ANDROID platforma
izgradnja korisničkog interfejsa
unos, meniji, dialog box-ovi

Mobilni i distribuirani informacioni sistemi

Mr Bratislav Predić

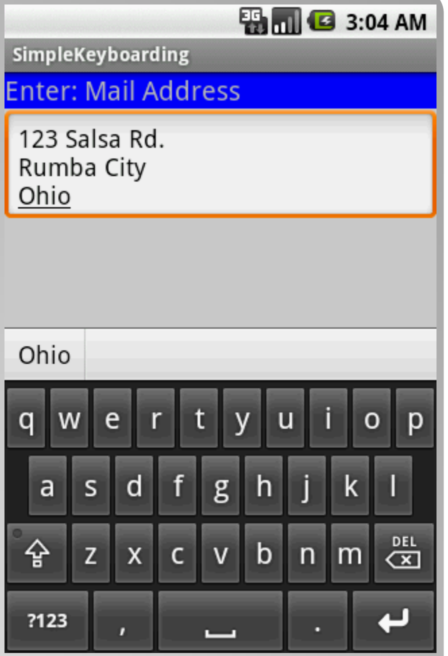
2012. godina

Input Method Framework (IMF)

- Android od verzije 1.5 uvodi Input Method Framework (IMF)
- Ideja je da se apstrahuju različiti metodi unosa na različitim uređajima
  - Fizičke i virtuelne tastature
  - Prepoznavanje govora
  - Prepoznavanje rukopisa
  -

Input Method Framework (IMF)

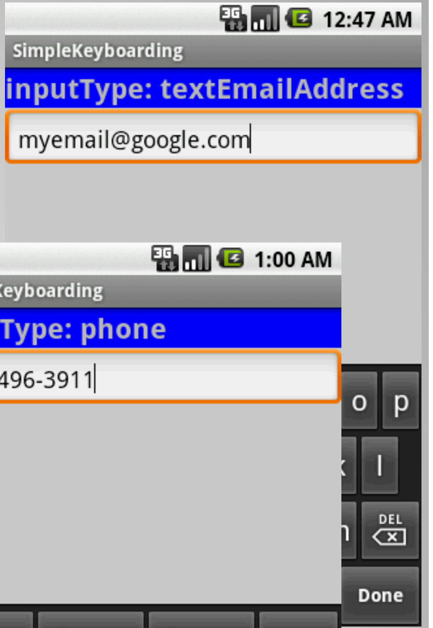
- IMF "zna" za dostupan hardver na uređaju i njegov trenutni status
- Ukoliko ne postoji hardverska tastatura kada se aktivira widget koji očekuje unos prikazaće se input method editor (IME)
- Može se definisati očekivani tip unosa za widget
  - U XML layout-u: *android:inputType="…"*
  - U kodu: *editTextBox.setRawInputType(int)*

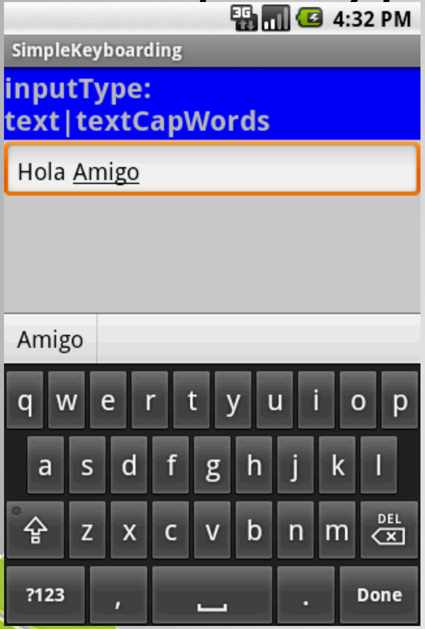


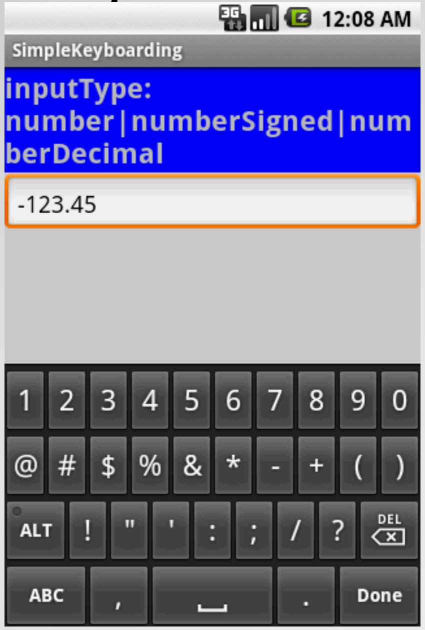
123 Salsa Rd.
Rumba City
Ohio

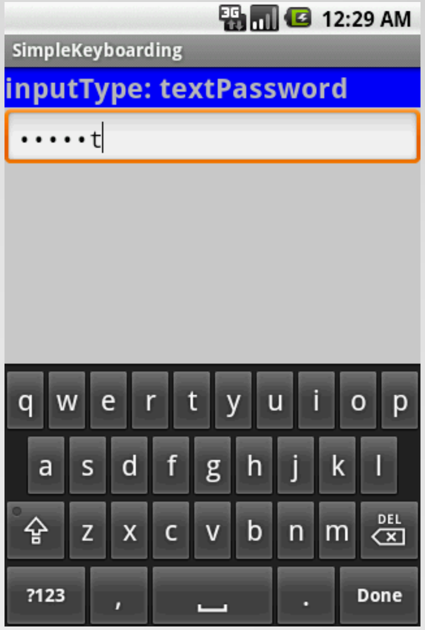
Input Method Framework (IMF)

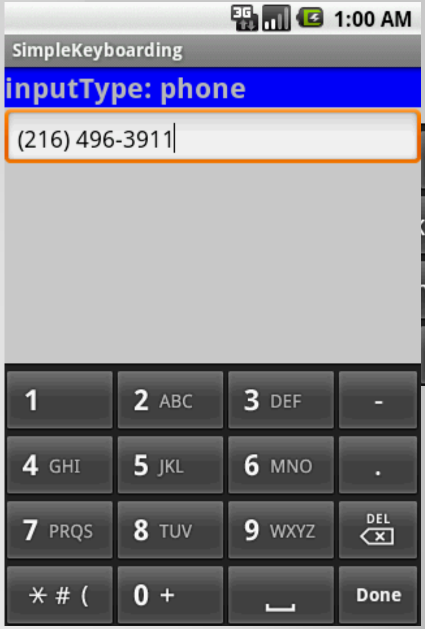
- InputType vrednosti
  - *inputType: text|textCapWords*
  - *inputType="number|numberSigned|numberDecimal"*
  - *inputType="textPassword"*
  - *inputType="textEmailAddress"*
  - *inputType="phone"*


12:47 AM
SimpleKeyboarding
inputType: textEmailAddress
myemail@google.com


4:32 PM
SimpleKeyboarding
inputType: text|textCapWords
Hola Amigo
Amigo


12:08 AM
SimpleKeyboarding
inputType: number|numberSigned|numberDecimal
-123.45


12:29 AM
SimpleKeyboarding
inputType: textPassword
••••••t


1:00 AM
SimpleKeyboarding
inputType: phone
(216) 496-3911

Input Method Framework (IMF)

- Ukoliko za widget želimo da isključimo virtuelnu tastaturu
  - Možemo podesiti input type

```
txtBox.setInputType( InputType.TYPE_NULL);
```

  - Možemo postaviti i prazan touch listener

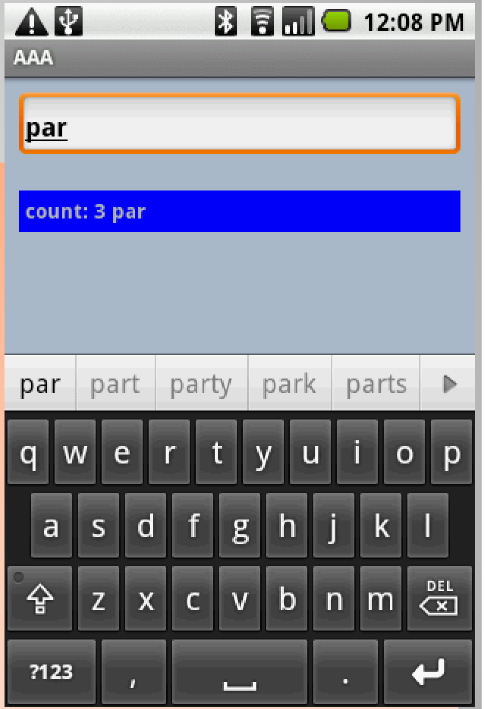
```
txtBox.setOnTouchListener(new OnTouchListener() {
  @Override
  public boolean onTouch(View arg0, MotionEvent arg1) {
    // return true to consume the touch event without
    // allowing virtual keyboard to be called
    return true;
  }
});
```

