

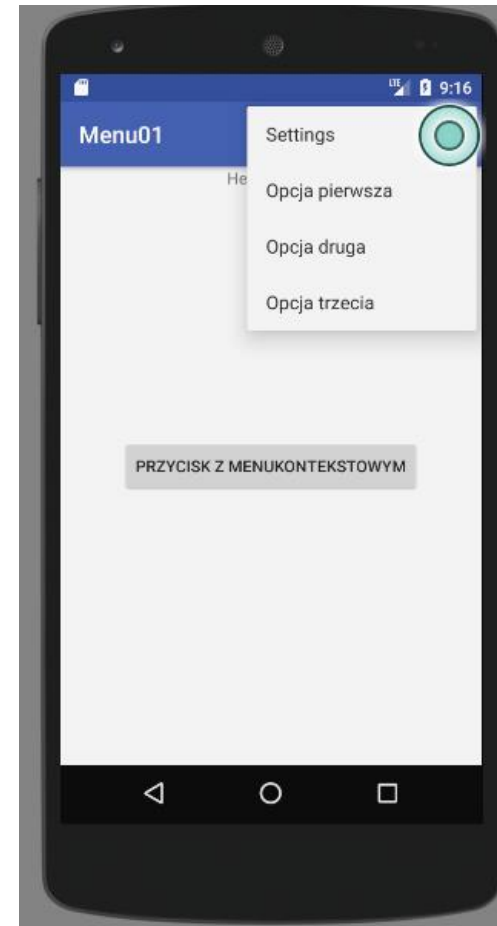
APLIKACJE MOBILNE

MENU, MENU KONTEKSTOWE

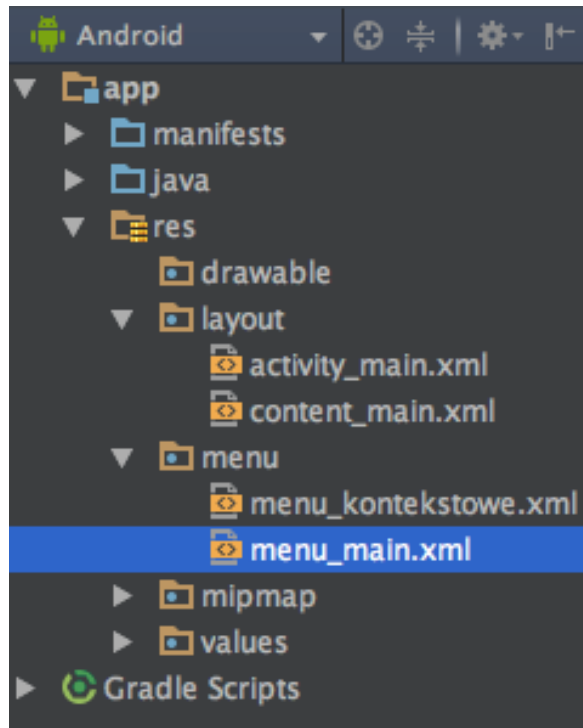
dr Artur Bartoszewski

Menu opcji

Standardowe menu opcji w
Androidzie



I – przygotowanie zawartości menu



W folderze res tworzymy folder menu. W nim dodajemy plik main_menu.xml

```
menu item
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      tools:context="pl.uniwersytetradom.bartoszewski.artur.menu01.MainActivity">
    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:title="Settings"
        app:showAsAction="never" />
    <item
        android:id="@+id/zrob_cos_01"
        android:orderInCategory="101"
        android:title="Opcja pierwsza"
        app:showAsAction="never" />
    <item
        android:id="@+id/zrob_cos_02"
        android:orderInCategory="102"
        android:title="Opcja druga"
        app:showAsAction="never" />
    <item
        android:id="@+id/zrob_cos_03"
        android:orderInCategory="103"
        android:title="Opcja trzecia"
        app:showAsAction="never" />
</menu>
```

[118N] Hardcoded string "Opcja pierwsza", should use @string resource [more...](#) (%F1)

