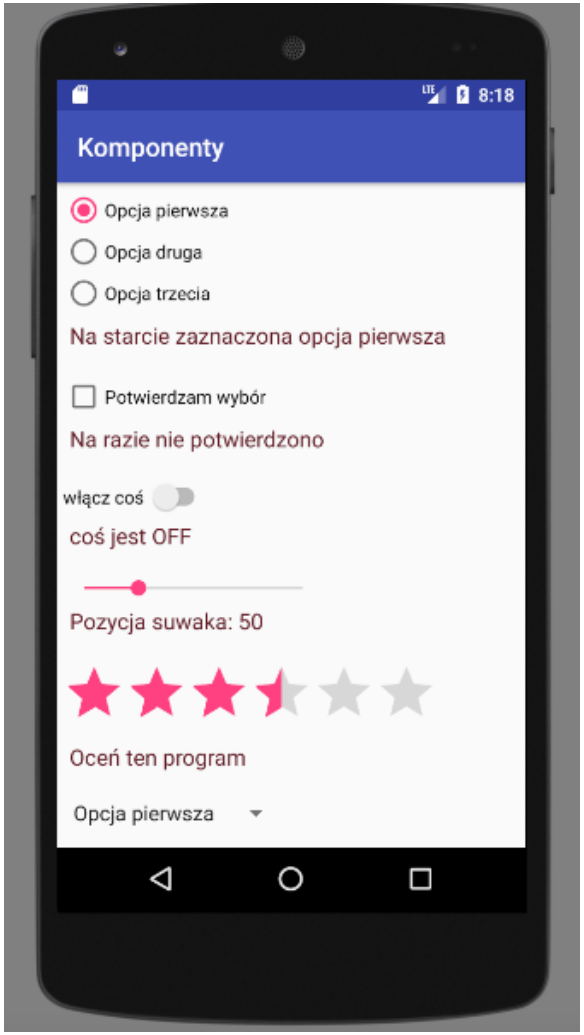
APLIKACJE MOBILNE

Wykład 05

dr Artur Bartoszewski

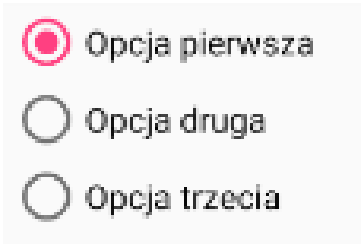
Kontrolki systemu

Android udostępnia kilka standardowych kontrolek



RadioGroup i RadioButton

```
23
24      <RadioGroup
25          android:id="@+id/radiGroup01"
26          android:layout_width="wrap_content"
27          android:layout_height="wrap_content"
28          android:orientation="vertical">
29
30          <RadioButton
31              android:id="@+id/rb1"
32              android:checked="true"
33              android:text="Opcja pierwsza"
34              android:layout_width="match_parent"
35              android:layout_height="wrap_content"
36              android:onClick="wyborOpcji"
37              />
38          <RadioButton...>
45          <RadioButton...>
52      </RadioGroup>
```



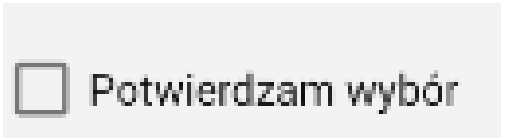
W pliku XML – tworzymy RadioGroup (dziedziczy po widoku LinearLayout) i wewnątrz umieszczamy widoki RadioButton. Przypisujemy im akcje onClick

RadioGroup i RadioButton

```
76      //------------RadioButton----------------------------
77      public void wyborOpcji(View view) {
78          TextView wynik = (TextView) findViewById(R.id.tekst);
79
80          switch (view.getId()) {
81              case R.id.rb1:
82                wynik.setText("Wybrano opicje pierwszą");
83                  break;
84              case R.id.rb2:
85                  wynik.setText("Wybrano opcję drugą");
86                  break;
87              case R.id.rb3:
88                  wynik.setText("Wybrano opcję trzecią");
89                  break;
90          }
91
92      }
```

W pliku .java tworzymy metodę obsługi akcji onClick.
To, który RadioButton wybrano rozpoznajemy za pomocą identyfikatora przekazanego przez parametr view.

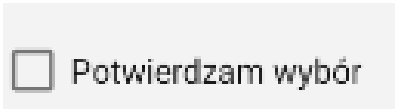
CheckBox




```
62      <CheckBox
63
64          android:id="@+id/potwierdzenie"
65          android:layout_width="wrap_content"
66          android:layout_height="wrap_content"
67          android:text="Potwierdzam wybór"
68          app:layout_constraintTop_toBottomOf="@id/tekst"
69          android:onClick="zatwierdzenieOpcji"
70          />
```

W pliku XML – tworzymy CheckBox. Przypisujemy mu akcje onClick

CheckBox




```
//-------------Switch-----------------------------------------
public void przelaczania(View view) {
    TextView wynik3 = (TextView) findViewById(R.id.tekst3);
    Switch przełacznik = (Switch) findViewById(R.id.switch01);
    if (przełacznik.isChecked()==Boolean.TRUE)  wynik3.setText("coś jest ON");
    else wynik3.setText("coś jest OFF");
}
```

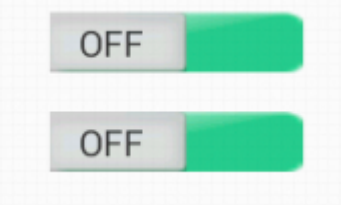
- W pliku .java – tworzymy metodę obsługi.
- Widok CheckBox ma właściwość „isChecked” (True/False) – dzięki temu rozpoznajemy, czy jest on zaznaczony.

