Aplicații cu mai multe activități

O activitate reprezintă un ecran pe care se pot afișa diferite informații și/sau pune la dispoziția utilizatorului diferite componente grafice ce pot permite o interacțiune a acestuia cu aplicația în sine. Sunt situații în care, o aplicație mai complexă, are nevoie de mai multe activități pentru a expune informațiile/funcționalitățile sale. De exemplu, în urma selecției unei opțiuni dintr-o listă se dorește afișarea unor informații suplimentare într-o alta acvtivitate decât cea curentă. Se obține astfel o arhitectură de tipul **top to down**.

Android pune la dispoziție un mecanism simplu care permite crearea unei aplicații ce poate să conțină mai multe activități. In exemplu următor ne propunem să dezvoltam o aplicație care conține următoarea funcționalitate: mesajul scris de utilizator în activitatea principală sa fie afișat într-o altă activitate.

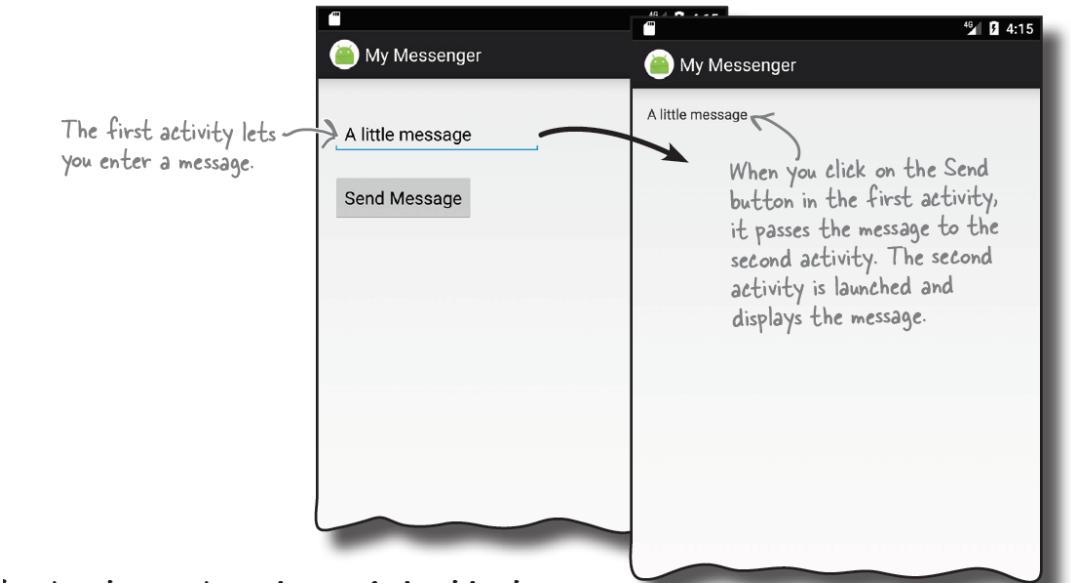

Figura 1 Exemplu Aplicație cu două activități

După ce proiectul Android a fost creat, aplicația are o singură activitate, cea principală care este afișată atunci când aplicația este deschisă. Pentru a creea o altă activitate în cadrul aplicației, se selectează pentru pachetul de surse opțiunea `File → New → Activity → Empty Activity`.

Observații:

1. Crearea unei noi activități conduce atât la crearea fișierului sursă java, cât și la crearea fișierului `layout.xml`.
2. Fiecare activitate din cadrul proiectului este configurată în fișierul `AndroidManifest.xml`.

Fișierul `manifest.xml` cuprinde informații importante pentru aplicația curentă, precum activitățile sale, librăriile necesare și alte declarări.

`activity android:name=".CreateMessageActivity"` //numele aplicației

// acestă activitate este afișată la lansarea aplicației

`<category android:name="android.intent.category.LAUNCHER" />`

//fiecare activitate trebuie specificată

```
application
...
      ...>
      <activity
      android:name=".MyActivityClassName"
      ...
      ...>
      ...
      </activity>
...
</application>
```

Conceptul Intent

Noțiunea `intent` reprezintă nucleul mecanismului prin care o activitate poate să comunice cu altă activitate. Practic, atunci când dintr-o activitate se dorește „lansarea" unei alte activități, se anunță platforma Android de intenția de a se realiza acestă operație („intent to do something"). Astfel, se obține o modalitate de a realiza o legătură între două obiecte.

Sinatxa*:* `Intent intent = new Intent(this, Target.class);`

Prin al doilea argument al constructorului clasei `Intent` se specifică activitatea ce urmează a fi deschisă. Pentru a îi transmite platformei Android intenția de a deschide activitatea se apelează metoda: `startActivity(intent)`.

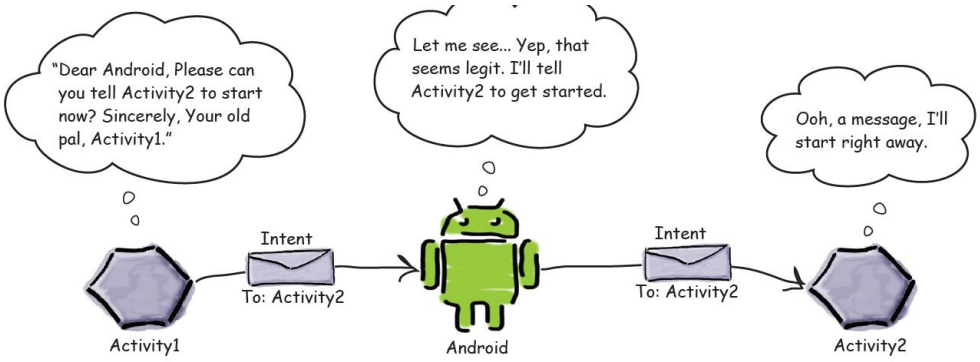

Figura 2 Lansarea unui Intent

Mai jos este listat codul java al activității principale prin care se anunță intenția de a deschide o nouă activitate cu denumirea ActivitateNoua.

```
public class TrimiteMesajActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

    Intent intent = new Intent(this, ActivitateNoua.class);
    startActivity(intent);

    }
}
```

