Aplicatii mobile pentru afaceri

Master SIA

Lect. Octavian Dospinescu

2012

Tematica generala

- Controale grafice in Android
- Proprietati, metode, evenimente, listener-i specifici
- Interfete grafice
- Controale statice vs. controale dinamice

Controale grafice frecvente

- TextView
- EditText
- Button, ImageButton, ToggleButton
- CheckBox
- RadioButton, RadioGroup
- Spinner
- Lists

TextView

- este cel mai simplu widget pe care-l vom întâlni cel mai des, iar cu ajutorul său putem afișa text needitabil;

- datorită utilității sale îl putem privi ca pe o etichetă, deși cu ajutorul său **putem afișa link-uri active către pagini web, numere de telefon sau adrese de email**

TextView - atribute

- android:typeface - setează tipul caracterelor care vor fi folosite pentru afișarea textului, spre exemplu *serif*;

- android:textStyle - specifică stilul fontului, **bold**, *italic* sau ***combinate***;

TextView - atribute

- android:textColor – setează culoarea textului în format RGB, spre exemplu #fd3099 pentru roz bombon ☺;
- mai multe detalii aici: http://developer.android.com/reference/android/graphics/Color.html
- android:autoLink – putem seta dacă adresele web, adresele de email sau numerele de telefon din textul care va fi afișat, vor fi evidențiate și activate.

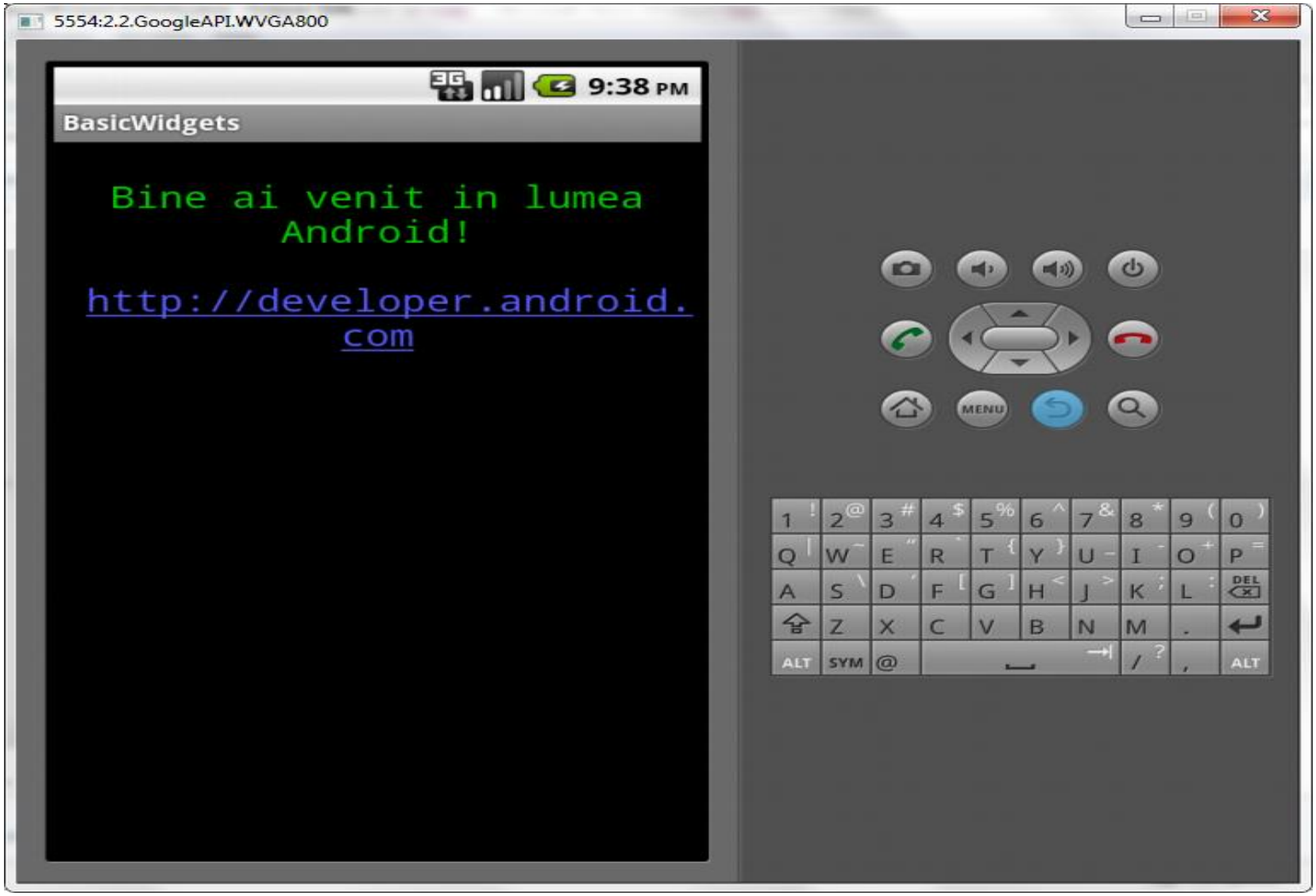
TextView – atribute in xml

```
<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="@string/bun_venit"
   android:typeface="monospace"
   android:textColor="#00cd00"
   android:textSize="20sp"
   android:autoLink="all"
   android:gravity="center"
   />
```

