

Projektowanie aplikacji mobilnych

Marek Kwak □ Piotr Węgrzyn

Plan prezentacji



```
graph LR; A((Interfejs graficzny)) --- B((Oprogramowanie)); B --- C((Baza danych)); C --- D((Zadania))
```

Interfejs
graficzny

Oprogramo
wanie

Baza
danych

Zadania

Interfejs graficzny – XML

Extensible Markup Language – XML

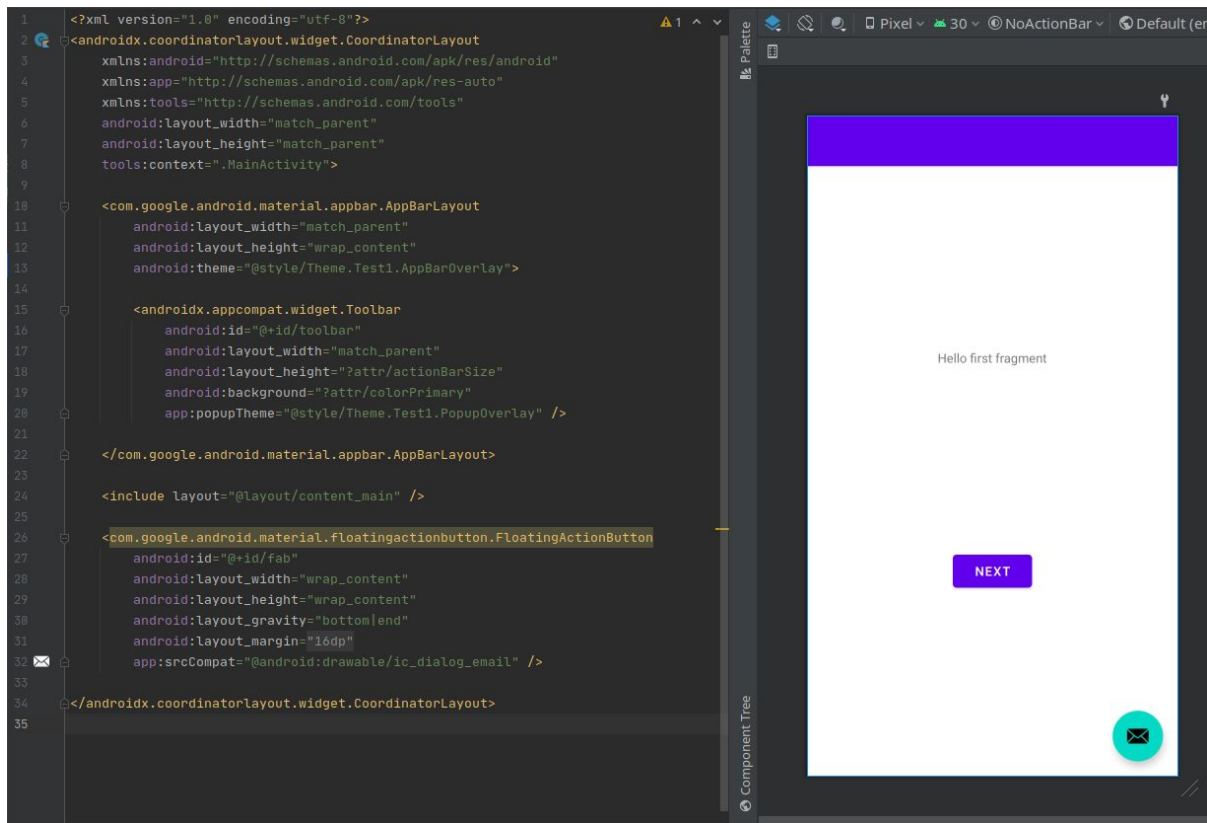
XML to uniwersalny język znaczników (podobny do HTML) szerokiego zastosowania. Służy do uporządkowanego zapisu danych.

W projektowaniu interfejsu dla aplikacji mobilnych na system Android XML spełnia funkcję opisu wyglądu aplikacji. Za jego pomocą możemy łatwo dodawać i formatować położenie elementów na ekranie.

Interfejs w XML

Elementy w interfejsie XML znajdują się w “Layout’ach” (które też są elementami), czyli agregatach wyglądu – określają one tryb rozkładu elementów na ekranie.

Elementy posiadają atrybuty, takie jak wysokość, szerokość, marginesy, paddingi, kolory, zawartości graficzne i tekstowe etc.



XML – przykładowe parametry elementów interfejsu

— — —

`android:layout_width`

`android:background`

`android:layout_height`

`android:foreground`

`android:gravity`

`android:padding`

`android:text`

`android:margin`

`android:textSize`

`android:orientation`

`android:textStyle`

[i wiele innych](#)

```
1 <TextView
2     android:layout_width="match_parent"
3     android:layout_height="wrap_content"
4     android:textSize="24sp"
5     android:textStyle="bold"
6     android:padding="5dp"
7     android:gravity="center"
8     android:text="TEXT" />
```

Cechy elementu w XML

```
1 textview {
2     display: block;
3     height: fit-content;
4     font-size: 1.2em;
5     font-weight: bold;
6     padding: 5px;
7     align-content: center;
8 }
```

Cechy elementu w CSS

Oprogramowanie – język Kotlin

Język programowania Kotlin

Kotlin jest multiplatformowym językiem programowania wysokiego poziomu. Jest w pełni kompilowalny do Javy, ale ma przyjemniejszą składnię – ponadto można stosować te dwa języki równocześnie.