  - Virtuelna tastatura se zatvara *Back* tasterom ili

```
InputMethodManager imm=
   (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
imm.hideSoftInputFromWindow(theEditTextField.getWindowToken(),0);
```

Input Method Framework (IMF)

- Možemo pratiti izmene u EditBox-u tako što uz njega registrujemo *TextWatcher*
- Metode *TextWatcher-a* su:
  - *public void afterTextChanged(Editable theWatchedText)*
  - *public void beforeTextChanged(…)*
  - *public void onTextChanged(…)*

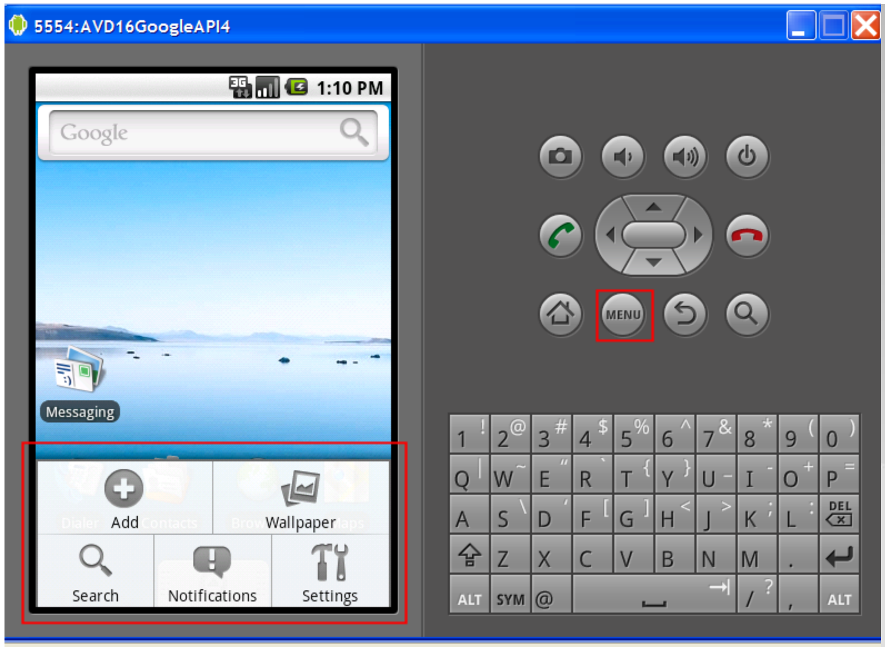


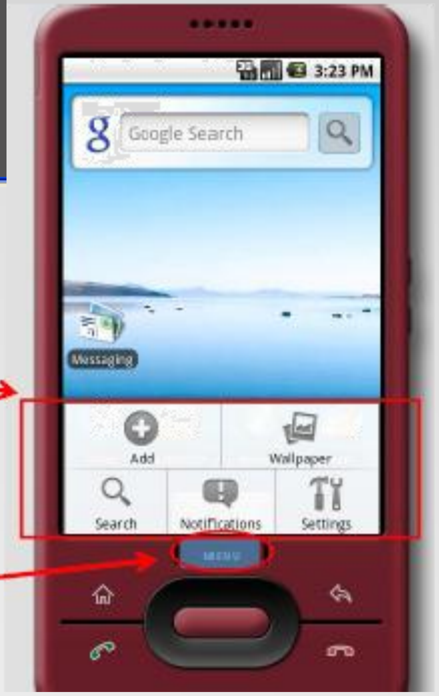
```
txtInput.addTextChangedListener(new TextWatcher() {
  public void afterTextChanged(Editable theWatchedText) {
    String msg= "count: " +
         txtInput.getText().toString().length() + " " +
         theWatchedText.toString();
    txtMsg.setText( msg);
  }
  public void beforeTextChanged(CharSequencearg0,
                         intarg1, intarg2, intarg3) {
    Toast.makeText(getApplicationContext(),
         "BTC " + arg0, 1).show();
  }
  public void onTextChanged(CharSequencearg0, intarg1, intarg2, intarg3) {
    Toast.makeText(getApplicationContext(), "OTC " + arg0, 1).show();
  }
}); //addTextChangedListener
```

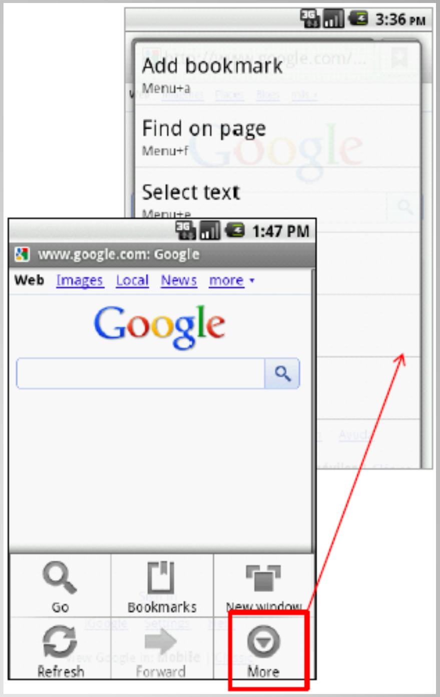
Android sistem menija

- Meni proširuje funkcionalnost aplikacije tako što prikazuje dodatne operacije na malom preklapajućem panelu
- Na Androidu postoje dva tipa menija
  - *Options menu*
    Ovaj meni se aktivira pritiskom na hardversko Menu dugme na telefonu
  - *Context menu*
    Ovaj meni se aktivira tap-and-hold akcijom nad widget-om za koji je meni vezan
- Options menu ima ograničen broj stavki koje može da prikaže