Transmiterea informațiilor între activități

După cum ați vazut mai sus, odată creat obiectul de tip `Intent`, se anunță intenția de a deschide o activitate nouă creată în prealabil în cadrul proiectului aplicației. Sunt situații în care noua activitate poate să aibă nevoie de informațiile activității curente, precum texte, imagini etc. Astfel, obiectului de tip `Intent` i se pot asocia diferite informații ce pot fi preluate ulterior în activitatea nou lansată prin metoda apleul metodei membre `putExtra("message", value)`. Primul argument specifică numele valorii care umează a fi transmisă noii activități, iar al doilea argument specifică valoarea în sine.

Metoda `putExtra` este supraîncărcată, astfel informația transmisă poate fi de orice tip primitiv, String sau un tablou.

Informația este ulterior preluată prin apelul metodei `getIntent()`.

Exemple:

```
Intent intent = getIntent();

String string = intent.getStringExtra("message");
```

```
int intNum = intent.getIntExtra("name", default_value);
```

Observație

- Primul argument al metodei putExtra poate fi un câmp static, final al clasei care modelează noua activitate.

Exemplu:

În clasa activității principale:

```
intent.putExtra(PrimitMesaj.CAMPSATAICFINAL, mesajText);
```

În clasa activității secundare:

```
public static final String CAMPSATAICFINAL = "message";
```

Aplicație practică: Realizați aplicația **StarBuzz**, cu următoarea funcționalitate:

- La pornirea aplicației se afișează un Logo StarBuzz și o listă de categorii de produse comercializate
- La selecția unei categorii, se deschide o activitate ce conține un Logo StarBuzz și o listă de produse din categoria respectivă
- La selecția unui produs din lista de produse, se deschide o alta activitate care conține informații despre produsul selectat, precm o imagibe, un text descriptiv și prețul.

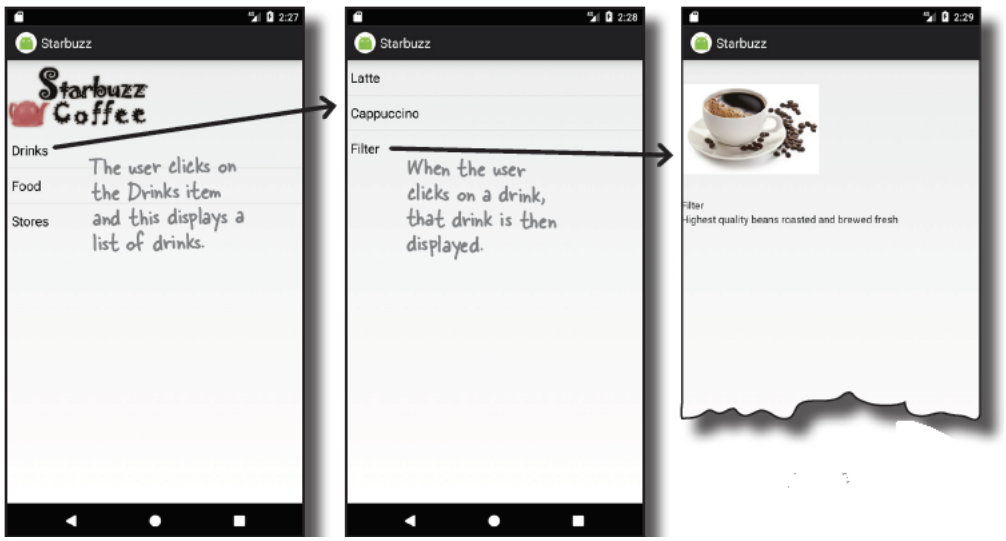

Figura 3 Aplicația StarBuzz

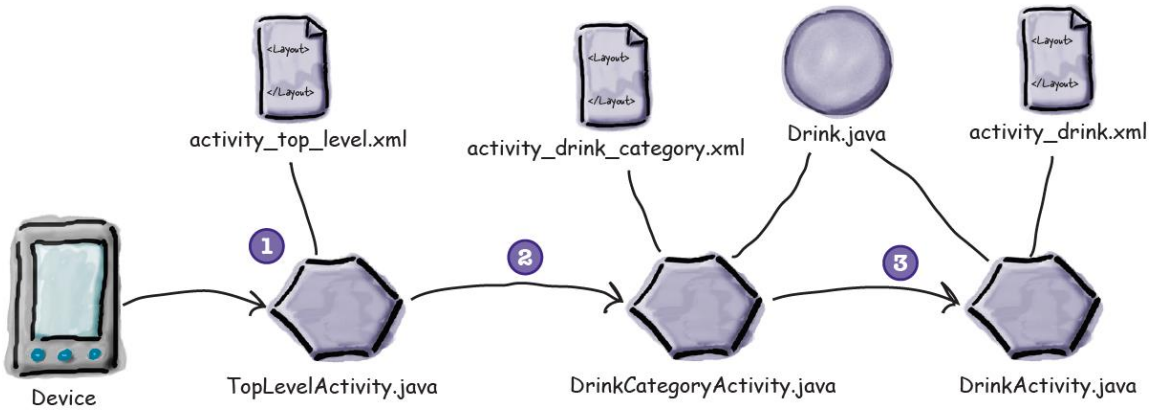

Figura 4 Structura proiect