# Application development for mobile devices

Dialogs, Menus, Preferences

# Themes

- Preferences
- Menus
- Dialogs

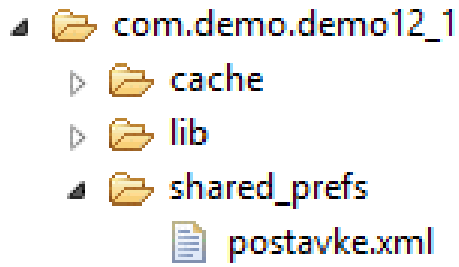
# Preferences

- Characteristics
  - Saving / loading data
  - Sharing data between components
- More complex data
  - File
  - Database
- Simple data
  - SharedPreferences

# SharedPreferences

- saved in the application directory (permanently)
- application can have multiple settings / files

1. val sharedPreferences = context.getSharedPreferences (String name, int mode)

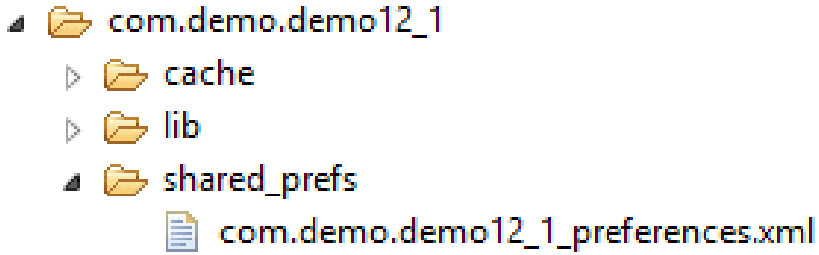



FileName

AccessMode

MODE_PRIVATE

2. SharedPreferences PreferenceManager.getDefaultSharedPreferences(Context context)



# SharedPreferences – Reading data

```
var sp = PreferenceManager.getDefaultSharedPreferences(this)

var b = sp.getBoolean (“myBoolean", true)
val f = sp.getFloat (“myFloat", 0)
val i = sp.getInt (“myInt", 0)
val l = sp.getLong (“myLong", 0)

val s = sp.getString ("myString", "")
val ss = sp.getStringSet (“myStringSet", emptySet())
```

# SharedPreferences – Saving data

```
val sp = PreferenceManager.getDefaultSharedPreferences(this)
Editor editor = sp.edit()
editor.putBoolean (“myBoolean", true)
editor.putFloat (" myFloat", 1)
editor.putInt (" myInt", 2)
editor.putLong (" myLong", 3)
editor.putString (" myString", "Hello World")
val ss = emptySet<String>()
ss.add("Hello")
editor. putStringSet (" myStringSet", ss)
editor.commit()
```

# Interface for changing settings

- PreferenceScreen + PreferenceFragment
  - Consistency with other applications
  - Automatic connection with SharedPreference
- PreferenceScreen
  - XML document (/res/xml direktorij)
  - Consists of
    - controls
    - categories with controls

# PreferenceScreen

- Controls

CheckBoxPreference
SwitchPreference
EditTextPreference

ListPreference
MultiSelectListPreference

| | |
|---|---|
| key | Identifier (key inside SharedPreference) |
| title | Description |
| summary | Long description |
| defaultValue | Default value |

# PreferenceFragment

- Create a class that inherits PreferenceFragmentCompat()
- Add settings from an XML document

```
class PreferenceFragment : PreferenceFragmentCompat() {
    override fun onCreatePreferences(
       savedInstanceState: Bundle?,
       rootKey: String?) {
         addPreferencesFromResource(R.xml.preferences)
    }
}
```