Switch

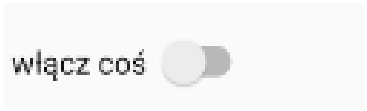

```
80      <Switch
81          android:id="@+id/switch01"
82          android:layout_width="wrap_content"
83          android:layout_height="wrap_content"
84          android:text="włącz coś "
85          android:textOn="ON"
86          android:textOff="OFF"
87          app:layout_constraintTop_toBottomOf="@id/tekst2"
88          android:onClick="przelaczania"
89          />
```

W pliku XML – tworzymy Switch. Przypisujemy mu akcje onClick

Parametry „textOn” i „textOff” są przeznaczone do wersji kontrolki w starszych Androidach



Switch




```
//-------------Switch-----------------------------------------------
public void przelaczania(View view) {
    TextView wynik3 = (TextView) findViewById(R.id.tekst3);
    Switch przełacznik = (Switch) findViewById(R.id.switch01);
    if (przełacznik.isChecked()==Boolean.TRUE)  wynik3.setText("coś jest ON");
    else wynik3.setText("coś jest OFF");
}
```

W pliku .java – tworzymy metodę obsługi.

Widok Switch ma właściwość „isChecked” (True/False) – dzięki temu rozpoznajemy, czy jest on włączony.

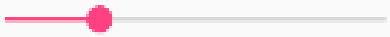
SeekBar

```
<SeekBar
    android:id="@+id/seekBar01"
    android:layout_width="0dp"
    app:layout_constraintTop_toTopOf="parent"
    android:layout_height="wrap_content"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    android:layout_marginStart="50dp"
    android:layout_marginEnd="50dp"
    android:layout_marginTop="20dp"
    android:max="100"
    android:min="-100"
    android:progress="0"
    />
```

Suwak posiada własność **progres** odpowiadającą za aktualną wartość, oraz własności **max i min** wyznaczające jej rozpiętość.

Słuchacz zdarzeń tworzymy w metodzie onCreate

SeekBar



```
final SeekBar suwak01 = findViewById(R.id.seekBar01);
suwak01.setOnSeekBarChangeListener(new );
```

SeekBar.OnSeekBarChangeListener{...} (android.widget.SeekBar)
boolean[]
byte[]
char[]
double[]
float[]
int[]
long[]
short[]
void[]
MainActivity (com.example.kontrolki)
Press Ctrl+Shift+Space to show only variants that are suitable by type

Dodając słuchacza zdarzeń korzystamy z kreatora

Kreator stworzy kod metod obsługo zdarzeń suwaka.

Aktualna wartość suwaka (podczas jego przesuwania) to parametr „progress”

SeekBar

```
final SeekBar suwak01 = findViewById(R.id.seekBar01);
suwak01.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
        //akcja podczas przesuwania suwaka
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
        //akcja na początku przesuwania suwaka
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        //akcja na koniec przesuwania suwaka
    }
});
```

Poza metodami słuchacza odczytać możemy wartość suwaka za pomocą metody **.getProgress()**

RatingBar

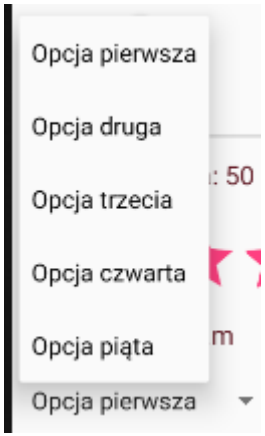
```
9   <RatingBar
10      android:id="@+id/ratingBar01"
11      android:layout_width="wrap_content"
12      app:layout_constraintTop_toTopOf="parent"
13      android:layout_height="wrap_content"
14      app:layout_constraintLeft_toLeftOf="parent"
15      app:layout_constraintRight_toRightOf="parent"
16      android:numStars="7"
17      android:rating="3.5"
18      />
```

Właściwości:
numStars – maksymalna liczba gwiazdek
rating – aktualna ocena (wyświetlana z dokładnością do pół gwiazdki

RatingBar

```
final RatingBar gwiazdki01 = findViewById(R.id.ratingBar01);
gwiazdki01.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float rating, boolean fromUser) {
        //Akcja po zmianie "ratingu"
    }
});
```

Obsługa podobnie jak SeekBar
Do odczytania wartości poza słuchaczem zdarzeń służy metoda **getRating()**



