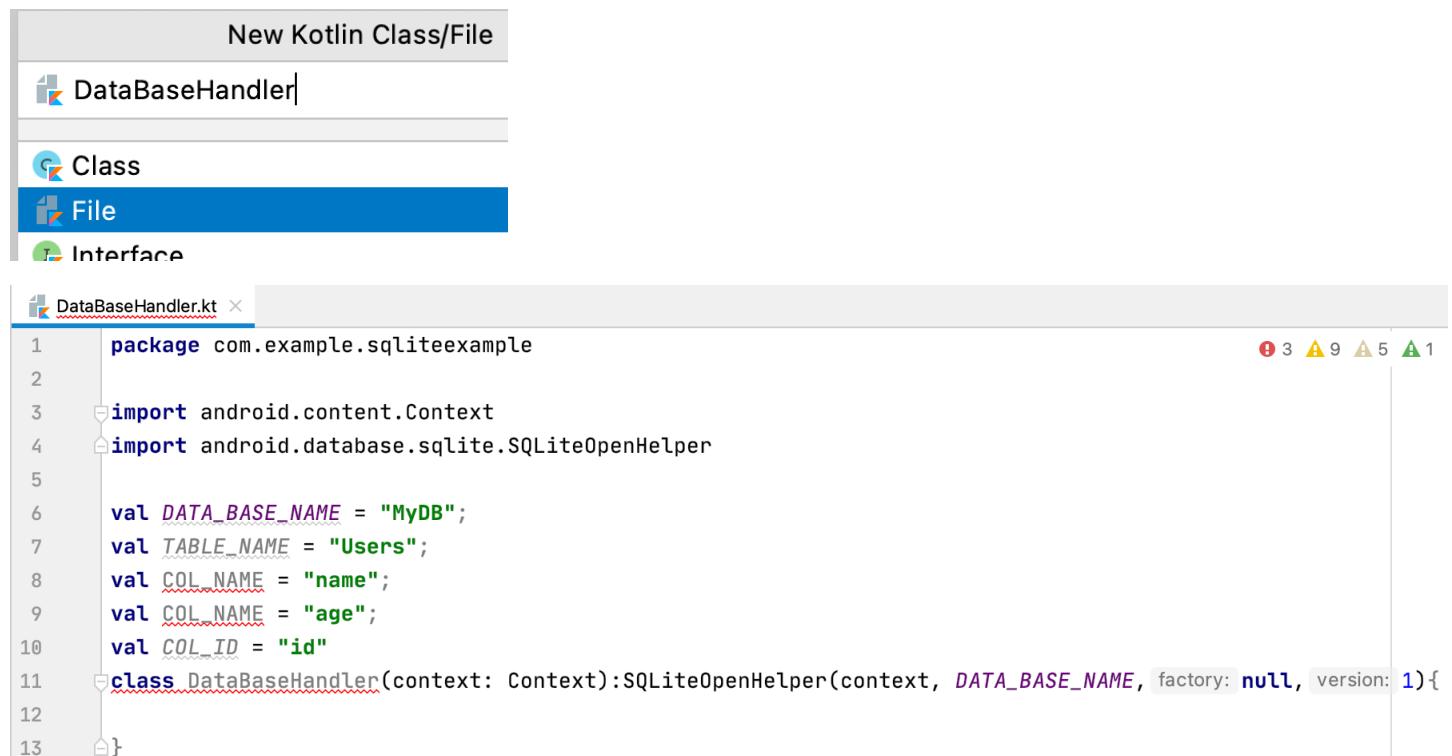


```

1 package com.example.sqliteexample
2
3 class User {
4     var id: Int = 0;
5     var name: String = "";
6     var age: Int = 0;
7     constructor(name:String,age:Int){
8         this.name = name;
9         this.age = age;
10    }
11 }

```

W podobny sposób tworzymy klasę tworzącą bazę danych:



```

1 package com.example.sqliteexample
2
3 import android.content.Context
4 import android.database.sqlite.SQLiteOpenHelper
5
6 val DATA_BASE_NAME = "MyDB";
7 val TABLE_NAME = "Users";
8 val COL_NAME = "name";
9 val COL_NAME = "age";
10 val COL_ID = "id"
11 class DataBaseHandler(context: Context):SQLiteOpenHelper(context, DATA_BASE_NAME, factory: null, version: 1){
12
13 }

```

Importujemy metody:

Implement Members

android.database.sqlite.SQLiteOpenHelper

- onCreate(db: SQLiteDatabase!): Unit
- onUpgrade(db: SQLiteDatabase!, oldVersion: Int, newVersion: Int): Unit

```

12  class DataBaseHandler(context: Context):SQLiteOpenHelper(context, DATA_BASE_NAME, factory: null, version: 1){
13
14     override fun onCreate(db: SQLiteDatabase?) {
15         TODO("Not yet implemented")
16     }
17
18     override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
19         TODO("Not yet implemented")
20     }
21 }

```

Metoda onCreate

```

14  class DataBaseHandler(var context: Context):SQLiteOpenHelper(context, DATA_BASE_NAME, factory: null, version: 1){
15      //tworzy bazę danych
16      override fun onCreate(db: SQLiteDatabase?) {
17          val createTable = "CREATE TABLE $TABLE_NAME (" +
18              "$COL_ID INTEGER PRIMARY KEY AUTOINCREMENT," +
19              "$COL_NAME VARCHAR(256)," +
20              "$COL_AGE INTEGER)";
21          db?.execSQL(createTable)
22      }