# Menus

# Menus

**>= API Level 11**

- Action Bar
- Overflow menu
- Floating menu

**< API Level 11**

- Icon menu
- Expanded menu

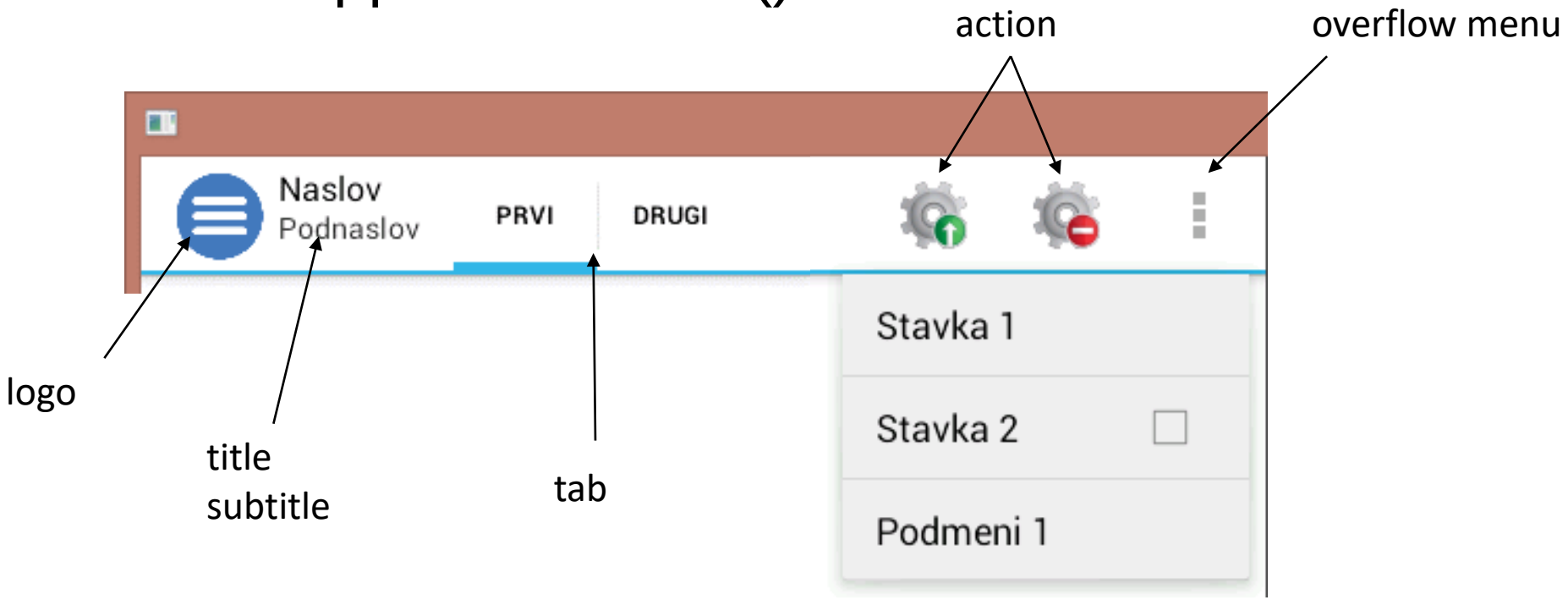
- Submenu
- Context menu
- Popup menu
- Icon
- Title
- Checkbox
- Radiobutton