Android sistem menija





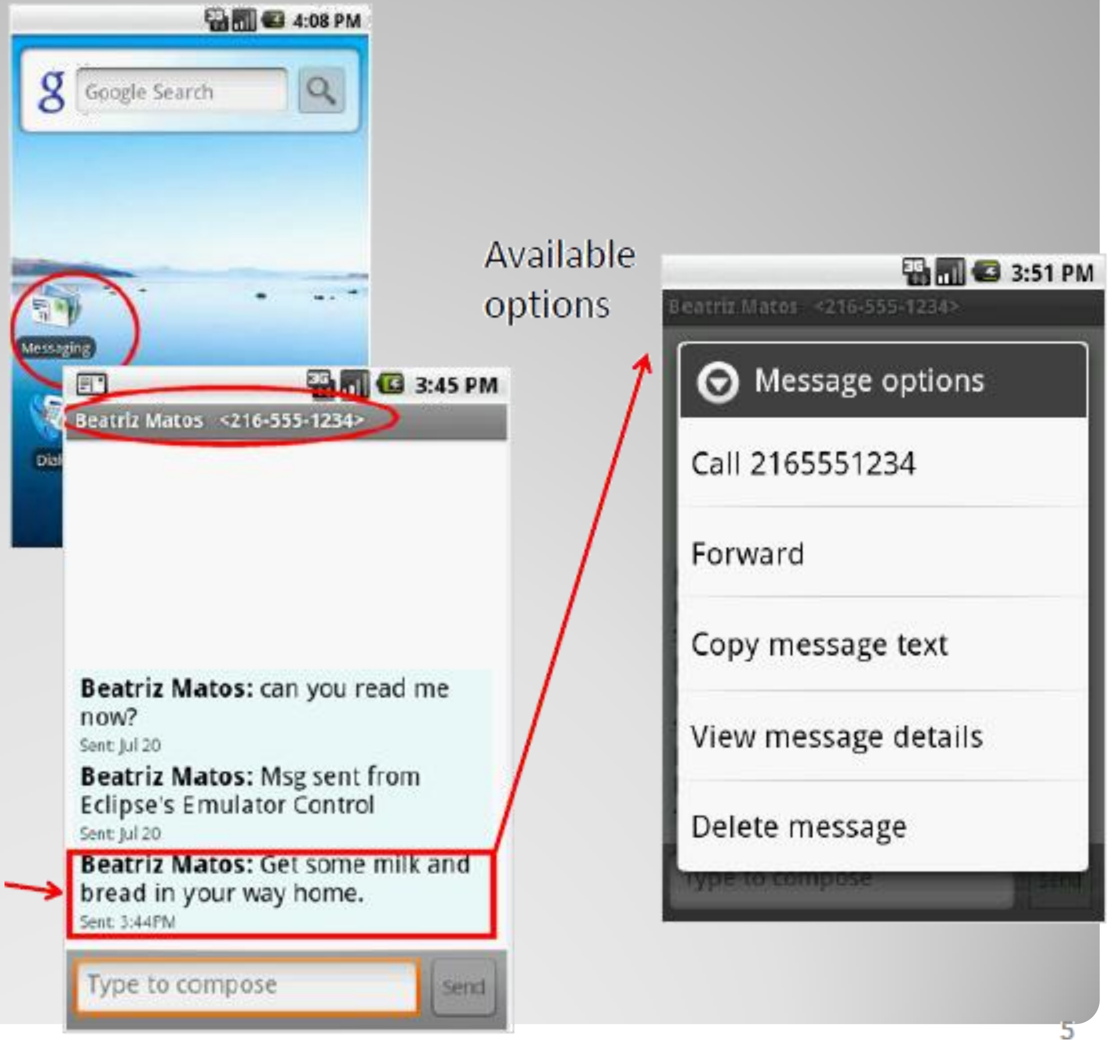


Options available in this context

Press **Menu** button

Android sistem menija

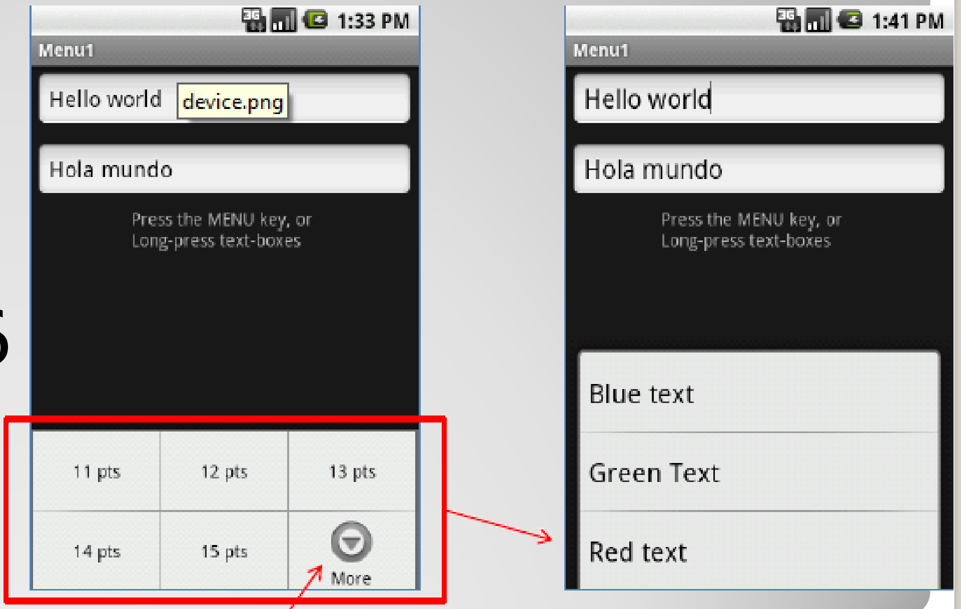
- Primer kontekstnog menija
  - SMS aplikacija



Beatriz Matos: can you read me now?
Sent: Jul 20
Beatriz Matos: Msg sent from Eclipse's Emulator Control
Sent: Jul 20
Beatriz Matos: Get some milk and bread in your way home.
Sent: 3:44PM

Android sistem menija

- Obe vrste Android menija mogu da sadrže
  - Tekst
  - Ikonu
  - Radio dugme
  - Check polje
  - Podmeni
  - Prečice (shortcuts)
- Meni prikazuje maksimalno 6 opcija
- Ukoliko ima više od 6 opcija automatski se dodaje *More* polje

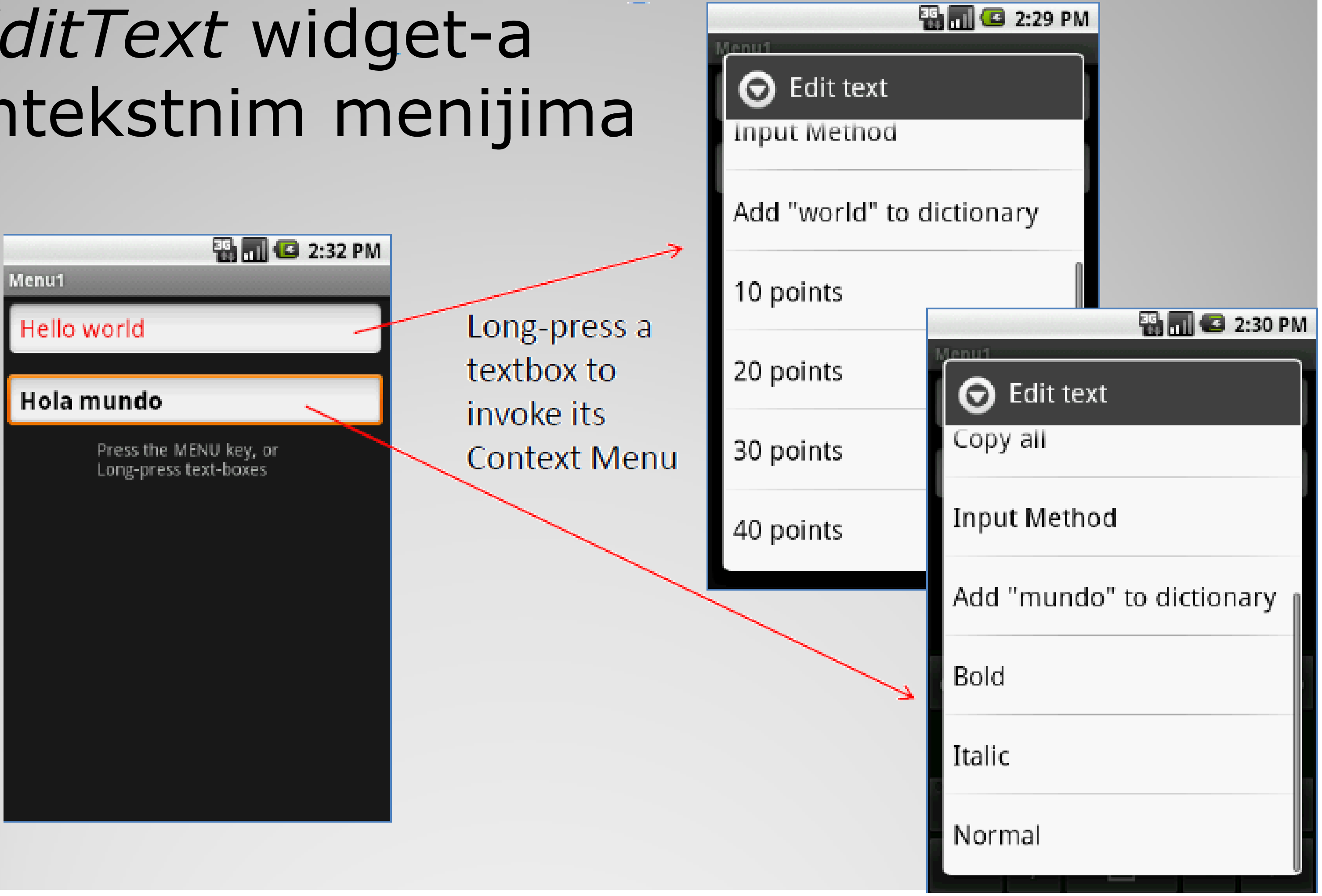


Press the MENU key, or Long-press text-boxes

Press the MENU key, or Long-press text-boxes

Primer za menije

- Uvezaćemo obe vrste menija
- Dva *EditText* widget-a sa kontekstnim menijima



Long-press a textbox to invoke its Context Menu

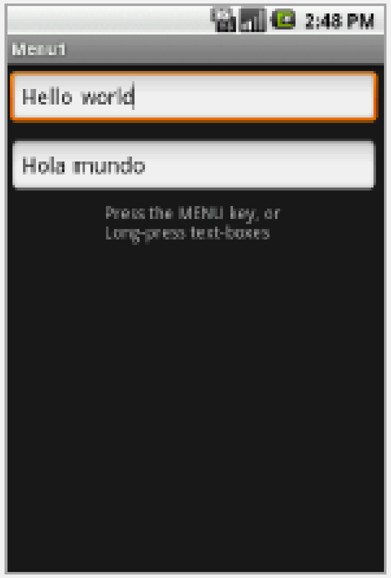
Primer za menije

```
<?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"  >

<EditText
     android:id="@+id/etMessage1"
     android:text="Hello world"
     android:layout_width="fill_parent"
     android:layout_height="wrap_content"
     android:layout_margin = "5dp" />

<EditText
     android:id="@+id/etMessage2"
     android:text="Hola mundo"
     android:layout_width="fill_parent"
     android:layout_height="wrap_content"
     android:layout_margin = "5dp" />

<TextView
     android:text="Press the MENU key, or \nLong-press text-boxes"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_gravity="center" />
</LinearLayout>
```



2:48 PM

Menu1

Hello world

Hola mundo

Press the MENU key, or
Long-press text-boxes

Primer za menije

```
import android.app.Activity;
…
Public class Menu1Act1 extends Activity {
  EditText etMessage1;
  EditText etMessage2;
  Integer[] arrayPointSize = {10, 20, 30, 40, 50};

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    etMessage1= (EditText)findViewById(R.id.etMessage1);
    etMessage2= (EditText)findViewById(R.id.etMessage2);

    // you may register an individual context menu for each view
    registerForContextMenu(etMessage1);
    registerForContextMenu(etMessage2);
  } //onCreate
...
```