```

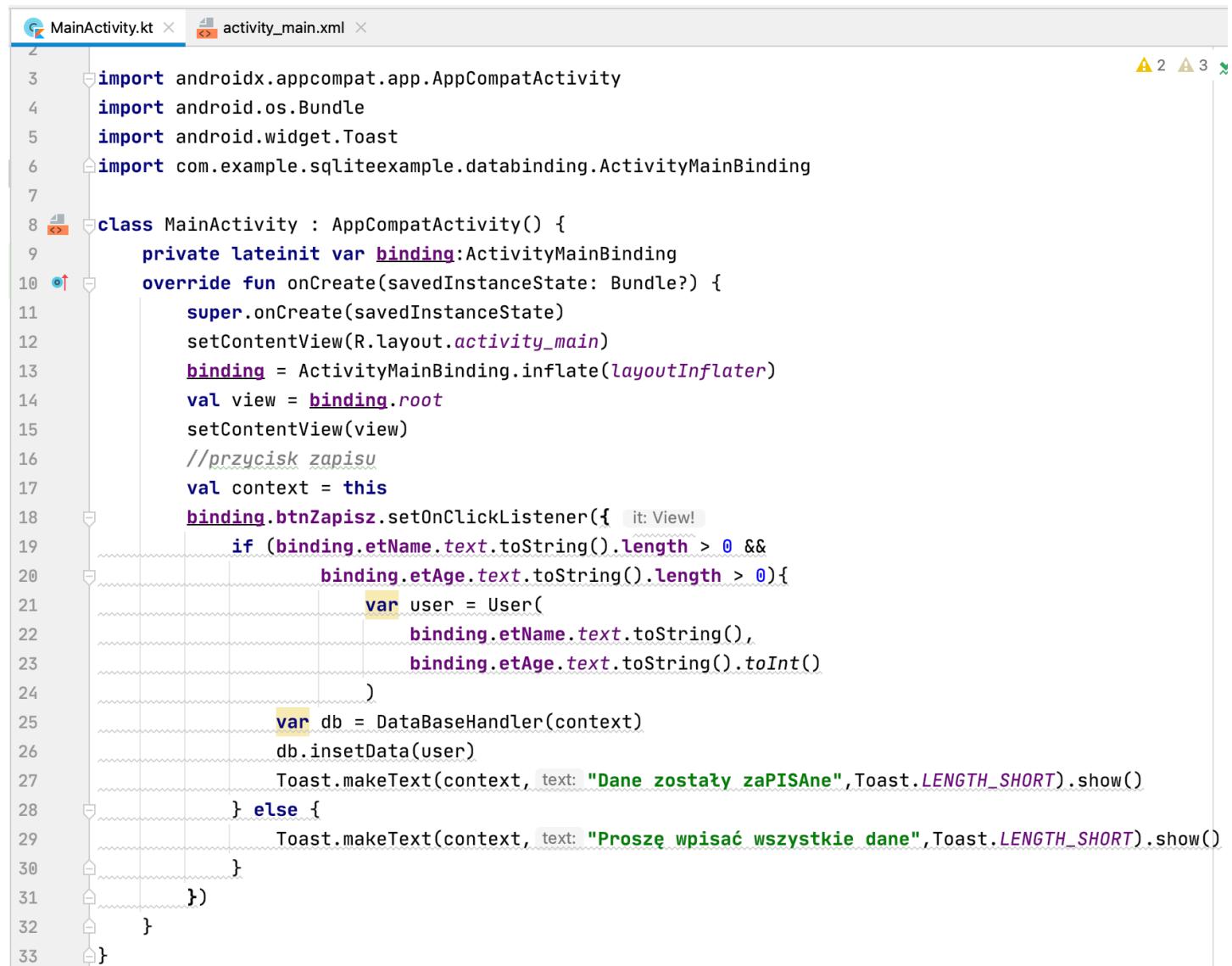
Budujemy też funkcję obsługującą przycisk i wpisującą dane do tabeli:

```

27  //funkcja wstawiajaca dane
28  fun insetData(user:User){
29      val db = this.writableDatabase
30      var cv = ContentValues()
31      cv.put(COL_NAME,user.name)
32      cv.put(COL_AGE,user.age)
33      var result = db.insert(TABLE_NAME, nullColumnHack: null, cv)
34      if (result == -1.toLong()){
35          Toast.makeText(context, text: "Failed",Toast.LENGTH_SHORT).show()
36      } else{
37          Toast.makeText(context, text: "Success",Toast.LENGTH_SHORT).show()
38      }
39  }

```

Zmieniamy zachowanie aplikacji po kliknięciu w przycisk wpisujący dane. Przechodzimy do MainActivity.kt :



```
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.widget.Toast
6 import com.example.sqliteexample.databinding.ActivityMainBinding
7
8 class MainActivity : AppCompatActivity() {
9     private lateinit var binding: ActivityMainBinding
10    override fun onCreate(savedInstanceState: Bundle?) {
11        super.onCreate(savedInstanceState)
12        setContentView(R.layout.activity_main)
13        binding = ActivityMainBinding.inflate(layoutInflater)
14        val view = binding.root
15        setContentView(view)
16        //przycisk zapisu
17        val context = this
18        binding.btnZapisz.setOnClickListener{ it: View!
19            if (binding.etName.text.toString().length > 0 &&
20                binding.etAge.text.toString().length > 0){
21                var user = User(
22                    binding.etName.text.toString(),
23                    binding.etAge.text.toString().toInt()
24                )
25                var db = DataBaseHandler(context)
26                db.insertData(user)
27                Toast.makeText(context, text: "Dane zostały zaPISane", Toast.LENGTH_SHORT).show()
28            } else {
29                Toast.makeText(context, text: "Proszę wpisać wszystkie dane", Toast.LENGTH_SHORT).show()
30            }
31        }
32    }
33}
```

Cz.2 c.d.n.