# Action Bar

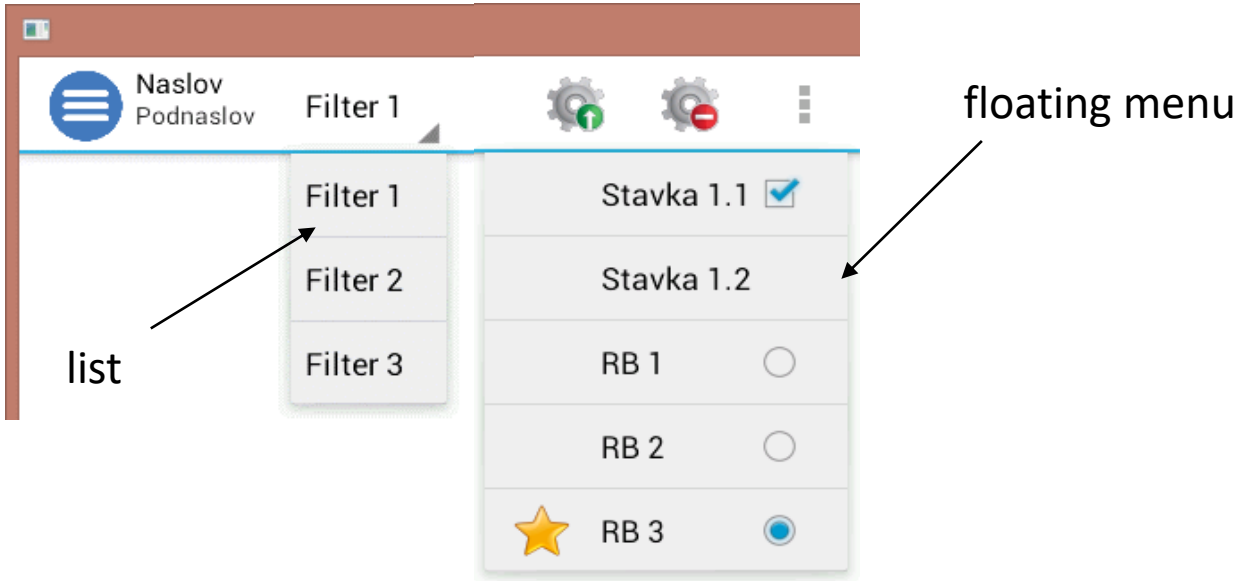
Prerequisites

- API Level >= 11

Context.supportActionBar()



# Action Bar

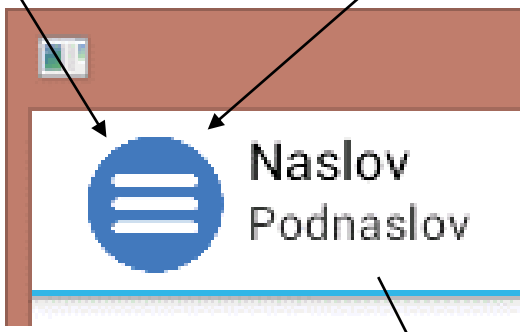



| | Action | Overflow | Floating |
|---|---|---|---|
| **Slika** | + | - | + |
| **Tekst** | + | + | + |
| **RadioButton** | - | + | + |
| **CheckBox** | - | + | + |

# Logo and text

actionBar.setDisplayUseLogoEnabled(true);
actionBar.setLogo (int resID)

actionBar.setDisplayShowHomeEnabled(true);




actionBar.setDisplayShowTitleEnabled(true);
actionBar.setTitle("Naslov");
actionBar.setSubtitle("Podnaslov");

# Menus - XML

- Items can be defined in an XML document
- res/menu directory

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/miPreferences"
        android:title="@string/preferences"
        android:icon="@drawable/preferences"
        app:showAsAction="always"/>