Primer za menije

```
// set the option menu for the current activity
@Override
publi cboolean onCreateOptionsMenu(Menu menu) {
  // only one Option menu per activity
  populateMyFirstMenu(menu);
  return super.onCreateOptionsMenu(menu);
}

// detect what view is calling and create its context menu
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
      ContextMenuInfo menuInfo) {
  super.onCreateContextMenu(menu, v, menuInfo);
  // decide what context menu needs to be made
  if(v.getId() == etMessage1.getId()){
    // create a menu for etMessage1 box
    populateMyFirstMenu(menu);
  }
  if(v.getId() == etMessage2.getId()){
    // create a menu for etMessage2 box
    populateMySecondMenu(menu);
  }
} //onCreateContextMenu
```

Primer za menije

```
private void populateMyFirstMenu(Menu menu){
  int groupId= 0; int order= 0;
  //arguments: groupId, optionId, order, title
  menu.add(groupId, 1, 1, "10 points");
  menu.add(groupId, 2, 2, "20 points");
  menu.add(groupId, 3, 3, "30 points");
  menu.add(groupId, 4, 4, "40 points");
  menu.add(groupId, 5, 5, "50 points");
  menu.add(groupId, 6, 8, "Red text");
  menu.add(groupId, 7, 7, "Green Text");
  menu.add(groupId, 8, 6, "Blue text");
} //populateMyMenu

private void populateMySecondMenu(Menu menu){
  int groupId= 0; int order= 0;
  //arguments: groupId, optionId, order, title
  menu.add(groupId, 9, 1, "Bold");
  menu.add(groupId, 10, 2, "Italic");
  menu.add(groupId, 11, 3, "Normal");
}//populateMySecondMenu
```

Primer za menije

```
// called whenever an item in your context menu is selected
@Override
public boolean onContextItemSelected(MenuItem item) {
  return(applyMenuOption(item) ||
         super.onContextItemSelected(item) );
}

// called whenever an item in your options menu is selected
@Override
public boolean onOptionsItemSelected(MenuItem item) {
  return(applyMenuOption(item) ||
         super.onOptionsItemSelected(item) );
}
```

- Meniji se koriste za promenu veličine, boje i stila za tekst
- Event observer-i koji vraćaju *boolean* vraćaju *true* ako je event obrađej i *false* ako se event prosleđuje dalje

Primer za menije

```
// apply the action associated to selected item
private boolean applyMenuOption(MenuItem item){
  int menuItemId = item.getItemId(); // 1, 2, 3, ...11
  String strMsg2 = etMessage2.getText().toString();
  if(menuItemId<= 5) {
    // first five option are for setting text size
    int newPointSize= arrayPointSize[menuItemId-1];
    etMessage1.setTextSize(newPointSize);
    etMessage2.setTextSize(newPointSize);
  }else{
    // either change color on box text1 or style on text2
    if(menuItemId== 6){
      etMessage1.setTextColor(color.background_dark| Color.RED);
      etMessage1.setTextColor(0xffff0000); // red
    } else if(menuItemId== 7)
      etMessage1.setTextColor(0xff00ff00); // green
    else if(menuItemId== 8)
      etMessage1.setTextColor(0xff0000ff); // blue
    else if(menuItemId== 9)
      etMessage2.setText(beautify(strMsg2, "BOLD")); //bold
    else if(menuItemId== 10)
      etMessage2.setText(beautify(strMsg2, "ITALIC")); //italic
    else if(menuItemId== 11)
      etMessage2.setText(beautify(strMsg2, "NORMAL")); //normal
  }
  return false;
} //applyMenuOption
```

Primer za menije

```
   // changing text style using HTML formatting
   // Spanned is text to which you could add formatting features

   private Spanned beautify (String originalText, String selectedStyle){
     Spanned answer = null;
     if(selectedStyle.equals("BOLD"))
       answer = Html.fromHtml("<b>"+ originalText+"</b");
     else if(selectedStyle.equals("ITALIC"))
       answer = Html.fromHtml("<i>"+ originalText+"</i>");
     else if(selectedStyle.equals("NORMAL"))
       answer = Html.fromHtml("<normal>"+ originalText+"</normal");
     return answer;
   } //beautify
 } //Menu1Act1
```

- **Korak 1:** *registerForContextMenu(theWidget)* Registrujemo widget-e koji imaju vezan context menu
- **Korak 2:** *onCreateContextMenu(…)* Popunjavamo meni. Parametri: meni, *View* sa kojim je povezan i *ContextMenuInfo*

Primer za menije

- *onCreateContextMenu(…)*
  Poziva se svaki put kada ima potrebe da se prikaže kontekstni meni
- Za razliku od options menija kontekstni meni se uništava svaki put kad se izabere opcija ili se meni otkaže
- Kako bi smo uhvatili koja stavka kontekstnog menija je izabrana implementiramo metodu
  - *onContextItemSelected()*
- Ovo je metoda aktivnosti

Primer za menije

- U našem primeru
  - *onOptionsItemSelected()* – za options meni
  - *onContextItemSelected()* – za kontekstni meni
- Prosleđujemo kontrolu drugoj metodi

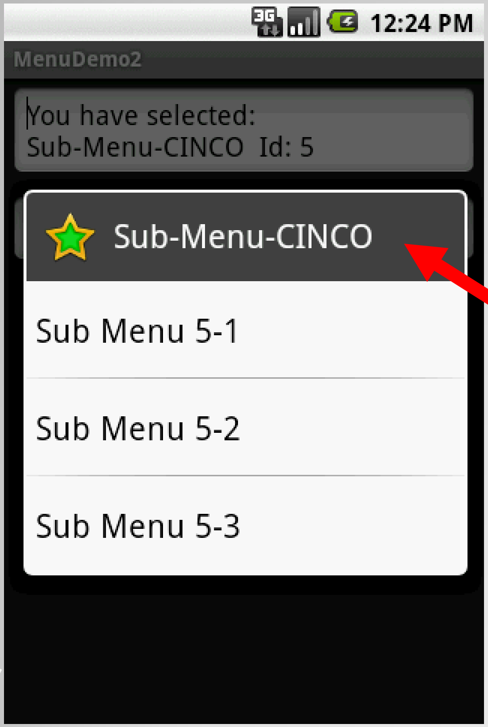
```
@Override
public boolean onOptionsItemSelected(MenuItemitem) {
  return(applyMenuChoice(item) );
}

@Override
public boolean onContextItemSelected(MenuItemitem) {
  return(applyMenuChoice(item));
}
```

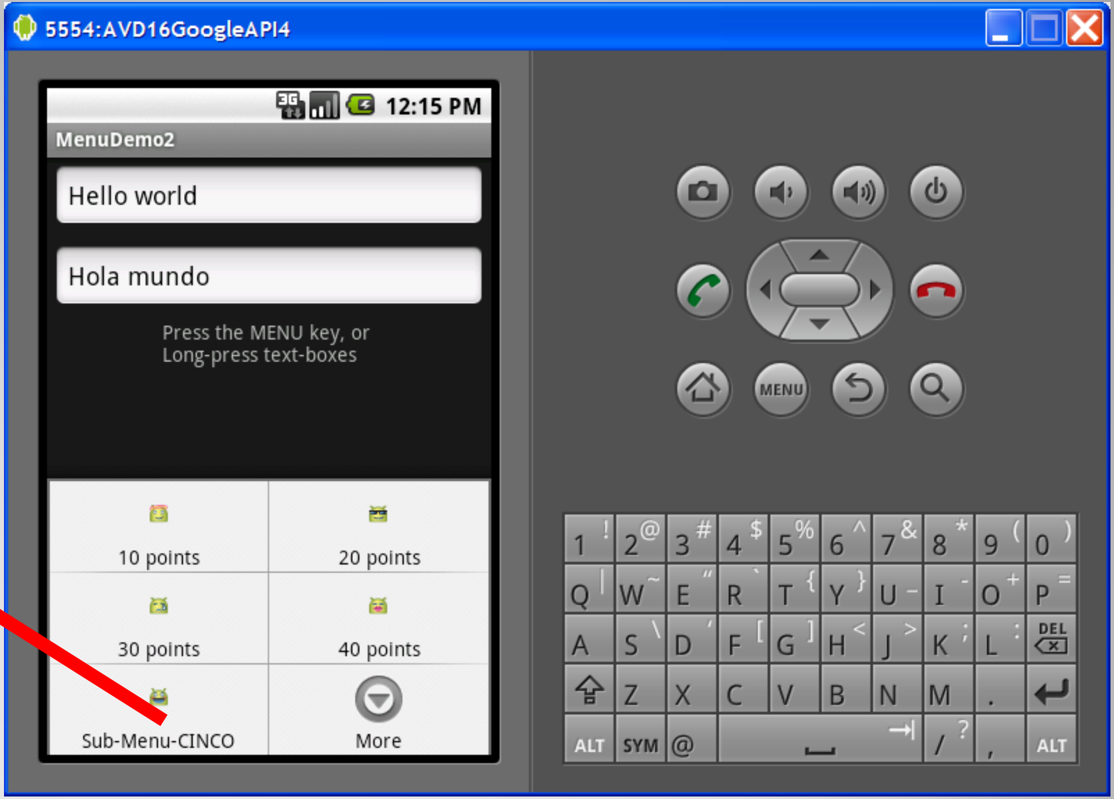
- Centralizovana obrada izbora stavke iz bilo kog menija

