Laboratorium 5 - instrukcje do ćwiczeń

1. Tworzenie listy za pomocą RecyclerView

1.1. Zdefiniowanie wyglądu pojedynczego elementu listy

- Utworzyć plik o nazwie ***item_resource.xml*** - widok pojedynczego elementu listy.

- Dodać klasę o nazwie ***ResourceViewHolder*** dziedziczącą po ***RecyclerView.ViewHolder:***

```
class ResourceViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val categoryText: TextView = view.category_text
    ...
}
```

- Utworzyć klasę o nazwie ***ResourceAdapter***, dziedziczącą po ***RecyclerView.Adapter<ResourceViewHolder>***:

```
class ResourcesAdapter(
    private val resources: List<Resource>,
    private val onResourceSelected: (Resource) -> Unit
) : RecyclerView.Adapter<ResourceViewHolder>() {

    override fun onCreateViewHolder(
        parent: ViewGroup,
        viewType: Int
    ): ResourceViewHolder = ResourceViewHolder(
        LayoutInflater.from(parent.context).inflate(R.layout.item_resource, parent, false)
    )
```

```
override fun onBindViewHolder(
    holder: ResourceViewHolder,
    position: Int
) {
    holder.apply {
        val resource = resources[position]
        categoryIcon.setImageResource(resource.categoryIconResId)
        // TODO categoryText.text = ...
        // TODO toggleStateIconVisibility(...)
        container.setOnClickListener { onContainerClicked(resource, position) }
    }
}

private fun ResourceViewHolder.toggleStateIconVisibility(
    isActive: Boolean
) {
    stateIcon.visibility =
        if (isActive) {
            View.VISIBLE
        } else {
            View.INVISIBLE
        }
}
```

```
    private fun onContainerClicked(
        resource: Resource,
        position: Int
    ) {
        if (!resource.isActive) {
            // TODO Zaznaczyć wybrany element oraz odznaczyć pozostałe elementy.
            // Element jest wybrany, jeśli flaga isActive przyjmuje wartość true.
            notifyDataSetChanged()
        }
    }

    override fun getItemCount() = resources.size
}
```

1.2. Utworzenie oraz konfguracja listy

- Utworzyć widok zawerający listę

(komponent ***androidx.recyclerview.widget.RecyclerView***).

- Z poziomu fragmentu skonfigurować listę w następujący sposób:

layoutManager: LinearLayoutManager

adapter: ResourcesAdapter

2. Integracja z Google Maps API

2.1. Konfiguracja biblioteki

- Wygenerować API key (https://cloud.google.com/maps-platform/?apis=maps)

- W pliku ***AndroidManifest.xml*** dodać następującą konfigurację:

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="YOUR_API_KEY"/>
```

- Dodać bliblioteke play-services-maps do projektu:
implementation 'com.google.android.gms:play-services-maps:16.1.0'

2.2. Implementacja

- Przygotować kontener do prezentacji mapy:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

- Dodać aktywność ***MapActivity*** wyświetlającą widok utworzony w poprzednim kroku.

- Utworzyć prezenter implementujący interfejs ***OnMapReadyCallback***:

```
override fun onMapReady(
    map: GoogleMap?
```

```
) {
    googleMap = map
    googleMap?.apply {
        val userCoordinates = LatLng(latitude, longitude)
        moveCamera(
            CameraUpdateFactory.newLatLngZoom(
                userCoordinates,
                INITIAL_ZOOM)
            )
            // TODO mapType = ... ustawić wybrany przez siebie typ mapy
    }
}
```

MAP ZOOM LEVELS:

1: World

5: Landmass / continent

10: City

15: Streets

20: Buildings