TextView – adaugare in mod dinamic

```
TextView label = new TextView(this);
    label.setText(R.string.bun_venit);
    label.setTextColor(Color.GREEN);
    label.setTextSize(TypedValue.COMPLEX_UNIT_SP, 20);
    label.setGravity(Gravity.CENTER);
    label.setTypeface(Typeface.MONOSPACE);
    label.setAutoLinkMask(Linkify.ALL);

    myLayout.addView(label);
```

TextView – aparitii grafice



EditText

- este o subclasă a TextView, singura facilitate față de acesta din urmă este faptul că ne permite să editățm textul pe care-l afișează;

- asimilat cu notiunea “clasica” de TextBox.

EditText - Atribute

- android:inputType, cu ajutorul acestuia specificăm tipul de date care vor fi preluate; astfel tastatura se va ajusta pentru o introducere mai ușoară a datelor. Spre exemplu daca vom da valoarea *phone,* când utilizatorul va apăsa pe căsuță va apărea o tastatură numerică;

- android:autoText, cu ajutorul acestui atribut putem seta ca sistemul să detecteze erorile gramaticale și să le corecteze;

EditText - Atribute

- android:singleLine, putem seta căsuța de text să fie expandabilă în funcție de textul introdus, sau să rămână pe o singură linie indiferent de acesta.

- android:hint, putem oferi o sugestie de completare a căsuței text atunci când este goală;

- android:digits, dacă dorim să poată fi introduse doar anumite numere, putem impune acestă limită cu ajutorul acestui atribut.

EditText - Exemple

```
<!-- Exemplu simplu, fiecare propozitie va incepe cu litera mare -->
    <TextView android:text="@string/label_simplu"
            android:layout_height="wrap_content"
            android:layout_width="match_parent" />
    <EditText android:id="@+id/etSimplu"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:capitalize="sentences"
            android:inputType="textMultiLine"
            android:gravity="top"
            android:lines="4">
    </EditText>
```

EditText – Exemple…exemple…

```
<!-- Vom oferi un hint pentru acest EditText -->
<TextView android:text="@string/label_hint"
        android:layout_height="wrap_content"
        android:layout_width="match_parent" />
<EditText android:id="@+id/etHint"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/et_label_hint"
        >
</EditText>
```

EditText – Exemple...exemple...exemple...

```
<!-- Vom configura aceasta casuta pentru
a introduce mai usor un numar de telefon -->

<TextView android:text="@string/label_telefon"
        android:layout_height="wrap_content"
        android:layout_width="match_parent" />
<EditText android:id="@+id/etTelefon"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="phone"
        >
</EditText>
```

EditText –

Exemple…exemple…exemple…☺

```
<!-- Casuta pentru introducerea unei parole -->

<TextView android:text="@string/label_parola"
        android:layout_height="wrap_content"
        android:layout_width="match_parent" />
<EditText android:id="@+id/etParola"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        >
</EditText>
```

EditText – Adaugare in mod dinamic

```
EditText etAdresa = new EditText(this);
    // setam lungimea si inaltimea
    etAdresa.setLayoutParams(new LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.MATCH_PARENT,
                        LinearLayout.LayoutParams.WRAP_CONTENT));
    // adaugam un indiciu
    etAdresa.setHint("Introduceti adresa dvs.");
    // aceasta casuta de text va fi pe mai multe linii
    // si fiecare propozitie va incepe cu majuscula
    etAdresa.setInputType(InputType.TYPE_TEXT_FLAG_MULTI_LINE
                        | InputType.TYPE_TEXT_FLAG_CAP_SENTENCES);
    // textul va fi aliniat stanga sus
    etAdresa.setGravity(Gravity.TOP);
    // casuta de text va avea 2 linii
    etAdresa.setLines(2);

    // adaug casuta de text in view
    myLayout.addView(etAdresa);
```