Primer za menije - proširenje

- Proširenje primera uvodi
  - Ikone
  - Shortcut-ove
  - Podmeni



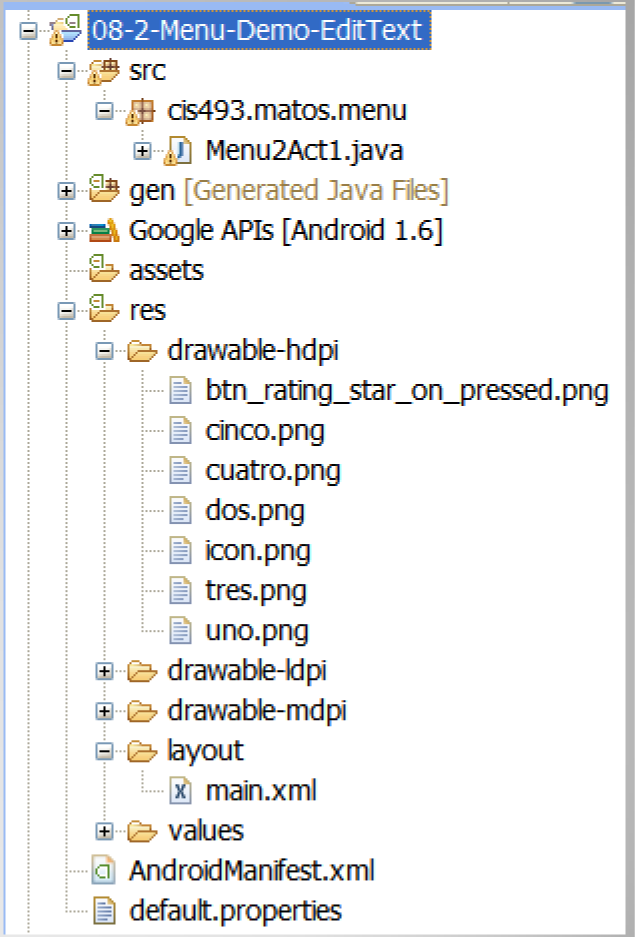
You have selected:
Sub-Menu-CINCO Id: 5



Press the MENU key, or
Long-press text-boxes

Primer za menije - proširenje

- Treba dodati ikone u resurse (za svaku gustinu piksela)
- Izmene u menju se rade u metodi
  - *populateMyFirstMenu()*
- Izbacujemo originalnu stavku 5
- Dodajemo ikone i shortcut-ove



```
08-2-Menu-Demo-EditText
  src
    cis493.matos.menu
      Menu2Act1.java
  gen [Generated Java Files]
  Google APIs [Android 1.6]
  assets
  res
    drawable-hdpi
      btn_rating_star_on_pressed.png
      cinco.png
      cuatro.png
      dos.png
      icon.png
      tres.png
      uno.png
    drawable-ldpi
    drawable-mdpi
    layout
      main.xml
    values
  AndroidManifest.xml
  default.properties
```

Primer za menije - proširenje

- Izmene u metodi *populateFirstMenu*

```
private void populateMyFirstMenu(Menu menu){
     int groupId = 0;
     //arguments: groupId, optionId, order, title
     MenuItem item1 = menu.add(groupId, 1, 1, "10 points");
     MenuItem item2 = menu.add(groupId, 2, 2, "20 points");
     MenuItem item3 = menu.add(groupId, 3, 3, "30 points");
     MenuItem item4 = menu.add(groupId, 4, 4, "40 points");
     //MenuItem item5 = menu.add(groupId, 5, 5, "50 points");

     MenuItem item6 = menu.add(groupId, 6, 8, "Red text");
     MenuItem item7 = menu.add(groupId, 7, 7, "Green Text");
     MenuItem item8 = menu.add(groupId, 8, 6, "Blue text");

     //set icons
     item1.setIcon(R.drawable.uno);
     item2.setIcon(R.drawable.dos);
     item3.setIcon(R.drawable.tres);
     item4.setIcon(R.drawable.cuatro);

     // shortcuts using device's keyboard-keypad
     // on a G1 open slide open the keyboard and
     // type letter u (same as pressing menu UNO)
     item1.setShortcut('1', '1');
     item2.setShortcut('2', '2');
     item3.setShortcut('3', '3');
     item4.setShortcut('4', '4');
```

Remove this line from previous version

Figures used in this example were taken from:
C:\android-sdk-windows\platforms\android-4\data\res\drawable

Primer za menije - proširenje

- Treba i posebno obraditi podmeni

```
private boolean applyMenuOption(MenuItem item){

    int menuItemId = item.getItemId(); //1, 2, 3, ...11

    String strMsg2 = etMessage2.getText().toString();

    if (menuItemId < 5) {
        // first four options are for setting text size
        int newPointSize = arrayPointSize[menuItemId - 1];
        etMessage1.setTextSize(newPointSize);
        etMessage2.setTextSize(newPointSize);
    }
    else if (menuItemId == 5) {
        // the sub-menu (attached to 5th item) is processed here
        etMessage1.setText (
                    "You have selected: \n" +item.getTitle()
                  + "\nId: " + menuItemId
                  + " order: " + item.getOrder() );
}
    // either change color on text1 or style on text2
    else if (menuItemId == 6)
        etMessage1.setTextColor(0xffff0000); // red
```

Same as before

Take care of sub-menu here

Primer za menije - proširenje

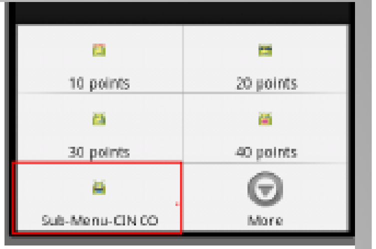
- Izmene u metodi *populateFirstMenu*

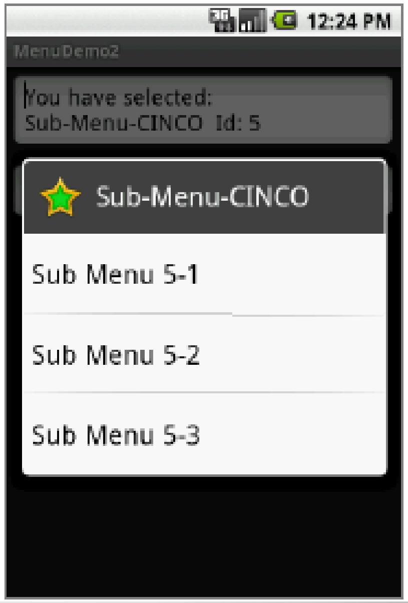
```
    // adding a sub-menu as fifth entry of this menu
    // .addSubMenu(int groupId, int itemId, int order, CharSequence title)
    int smGroupId = 0; // don't care, same as Menu.NONE
    int smItemId = 5;  // fifth element
    int smOrder = 5;   // don't care, same as Menu.NONE

    SubMenu mySubMenu5 =  menu.addSubMenu(smGroupId, smItemId, smOrder, "Sub-Menu-CINCO");
    mySubMenu5.setHeaderIcon(R.drawable.btn_rating_star_on_pressed);
    mySubMenu5.setIcon(R.drawable.cinco);
    // .add(int groupId, int itemId, int order, CharSequence title)

    MenuItem sub51 = mySubMenu5.add(smGroupId,5,1,"Sub Menu 5-1");
    MenuItem sub52 = mySubMenu5.add(smGroupId,5,2,"Sub Menu 5-2");
    MenuItem sub53 = mySubMenu5.add(smGroupId,5,3,"Sub Menu 5-3");


} //populateMyFirstMenu
```