Elementem głównym jest `<menu />`. Każdej pozycji odpowiada `<item />`

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}
```

- W kodzie .java nadpisujemy funkcję **onCreateOptionsMenu(Menu menu)**.
- W funkcji tej umieszczamy obiekt **getMenuInflater().inflate()** jego zadaniem jest rozwinięcie layoutu menu , który otrzymał w parametrze.
- Na tym etapie menu wyświetla się, lecz jeszcze nic nie robi.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();
    TextView t1 = (TextView) findViewById(R.id.textView01);

    switch (id) {
        case R.id.action_settings: t1.setText("Wybrano setings"); return true;
        case R.id.zrob_cos_01: t1.setText("Wybrano opcje pierwsza"); return true;
        case R.id.zrob_cos_02: t1.setText("Wybrano opcje druga"); return true;
        case R.id.zrob_cos_03: t1.setText("Wybrano opcje trzecia"); return true;
    }

    return super.onOptionsItemSelected(item);
}
```

Kolejnym krokiem jest dodanie funkcji **onOptionsItemSelected(MenuItem item)**.

Funkcja otrzymuje w parametrze wskaźnik do elementu listy, który ją wywołał (item)

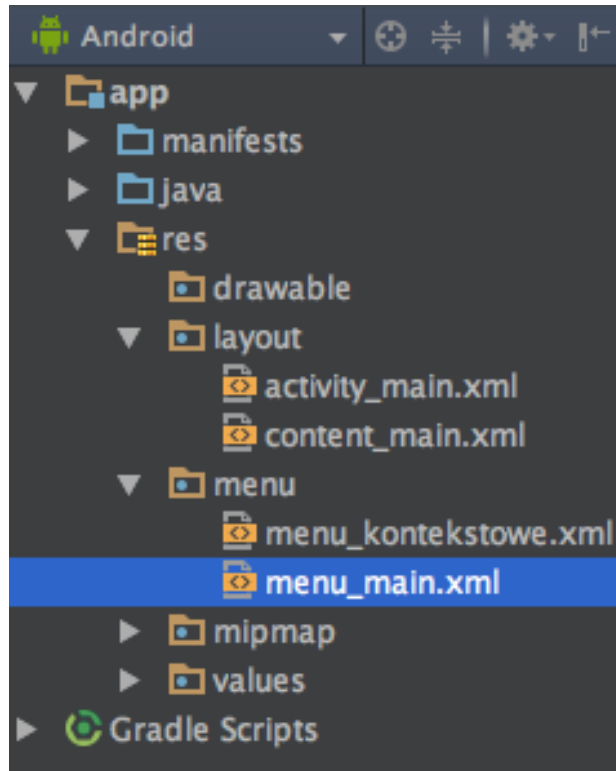
Menu kontekstowe

Menu pojawiające się po długim dotknięciu (przytrzymaniu) kontrolki.

Działa niezależnie od onClick oraz słuchacza kliknięć.