EditText – Folosire in practica

- De cele mai multe ori valorile din căsuțele de tip text vor fi preluate și analizate atunci când utilizatorul va apasă un buton, spre exemplu *Trimite*, dar sunt situații când dorim să efectuăm o anumită acțiune atunci când utilizatorul interacționează cu aceste căsuțe de text.

- Pentru a capta evenimentul click va trebui să folosim metoda ***setOnClickListener()*** sau pentru a capta evenimentul focus va trebui să folosim metoda ***setOnFocusChangeListener()***:

EditText – Folosire in practica

```
final EditText  etTelefon = (EditText)findViewById(R.id.etTelefon);


etTelefon.setOnFocusChangeListener(new onFocusChangeListener() {
        public void onFocusChange(View v, boolean hasFocus) {
                //preluam valoarea din casuta text
                String adresaPreluata = etTelefon.getText().toString();
                //afisam in log-ul pentru debug
                Log.d("#### InfoEc Debug ###", adresaPreluata);
        }
    }
);
```

EditText – Exemplu rezultat afisare

Button – prezentare generala

- 3 tipuri de butoane:
  - Button;
  - ImageButton;
  - ToggleButton.

Printre cele mai des folosite atribute ale unui buton se numără cele pentru specificarea id-ului, lățimii, înălțimii și cel pentru textul care va fi afișat în cadrul său.

Button – atribute specifice

```
<Button android:id="@+id/btnApasa"
        android:text="@string/label_apasa"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:textColor="@color/button_text_color">
    </Button>
```

Am creat in prealabil fisierul colors.xml in directorul res/values.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
            <color name="button_text_color">#ffcc0005</color>
</resources>
```

Button – folosire uzuala

```
// creez butonul pe baza declaratiilor din xml
Button btnApasa = (Button) findViewById(R.id.btnApasa);

btnApasa.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
                // afisez un mesaj atunci cand utilizatorul apasa pe buton
                Toast.makeText(
                CtrlGraficeButtonActivity.this,
                "De ce ma apesi? Tu nu stii ca sunt mic si ca ma doare?“,
                                Toast.LENGTH_LONG).show();
                                }
        });
```

ImageButton – model de implementare

Putem specifica imaginea fie din fișierul xml după cum urmează:

```
<ImageButton android:id="@+id/imageButton"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:src="@drawable/feaa"
      />
```

Sau putem specifica imagine în mod dinamic folosind metoda ***setImageResource()***:

```
ImageButton imageButton = (ImageButton)findViewById(R.id.imageButton);
      imageButton.setImageResource(R.drawable.feaa);
```

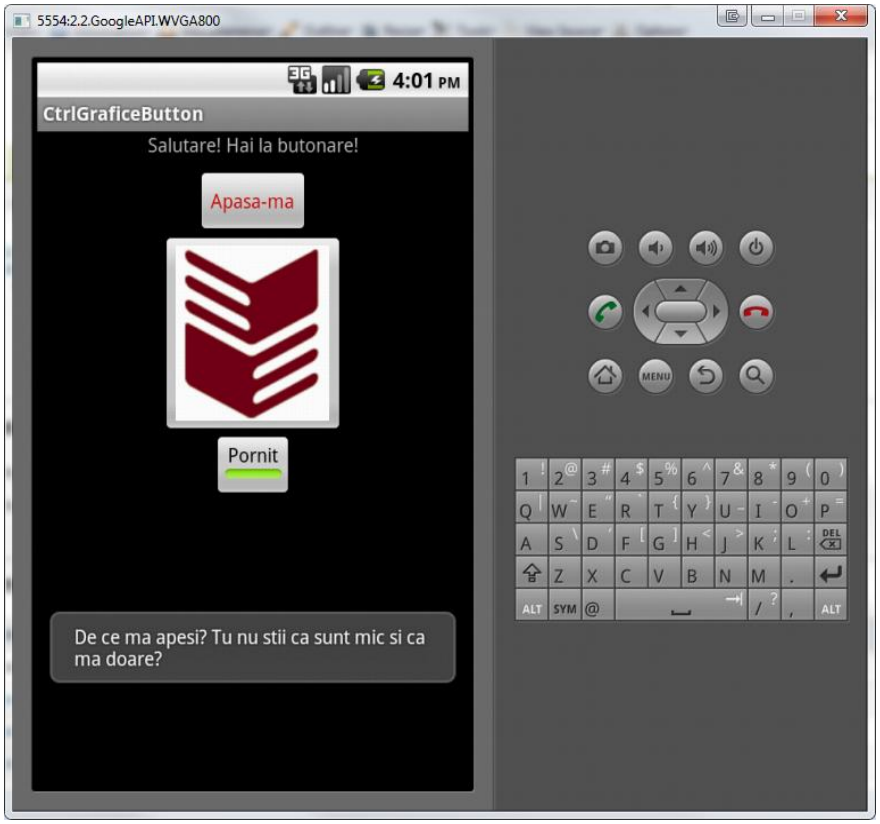
ToggleButton – prezentare generala

- Acest tip de buton se aseamănă mai mult cu un checkbox, deoarece este un buton cu două stări, On/Off.
- În funcție de starea în care se află afișează un led de culoare verde pentru **On** sau gri pentru **Off**.

- De notat este faptul că se poate particulariza textul afișat pentru fiecare stare în parte cu ajutorul atributelor *android:textOn* și *android:textOff*:

ToggleButton – implementare