Ze względu na swoją strukturę, Kotlin wymusza obsługę ewentualnych wyjątków `NullPointerException` oraz ułatwia zachowanie porządku w kodzie.

[Więcej informacji](#)

```

class MapsActivity : FragmentActivity(), OnMapReadyCallback {
    private val mMap: GoogleMap? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_maps)
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        val mapFragment = supportFragmentManager
            .findFragmentById(R.id.map) as SupportMapFragment?
        mapFragment!!.getMapAsync( callback: this)
    }

    override fun onMapReady(googleMap: GoogleMap) {
        val dane = Data.getInstance()
        val polylineOptions = PolylineOptions()
        for (location in dane.points) {
            val point = LatLng(location.Latitude, location.Longitude)
            polylineOptions.add(point)
        }
        polylineOptions.color(Color.rgb( red: 0, green: 120, blue: 200))
        googleMap.addPolyline(polylineOptions)
    }
}

```

Kod napisany w Kotlinie

```

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync( callback: this);
    }

    @Override
    public void onMapReady(@NotNull GoogleMap googleMap) {

        Data dane = Data.getInstance();
        PolylineOptions polylineOptions = new PolylineOptions();
        for (Location location : dane.getPoints()) {
            LatLng point = new LatLng(location.getLatitude(), location.getLongitude());
            polylineOptions.add(point);
        }
        polylineOptions.color(Color.rgb( red: 0, green: 120, blue: 200));
        googleMap.addPolyline(polylineOptions);
    }
}

```

Kod napisany w Javie

O co chodzi z tym pytajnikiem ?

```
var b: String? = "abc" // can be set null
b = null // ok
print(b)
var a: String = "abc" // Regular initialization means non-null by default
a = null // compilation error
```

```
if ( null != nullableVariable) {
    nullableVariable.someMethodCall()
} else {
    // fallback flow
}
```

```
nullableVariable?.someMethodCall()
```

Petle

```
for (i in 1..3) {  
    println(i)  
}
```

```
for (i in array.indices) {  
    println(array[i])  
}
```

```
val max = if (a > b) a else b
```

Baza danych – SQLite

SQLite – baza relacyjna

SQLite to wariacja na temat PostgreSQL'a. Ma w wielu środowiskach swoje implementacje i zastosowanie ze względu na szybkość działania i niewielki narzut.

W bibliotekach Androida jest implementowany przez dwie biblioteki – `android.database.sqlite` oraz `androidx.room`.

Plik z bazą danych znajduje się w pamięci wewnętrznej aplikacji, a dzięki użyciu Singletonu można otrzymać do niej dostęp z dowolnego miejsca w aplikacji.

Database Structure		
Browse Data Edit Pragmas Execute SQL		
Create Table Create Index Print		
Name	Type	Schema
Tables (3)		
1		CREATE TABLE "1" ("point_id" INTEGER, "point_latitude"
point_id	INTEGER	"point_id" INTEGER
point_latitude	REAL	"point_latitude" REAL
point_longitude	REAL	"point_longitude" REAL
Tracks		CREATE TABLE "Tracks" ("track_id" INTEGER, "track_name"
track_id	INTEGER	"track_id" INTEGER
track_name	TEXT	"track_name" TEXT
track_location	TEXT	"track_location" TEXT
track_date	TEXT	"track_date" TEXT
track_time	INTEGER	"track_time" INTEGER
track_distance	TEXT	"track_distance" TEXT
track_minimap	BLOB	"track_minimap" BLOB
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
Indices (0)		
Views (0)		
Triggers (0)		

```

20  val databaseHelper = dane.databaseHelper
21
22  val cursor: Cursor = databaseHelper.getCursorToTracksTable()
23
24  if (cursor.count != 0) {
25      while (cursor.moveToNext()) {
26          val temp = MainItem(
27              cursor.getInt(0),
28              cursor.getString(1),
29              cursor.getString(2),
30              cursor.getString(3),
31              cursor.getInt(4),
32              cursor.getString(5)
33          )
34          itemsList.add(temp)
35      }
36  }

```

Struktura bazy SQLite

Dane umieszczane są w tabelach. Każdy rekord to unikalny zestaw danych.

Wykorzystanie w kodzie

Odpowiedź na zapytanie otrzymujemy za pomocą klasy Cursor, z której następnie wyciągamy potrzebne nam dane.

Zadania

Streszczenie zadań – całość dostępna na GitHubie

Zadanie 1

Proste projektowanie elementów
interfejsu graficznego w XML

Zadanie 2

Podstawy Kotlinu na przykładzie
implementacji algorytmu

Zadanie 3

Praktyczne wykorzystanie bazy
danych w projekcie

— — —

Dziękujemy za uwagę

Powodzenia z zadaniami!