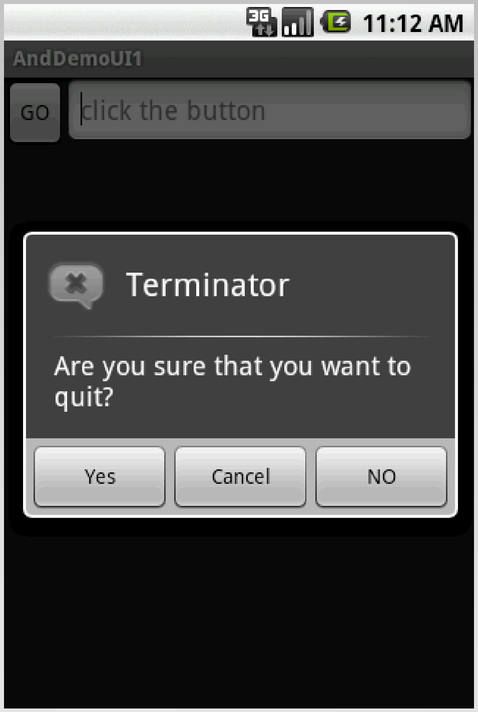




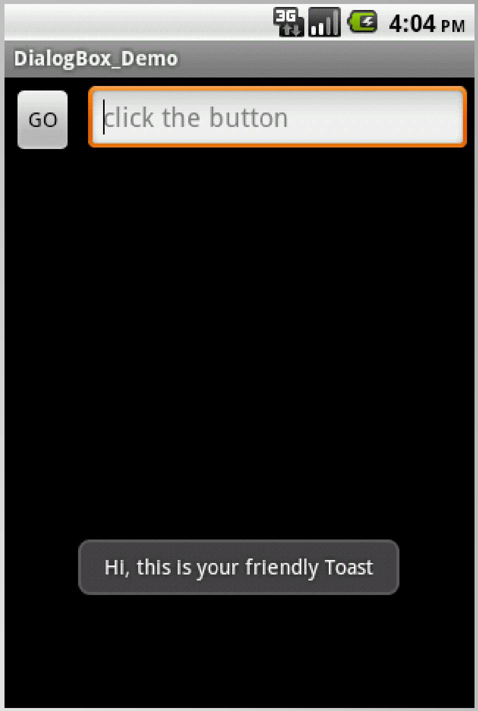
You have selected:
Sub-Menu-CINCO Id: 5

Dijalog box-ovi

- Android podržava dva tipa primitivnih dialog box-ova
  - AlertDialog
  - Toast
- AlertDialog
  - Skoro modalan
  - Prikazuje kratku poruku koja delimično zaklanja pozadinu
  - Prihvata kratak odgovor – kao klik na jedno od ponuđenih dugmića



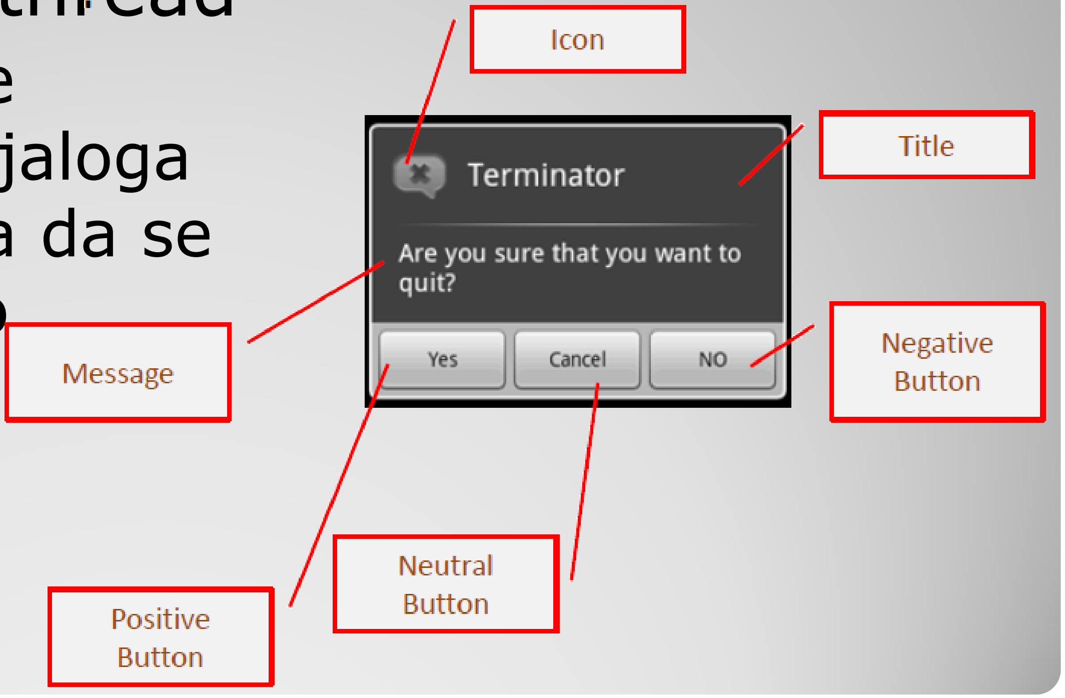
Are you sure that you want to quit?



Hi, this is your friendly Toast

Dijalog box-ovi

- AlertDialog je modalan zato što je neophodna intervencija korisnika da bi se zatvorio (ne zatvara se automatski na timeout)
- Nije tipičan modalan dijalog zato što ne zaustavlja glavni thread
    - Ostatak koda posle poziva za prikaz dijaloga normalno nastavlja da se izvršava i pre nego korisnik zatvori dijalog



Primer dijaloga

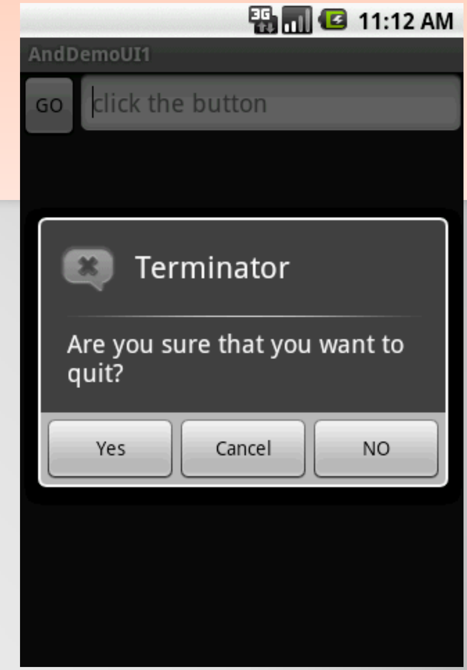
```
...
import android.content.DialogInterface;
import android.os.Bundle;
...

public class AndDemoUI1 extends Activity {
  Button btnGo;
  EditText txtMsg;
  String msg;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    txtMsg= (EditText)findViewById(R.id.txtMsg);
    btnGo= (Button) findViewById(R.id.btnGo);
    btnGo.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {
        AlertDialog dialBox= createDialogBox();
        dialBox.show();
        txtMsg.setText("I am here!");
      }
    });
  }//onCreate
```

Primer dijaloga

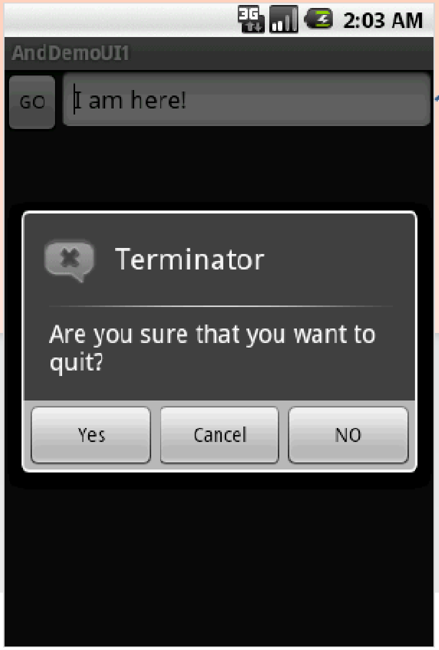
```
  private AlertDialog createDialogBox(){
    AlertDialog myQuittingDialogBox = new AlertDialog.Builder(this)
      //set message, title, and icon
    .setTitle("Terminator")
    .setMessage("Are you sure that you want to quit?")
    .setIcon(R.drawable.ic_menu_end_conversation)
    //set three option buttons
    .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
      public void onClick(DialogInterface dialog, int whichButton) {
        //whatever should be done when answering "YES" goes here
        msg= "YES "+ Integer.toString(whichButton);
        txtMsg.setText(msg);
      }
    })//setPositiveButton
```



Primer dijaloga

```
    .setNeutralButton("Cancel",newDialogInterface.OnClickListener() {
      public void onClick(DialogInterface dialog, int whichButton) {
        //whatever should be done when answering "CANCEL" goes here
        msg= "CANCEL "+ Integer.toString(whichButton);
        txtMsg.setText(msg);
      }//OnClick
    })//setNeutralButton

    .setNegativeButton("NO", new DialogInterface.OnClickListener() {
      public void onClick(DialogInterface dialog, int whichButton) {
        //whatever should be done when answering "NO" goes here
        msg= "NO "+ Integer.toString(whichButton);
        txtMsg.setText(msg);
      }
    })//setNegativeButton
    .create();
    .return myQuittingDialogBox;
  }// createDialogBox
}// class
```