```
<ToggleButton android:id="@+id/toggleButton"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:textOn="Pornit"
              android:textOff="Oprit"
              android:layout_gravity="center"  />
```



Checkbox – prezentare generala

- Acest widget are două stări: bifat și nebifat.
- Atunci când vom apăsa pe acesta stările se vor interschimba automat.

- Deoarece TextView este ”strămoșul” său, moștenește toate metodele sale.

- Din cod putem controla stările cu ajutorul metodelor ***setChecked()*** sau ***toggle()***; de asemenea putem obține starea curentă cu ajutorul metodei ***isChecked()*.**

Checkbox – o implementare

Vom crea 3 checkbox-uri in main.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
        <TextView android:layout_width="fill_parent"
                  android:layout_height="wrap_content"
                  android:text="@string/alege_culori" />
        <CheckBox android:id="@+id/ckbRosu"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Rosu"          />
        <CheckBox android:id="@+id/ckbGalben"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Galben" />
        <CheckBox android:id="@+id/ckbAlbastru"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Albastru" />
</LinearLayout>
```

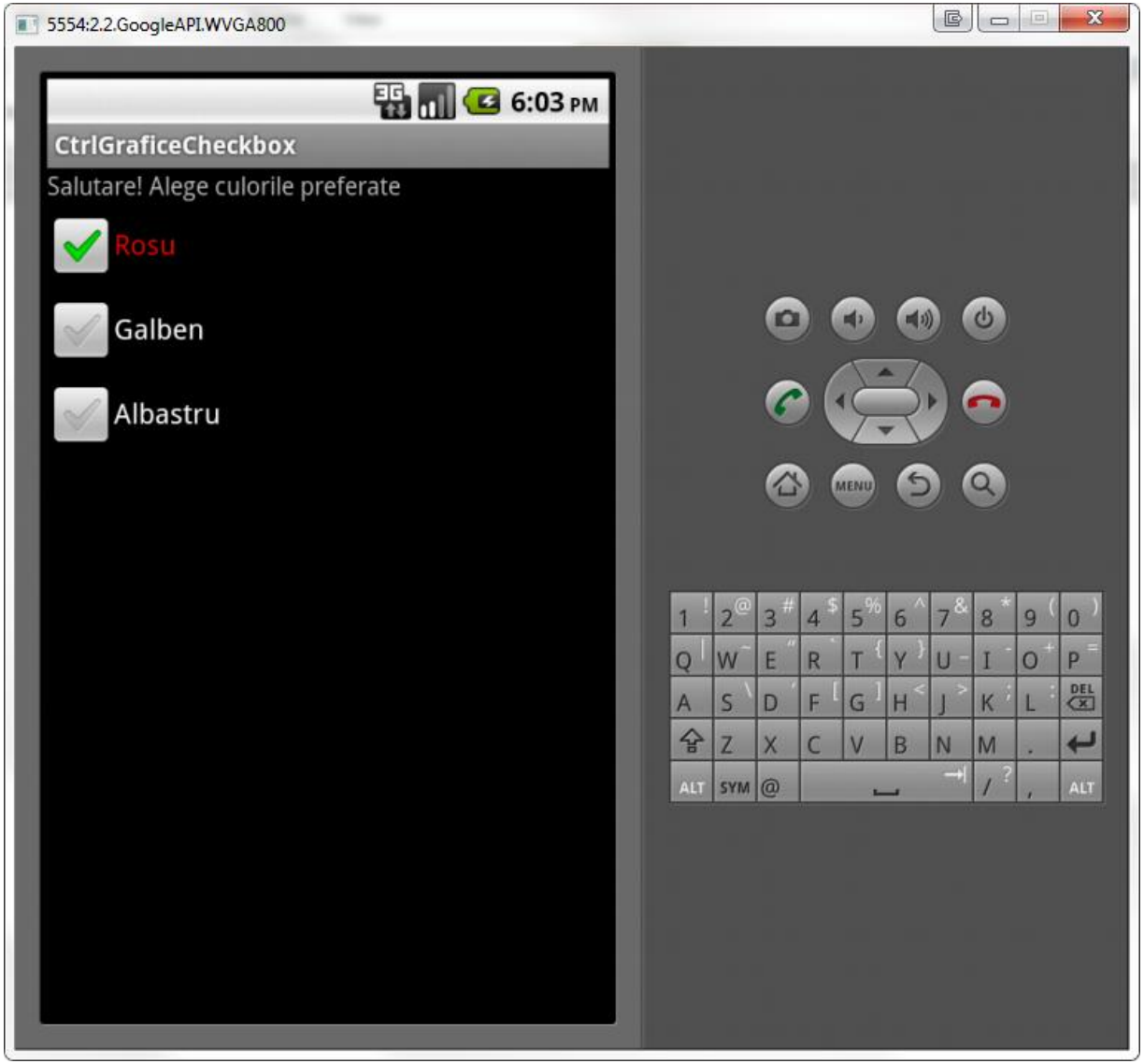
Checkbox – o implementare

Dacă utilizatorul bifează una dintre culorile listate vom evidenția textul aferent.

```
public class CtrlGraficeCheckboxActivity extends Activity implements
OnCheckedChangeListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        CheckBox ckbRosu = (CheckBox)findViewById(R.id.ckbRosu);
        ckbRosu.setOnCheckedChangeListener(this);