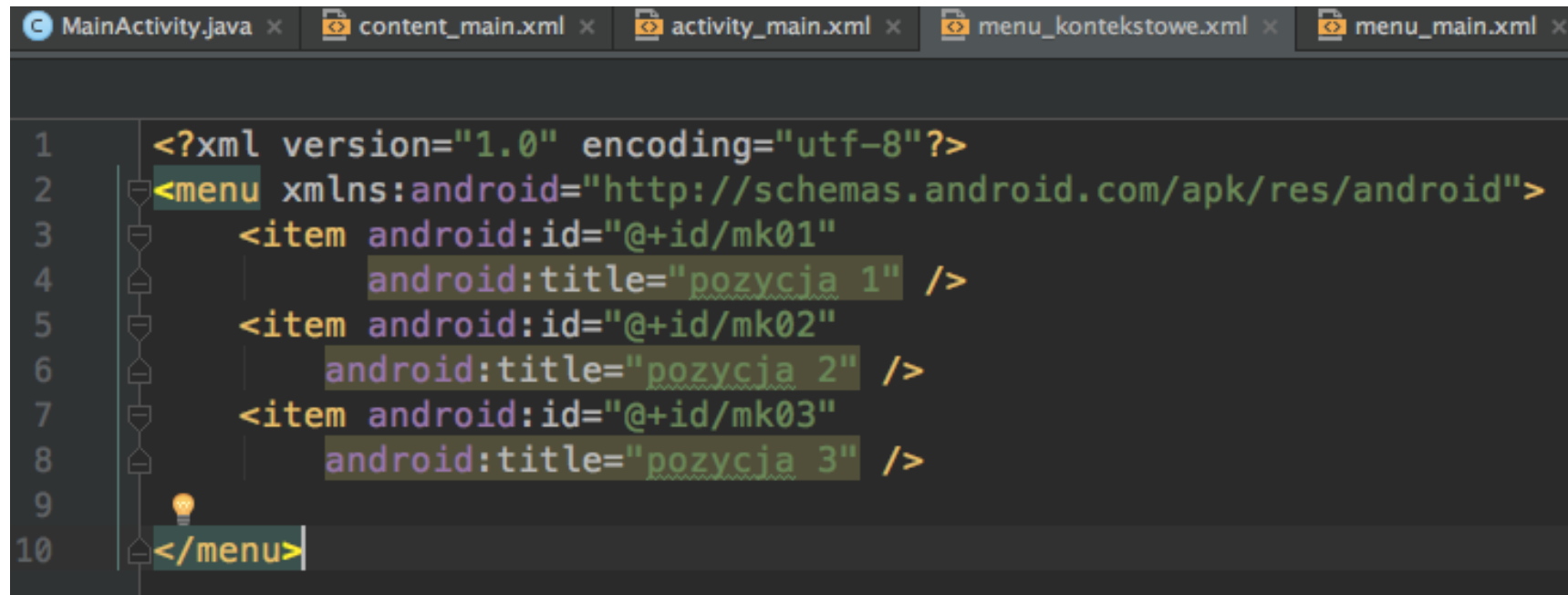
I – przygotowanie zawartości menu



W folderze res tworzymy folder menu.

W nim dodajemy **plik menu_kontekstowe.xml** (nazwa własna)

II – przygotowanie zawartości menu



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android">
3      <item android:id="@+id/mk01"
4          android:title="pozycja 1" />
5      <item android:id="@+id/mk02"
6          android:title="pozycja 2" />
7      <item android:id="@+id/mk03"
8          android:title="pozycja 3" />
9
10 </menu>
```

II – powiązanie menu z przyciskiem

```
17      @Override
18      protected void onCreate(Bundle savedInstanceState) {
19          super.onCreate(savedInstanceState);
20          setContentView(R.layout.activity_main);
21          Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
22          setSupportActionBar(toolbar);
23
24          Button button = (Button) findViewById(R.id.button01);
25          registerForContextMenu(button);
26
27      }
```

- W **onCreate** odnajdujemy uchwyt do przycisku (lub innego elementu, któremu który chcemy wyposażyć w menu kontekstowe).
- Rejestrujemy menu – poleceniem **registerForContextMenu()** z parametrem, którym jest uchwyt do przycisku

III – Wyświetlenie menu

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {
    getMenuInflater().inflate(R.menu.menu_kontekstowe, menu);
    super.onCreateContextMenu(menu, v, menuInfo);
}
```

- W kodzie .java nadpisujemy funkcję **onCreateContextMenu()**.
- W funkcji tej umieszczamy obiekt **getMenuInflater().inflate()** jego zadaniem jest rozwinięcie layoutu menu , który otrzymał w parametrze.
- Na tym etapie menu wyświetla się, lecz jeszcze nic nie robi.

IV – Obsługa zdarzenia kliknięcia

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();
    TextView t1 = (TextView) findViewById(R.id.textView01);

    switch (id) {
        case R.id.mk01: t1.setText("Wybrano opcje pierwsza z menu kontekstowego"); return true;
        case R.id.mk02: t1.setText("Wybrano opcje druga z menu kontekstowego"); return true;
        case R.id.mk03: t1.setText("Wybrano opcje trzecia z menu kontekstowego"); return true;
    }
    return super.onOptionsItemSelected(item);
}
```

- Kolejnym krokiem jest dodanie funkcji **onContextSelected(MenuItem item)**.
- Funkcja otrzymuje w parametrze wskaźnik do elementu listy, który ją wywołał (item)