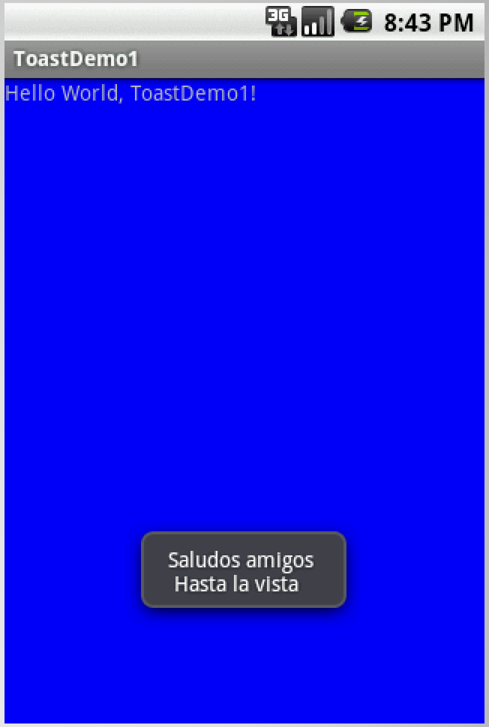


This text is set right after showing the dialog box

Toast

- Toast je privremen box sa porukom
- Automatski se gasi posle timeout-a bez neophodne korisnikove akcije
- Prikazuju kratak tekst obično
- Nikada ne dobijaju fokus
- Izgledaju kao da lebde iznad aplikacije
- Primer

  -
    ```
    Toast.makeText(
        context,
        message,
        duration ).show();
    ```



Hello World, ToastDemo1!

Saludos amigos
Hasta la vista

Toast

- Konkretan primer

```
    Toast.makeText(
        getApplicationContext(),
        "Saludos amigos \n Hasta la vista",
        Toast.LENGTH_LONG).show();
```

- Context – šta je to?
  - Uglavnom se koristi za učitavanje i pristup resursima
  - Svi widget-i imaju context kao parametar konstruktora
  - Generalno u Android aplikaciji postoje dva tipa konteksta
    - Activity context – ovaj tip se tipično prosleđuje konstruktorima
    - Application context

Toast

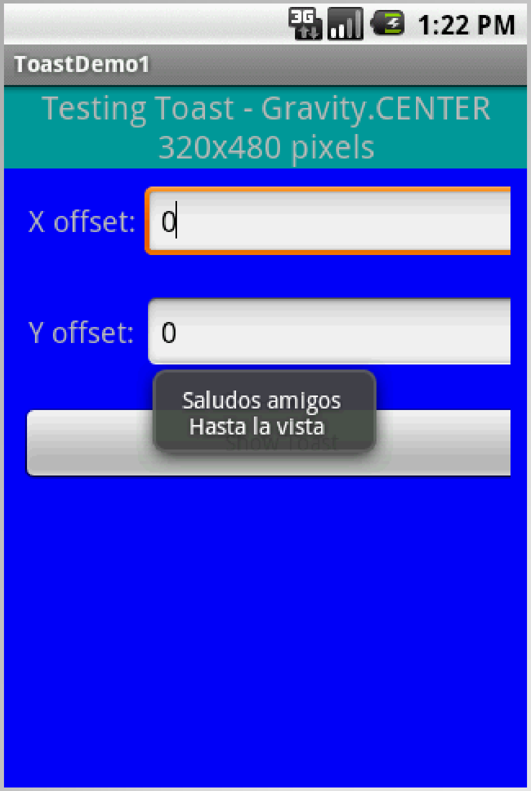
- Toast se podrazumevano prikazuje u dnu ekrana i po sredini
- Ovo može da se promeni
  - *void setGravity(int gravity, int xOffset, int yOffset)*

    Menja poziciju
  - *void setMargin(float horizontalMargin, float verticalMargin)*

    Menja margine
- Gravity parametar
  - *Gravity.CENTER, Gravity.TOP, Gravity.BOTTOM*
  - Offset parametri su relativni u pikselima
- Margine su podrazumevano postavljene na 50% sa obe strane

Toast - pozicioniranje

- Primer pozicioniranja toast-a



1:22 PM

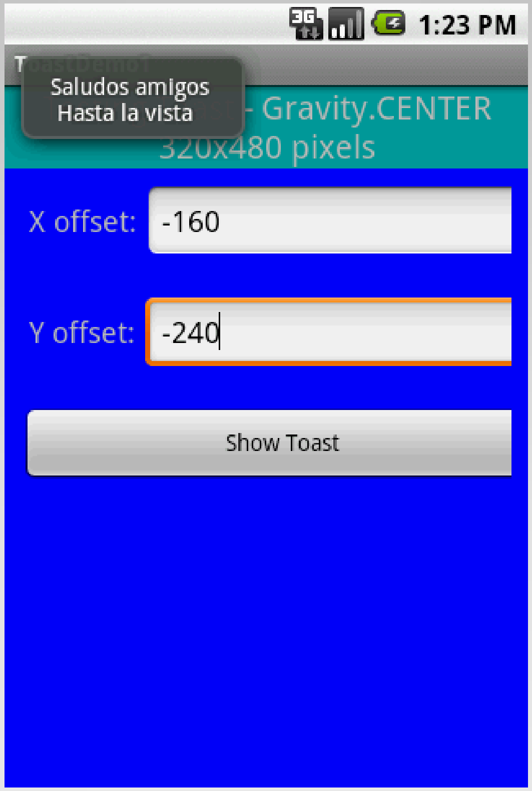
ToastDemo1

Testing Toast - Gravity.CENTER
320x480 pixels

X offset: 0

Y offset: 0

Saludos amigos
Hasta la vista



1:23 PM

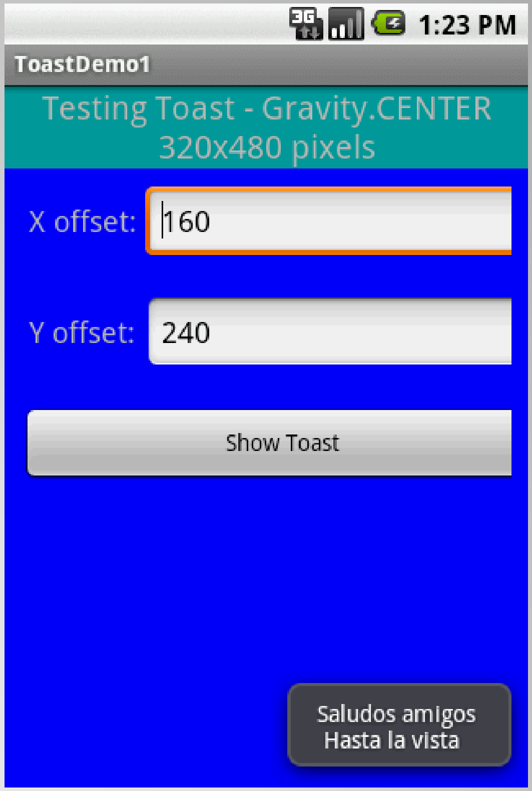
ToastDemo1

Testing Toast - Gravity.CENTER
320x480 pixels

Saludos amigos
Hasta la vista

X offset: -160

Y offset: -240



1:23 PM

ToastDemo1

Testing Toast - Gravity.CENTER
320x480 pixels

X offset: 160

Y offset: 240

Saludos amigos
Hasta la vista

Toast – modifikacija izgleda

- Toast se može modifikovati tako da mu se menja boja, oblik, tekst i pozadina
- Potrebni koraci za izmenu
  - Definisati XML layout novog custom view-a
  - Taj layout **MORA** da ima *TextView* sa imenom *text*
  - Opciono može da se doda *android:background* tom *TextView-u*
  - Pozadina može da bude slika (.png) ili XML definisan oblik

Toast – modifikacija izgleda

- Kreiramo novi custom my_toast_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/my_toast_layout_root"
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:padding="10dp”>
  <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="20dp"
        android:background="@drawable/my_border”>
  </TextView>
</LinearLayout>
```