        CheckBox ckbAlbastru = (CheckBox)findViewById(R.id.ckbAlbastru);
        ckbAlbastru.setOnCheckedChangeListener(this);

        CheckBox ckbGalben = (CheckBox)findViewById(R.id.ckbGalben);
        ckbGalben.setOnCheckedChangeListener(this);
    }
        @Override
        public void onCheckedChanged (CompoundButton buttonView, boolean isChecked) {
                if(isChecked){
                        buttonView.setTextColor(Color.RED);
                } else {
                        buttonView.setTextColor(Color.WHITE);
                }
        }
}
```

Checkbox – o implementare

Rezultatul:



RadioButton – prezentare generala

- Foarte asemănător cu widgetul Checkbox, RadioButton are ca ”strămoș” TextView, astfel ca putem particulariza aceste elemente folosindu-ne de toate metodele puse la dispoziție de TextView.

- Acest widget are de asemenea 2 stări bifat/nebifat.

- Putem folosi metoda ***isChecked()***, pentru a verifica dacă starea curentă a unui RadioButton și ***toggle()*** pentru a schimba starea.

RadioButton – prezentare generala

- De cele mai multe ori butoanele de tip radio sunt folosite în cadrul unui **RadioGroup**. Când mai multe butoane sunt plasate în cadrul unui RadioGrup, doar unul dintre ele poate fi bifat la un moment dat.

- Dacă atribuim un id grupului de butoane radio avem acces la următoarele metode:
  - *check()* – putem verifica starea unui anumit RadioButton din cadrul grupului, spre exemplu *radioGrup.check(R.id.radioButton1)*;
  - *clearCheck* – debifăm toate elementele din RadioGroup;
  - *getCheckedRadioButtonId()* – întoarce id-ul elementului bifat din cadrul RadioGroup-ului. Daca nici un element nu este bifat va întoarce -1.