    <item
        android:id="@+id/miExit"
        android:title="@string/exit"
        android:icon="@drawable/exit"
        app:showAsAction="ifRoom"/>
</menu>
```

# Important methods - Activity

onCreate

Action Bar initialization

onCreateOptionsMenu

Menu initialization (called once)

onPrepareOptionsMenu

Menu customization
(called every time prior presenting)

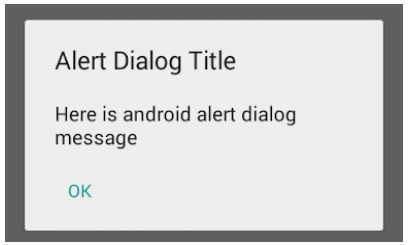
onOptionsItemSelected

Handling selection
(called after every selection)

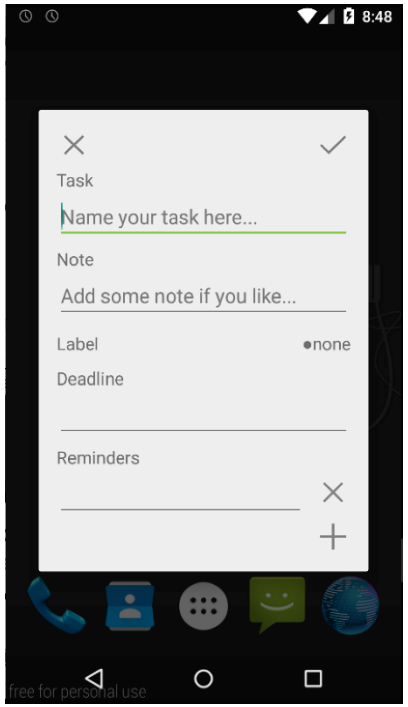
# Dialogs

# Dialog

- A window that requires the user
  - confirmation
  - additional information
- Smaller than the main window (mostly)
- Approaches
  - Class Dialog
  - Activities with a dialogue theme
  - Toast (short notices)

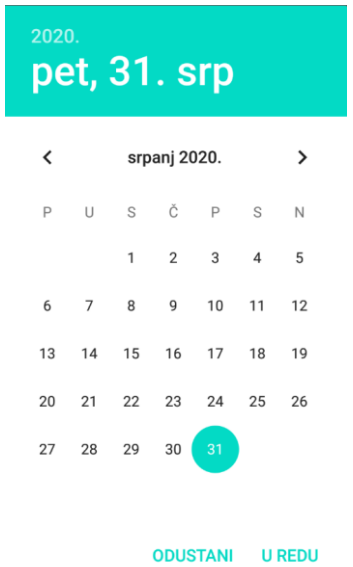


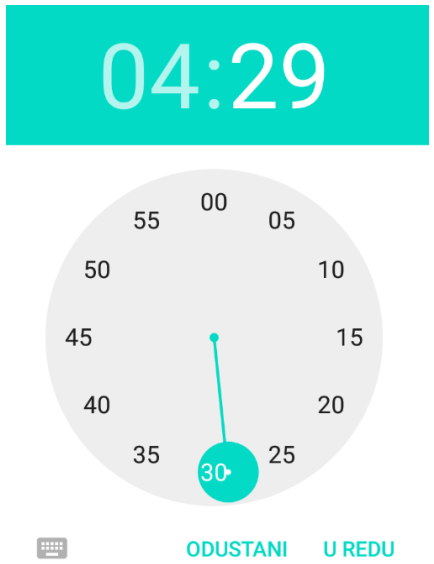


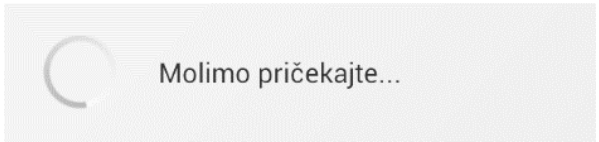


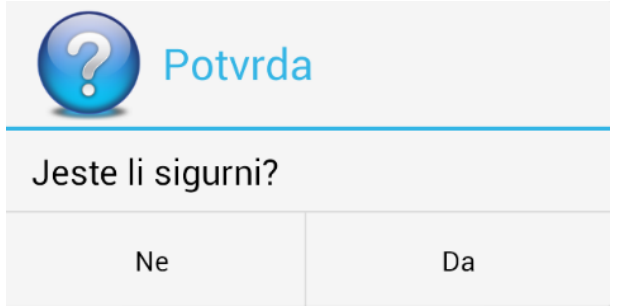
# Dialog class

- Embedded dialogs
- CharacterPickerDialog
  - DatePickerDialog
  - TimePickerDialog
  - ProgressDialog
  - AlertDialog

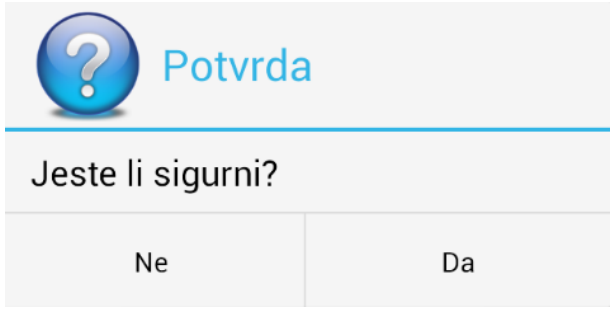



# AlertDialog

- *Builder pattern*




```
AlertDialog.Builder(this).apply {
    setTitle(R.string.potrvda)
    setMessage(getString(R.string.sigurni))
    setIcon(R.drawable.upitnik)
    setCancelable(true)
    setPositiveButton("Ok") { _, _ -> potvrdi() }
    setNegativeButton(getString(R.string.cancel), null)
    show()
}
```

# Demo

- DialogsMenuPreferences app


Source:http://www.jnhsolutions.com/contact-us/request-a-demo/

# Thank you for attention!