Toast – modifikacija izgleda

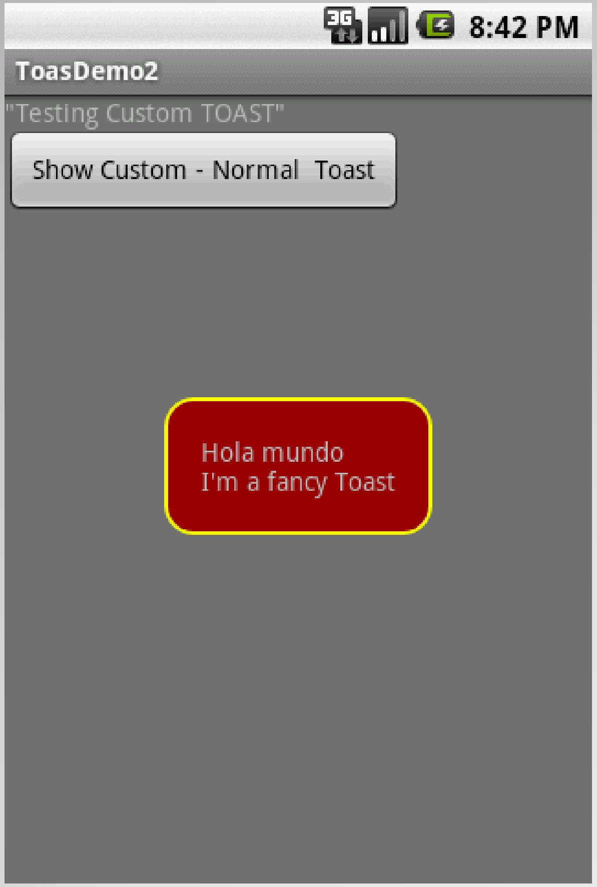
- Možemo da kreiramo i opcioni background element u resursima

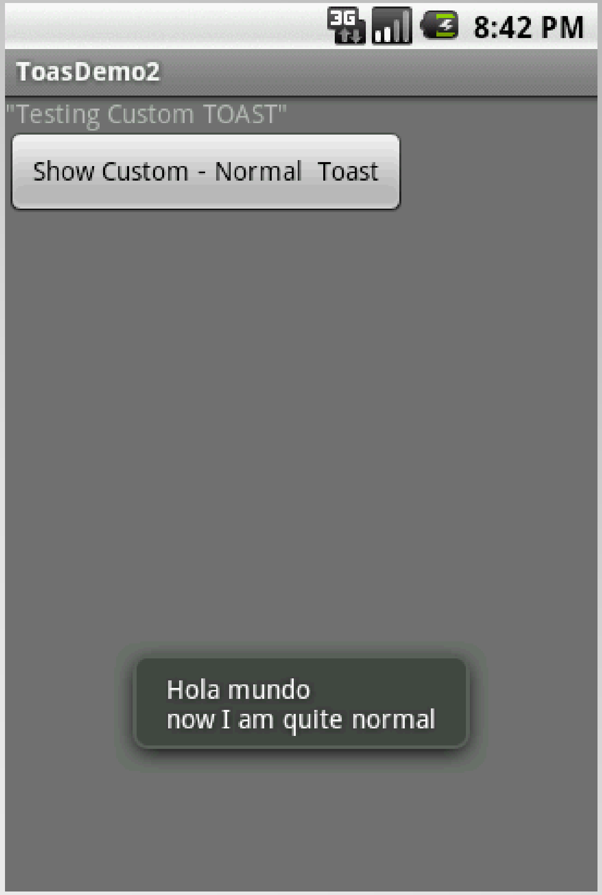
```
<?xml version="1.0" encoding="UTF-8" ?>
<shape
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:shape="rectangle">
  <stroke android:width="2dp" android:color="#ffffff00" />
  <solid android:color="#ff990000" />
  <padding android:left="10dp" android:top="4dp"
        android:right="10dp" android:bottom="4dp" />
  <corners android:radius="15dp" />
</shape>
```

- U ovom primeru koristimo SHAPE, ali može biti i PNG sličica

Toast – modifikacija izgleda

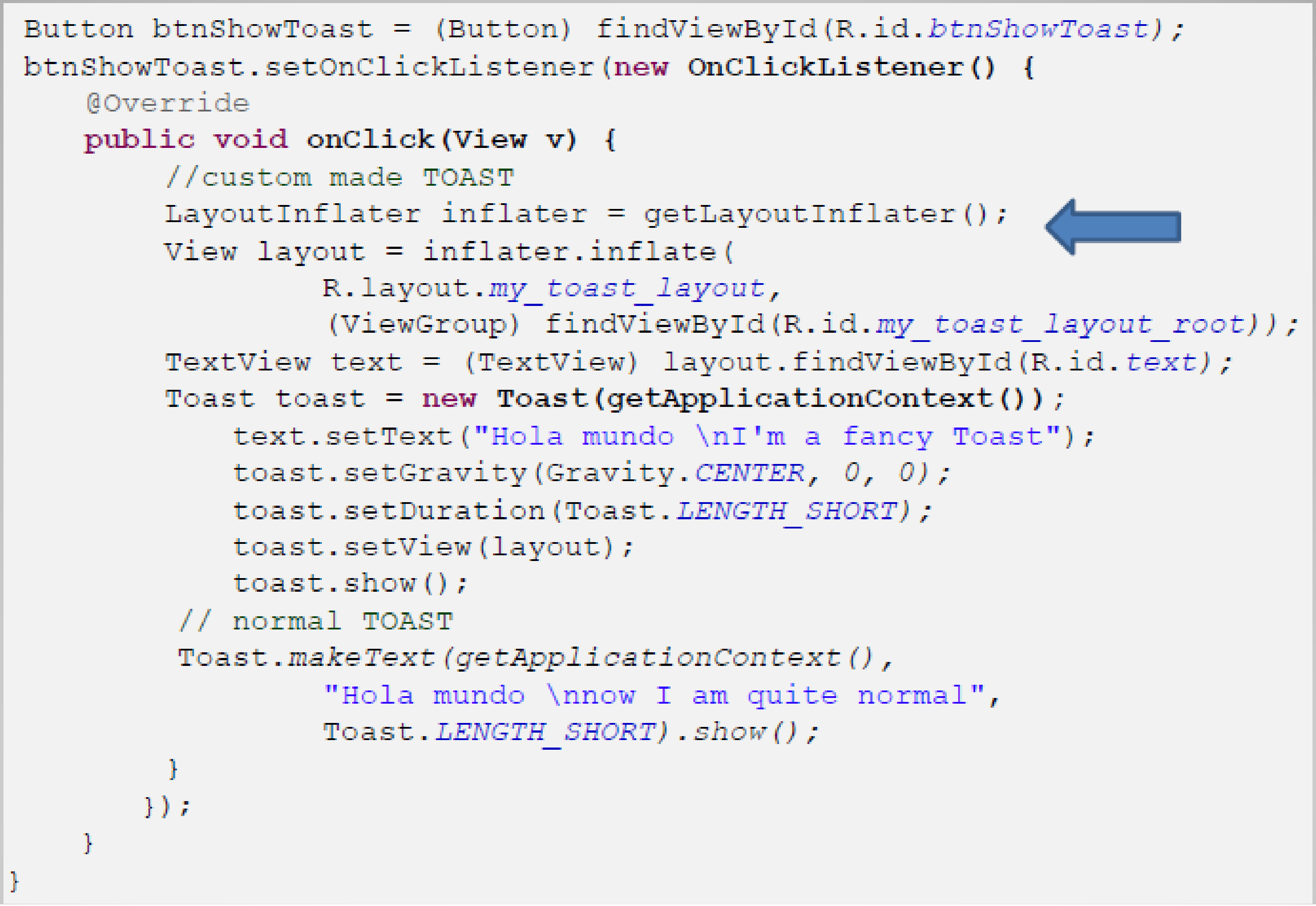
- Rezultat custom-izacije izgleda Toast-a





Toast – modifikacija izgleda

- Primena custom layout-a na Toast



```
Button btnShowToast = (Button) findViewById(R.id.btnShowToast);
btnShowToast.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
         //custom made TOAST
         LayoutInflater inflater = getLayoutInflater();
         View layout = inflater.inflate(
                  R.layout.my_toast_layout,
                  (ViewGroup) findViewById(R.id.my_toast_layout_root));
         TextView text = (TextView) layout.findViewById(R.id.text);
         Toast toast = new Toast(getApplicationContext());
             text.setText("Hola mundo \nI'm a fancy Toast");
             toast.setGravity(Gravity.CENTER, 0, 0);
             toast.setDuration(Toast.LENGTH_SHORT);
             toast.setView(layout);
             toast.show();
          // normal TOAST
          Toast.makeText(getApplicationContext(),
                  "Hola mundo \nnow I am quite normal",
                  Toast.LENGTH_SHORT).show();
         }
       });
    }
}
```

LayoutInflater

- Nekada je potrebno izmeniti kako Android prikazuje (render) konkretan View
- Kada je hijerarhija View-a učitana možemo da uzmemo bilo koji čvor i da ga "dopunimo" tako što u njega učitamo (inflate) neki layout
- *public View inflate (int resource, ViewGroup root)*
  - *resource* – ID resursa layout-a
  - *root* – čvor hijerarhije koji će biti parent inflate-ovanom layout-a

```
LayoutInflater inflater = getLayoutInflater();
View layout = inflater.inflate(
                R.layout.my_toast_layout,
                (ViewGroup) findViewById(R.id.my_toast_layout_root));
TextView text = (TextView) layout.findViewById(R.id.text);
```