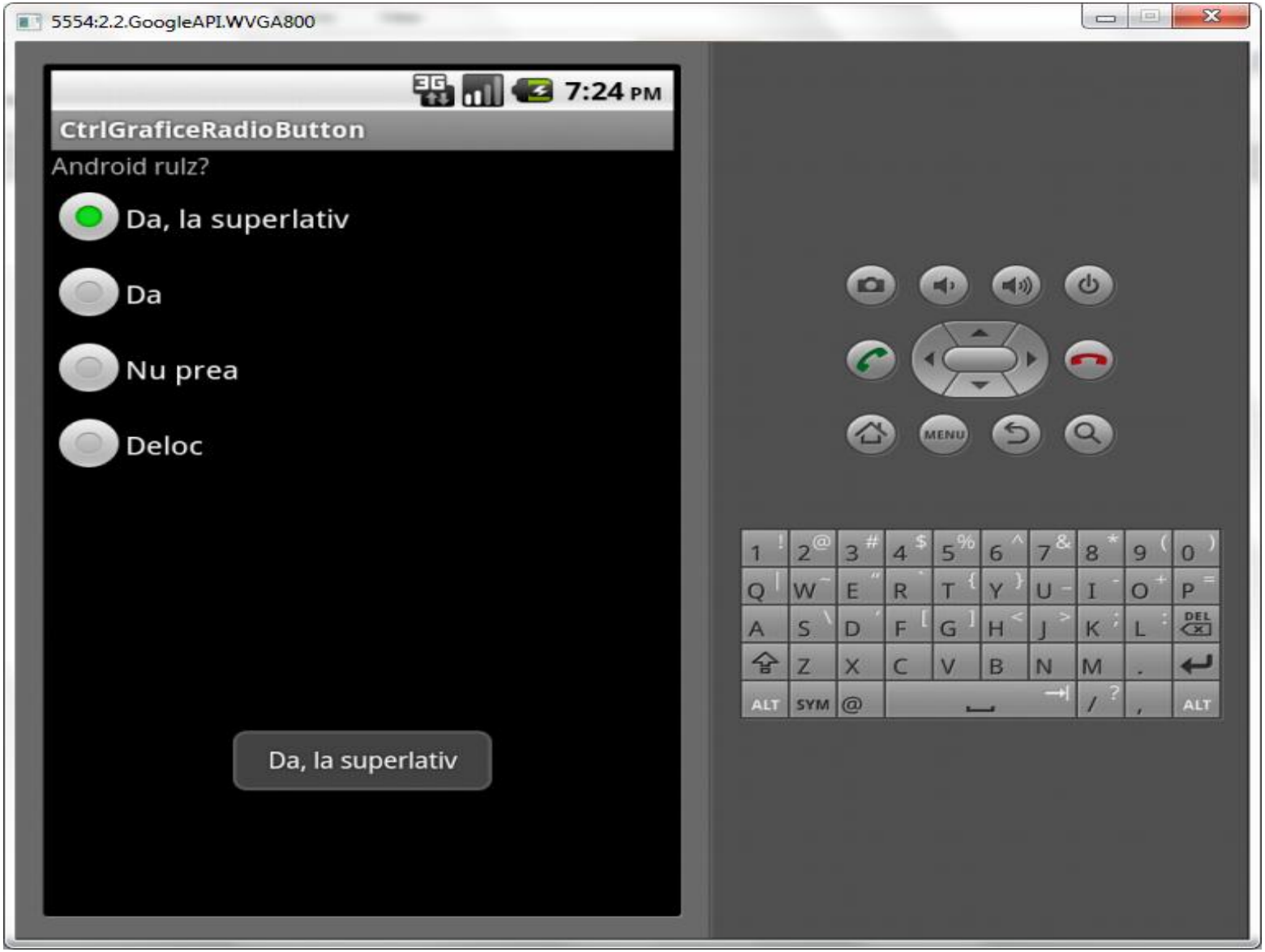
RadioButton – implementare (model)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical" android:layout_width="fill_parent"
        android:layout_height="fill_parent">
        <TextView android:layout_width="fill_parent"
                  android:layout_height="wrap_content"
                  android:text="@string/intrebare"
        />
        <RadioGroup android:id="@+id/rdGroup"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content">
                  <RadioButton android:id="@+id/rdButton1"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:text="Da, la superlativ" />
                  <RadioButton android:id="@+id/rdButton2"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:text="Da" />
                  <RadioButton android:id="@+id/rdButton3"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:text="Nu prea" />
                  <RadioButton android:id="@+id/rdButton4"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:text="Deloc" />
        </RadioGroup>
</LinearLayout>
```

RadioButton – implementare (model)

```
public class CtrlGraficeRadioButtonActivity extends Activity {
        /** Called when the activity is first created. */
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                RadioGroup rdGroup = (RadioGroup) findViewById(R.id.rdGroup);
                rdGroup.setOnCheckedChangeListener(new OnCheckedChangeListener() {
                        @Override
                        public void onCheckedChanged(RadioGroup rd, int idRadioBtn) {
                                RadioButton rdBtn;
                                String textSelectat = "Nu ai selectat nimic";
                                // iterez printre toate elementele grupului
                                // pentru a verifica care buton a fost selectat
                                for (int j = 0; j < rd.getChildCount(); j++) {
                                        rdBtn = (RadioButton) rd.getChildAt(j);
                                // daca id-ul curent este acelasi cu cel trimis ca
                                // paramentru retin textul
                                        if (rdBtn.getId() == idRadioBtn)
                                        textSelectat = rdBtn.getText().toString();
                                }
                                // afisez un mesaj atunci cand utilizatorul apasa pe buton
        Toast.makeText(CtrlGraficeRadioButtonActivity.this,textSelectat,Toast.LENGTH_SHORT).show();
                        }
                });
        }
}
```

RadioButton – rezultat☺



ListView – prezentare generala

- Android oferă mai multe controale pentru afișarea listelor. Ce mai des întâlnit și cel mai folositor este ListView. Cu ajutorul acestuia putem afișa **liste verticale de dimensiuni foarte mari.**

ListView – implementare

- Clasa principală care a fost generată la crearea proiectului poate extinde ***ListActivity***. Acesta găzduiește un ListView care poate fi populat cu date din diferite surse, fie un array fie un Cursor care conține rezultatele unei interogări SQL.

- *ListActivity* ne va insera o listă care se va extinde pe tot ecranul.

- În cazul în care dorim să particularizăm layout-ul vom introduce în fișierul XML un element ListView cu atributul **android:id="@android:id/list"** astfel încât *ListActivity* să știe care este elementul care va afișa lista.

ListView – implementare xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >
        <TextView android:id="@+id/judetSelectat"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="#ffffff00"
                android:textColor="#ff000000"
                android:textSize="16dp"
                android:gravity="center"
                android:padding="5dp"
                />
        <ListView android:id="@android:id/list"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:drawSelectorOnTop="false"
        />
</LinearLayout>
```

ListView – implementare java

```
public class CtrlGraficeListViewActivity extends ListActivity {
        final static String[] judete = new String[] { "Bucuresti", "Alba", "Arges",
                "Arad", "Bacau", "Bihor", "Bistrita-Nasaud", "Botosani", "Brasov",
                "Braila", "Buzau", "Caras-Severin", "Calarasi", "Cluj",
                "Constanta", "Covasna", "Dambovita", "Dolj", "Galati", "Giurgiu",
                "Gorj", "Harghita", "Hunedoara", "Ialomita", "Iasi", "Ilfov",
                "Maramures", "Mehedinti", "Mures", "Neamt", "Olt", "Prahova",
                "Satu Mare", "Salaj", "Sibiu", "Suceava", "Teleorman", "Timis",
                "Tulcea", "Vaslui", "Valcea", "Vrancea" };
        TextView judetSelectat;

        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                setListAdapter(new ArrayAdapter<String>(this,
                                android.R.layout.simple_list_item_1, judete));
                judetSelectat = (TextView) findViewById(R.id.judetSelectat);
        }

        public void onListItemClick (ListView lv, View v, int position, long id) {
                judetSelectat.setText(judete[position]);
        }
}
```

ListView – rezultat grafic ☺

ListView – implementare (ceva mai) extinsa

Dacă dorim să modificăm modul de afișare al listei și **să permită selecție multiplă** va trebui să facem următoarele modificări în codul nostru:

- în metoda *onCreate()* să înlocuim layout-ul *simple_list_item_1* cu ***simple_list_item_multiple_choice*** și să adăugăm următoarea linie de cod:

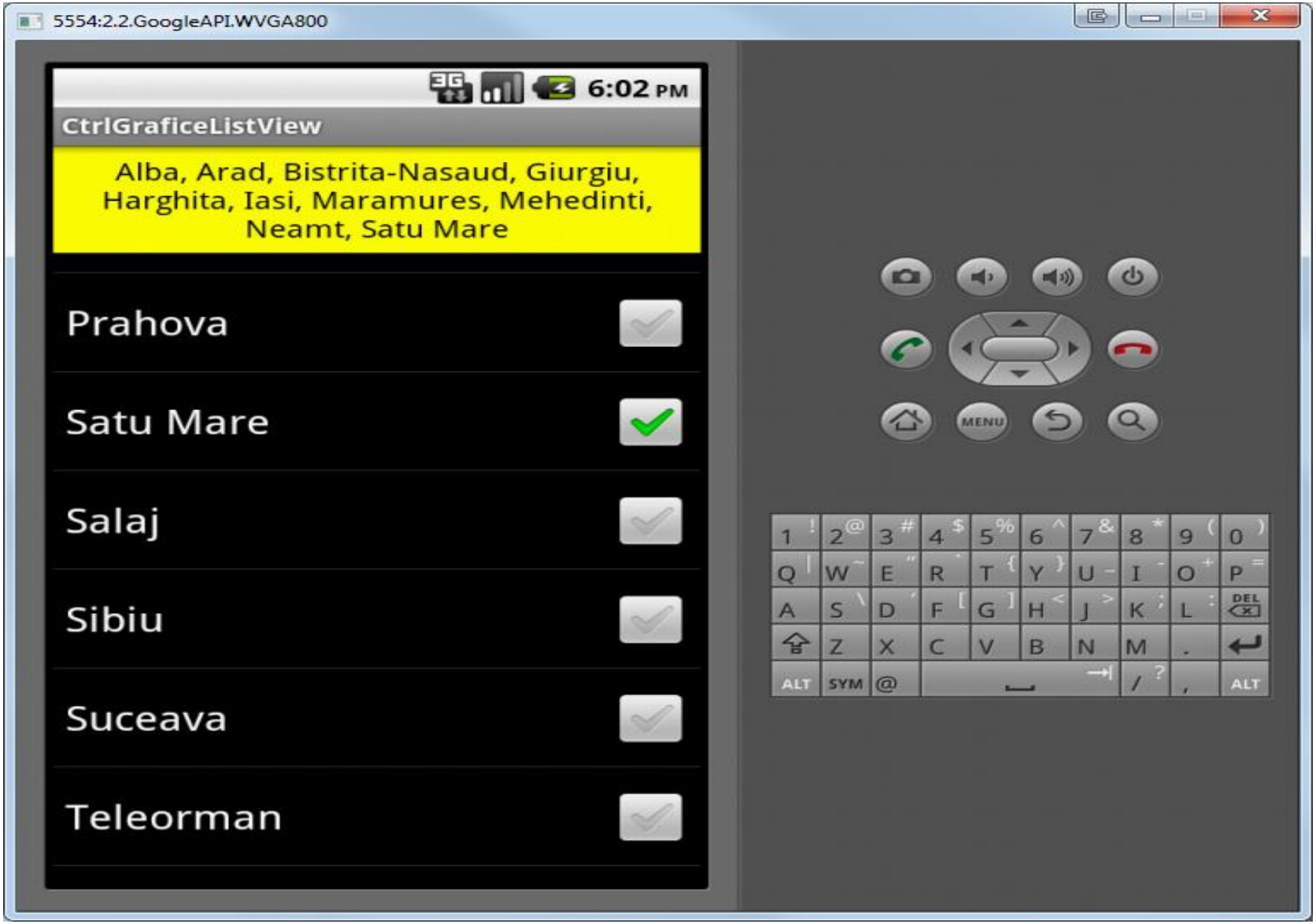
getListView().setChoiceMode(ListView.*CHOICE_MODE_MULTIPLE*);

ListView – implementare (ceva mai) extinsa

Codul java va suferi urmatoarele modificari (parcurgem lista de "selectii":

```
public void onListItemClick (ListView parent, View v, int position, long id) {
            SparseBooleanArray arrChecked = parent.getCheckedItemPositions();
            StringBuilder sb = new StringBuilder();
            String delim = "";
            for (int i = 0; i < arrChecked.size(); i++) {
                    if (arrChecked.valueAt(i)) {  //adica daca este selectat
                            sb.append(delim).append(

    parent.getItemAtPosition(arrChecked.keyAt(i)));
                            delim = ", ";
                    }
            }
            judetSelectat.setText(sb);
    }
```

ListView – un rezultat pentru selectie multipla



Alte implementari…

…la laborator ☺