Spiner – menu rozwijane

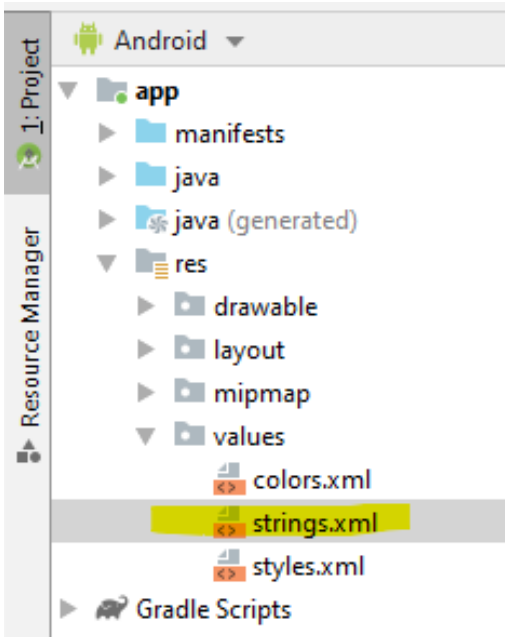
Spiner wymaga:

1. Przygotowania komponentu w pliku XML,
2. Przygotowania listy opcji w postaci tablicy stringów (w pliku strings.xml)
3. Połączenie tablicy z widokiem za pomocą ArrayAdaptera
4. Oprogramowania słuchacza zdarzeń

Aplikacje mobilne

Spiner – menu rozwijane

W pliku **strings.xml** dodajemy tablicę łańcuchów (string-array), której pozycjami (item) są wpisy w naszym menu typu „spinner”




```
<resources>
    <string name="app_name">kontrolki</string>
    <string-array name="zawartoscSpinnera">
        <item>Pozycja pierwsza</item>
        <item>Pozycja druga</item>
        <item>Pozycja trzecia</item>
        <item>Pozycja czwarta</item>
    </string-array>
</resources>
```

Spiner – menu rozwijane

Łączymy zmienną w programie z elementem layoutu

```
final Spinner menuSpinner = findViewById(R.id.spinner01);
```

Przygotowujemy ArrayAdapter – czyli obiekt definiujący w jaki sposób ma być wyświetlana tablica.

```
ArrayAdapter<String> adapter = new ArrayAdapter<String>( context: this,
        R.layout.support_simple_spinner_dropdown_item,
        getResources().getStringArray(R.array.zawartoscSpinnera));

adapter.setDropDownViewResource(R.layout.support_simple_spinner_dropdown_item);
```

Dodajemy adaprer do spinnera

```
menuSpinner.setAdapter(adapter);
```

Spiner – menu rozwijane

Aby program reagował na wybranie jednej z opcji dodajemy do spinnera słuchacza zdarzeń.

```
menuSpinner.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        //kcja po wybraniu pozycji
    }
});
```

O tym, którą pozycję wybraliśmy informuje parametr „position” zawierający jej indeks.

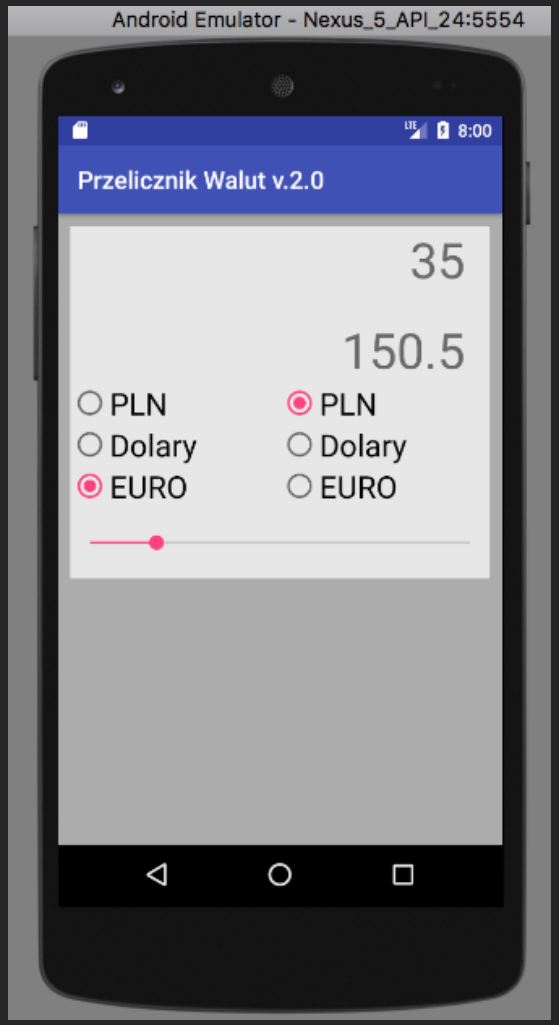
Aplikacje mobilne

Do nazwy pozycji w menu można dostać się za pomocą getResource

```
poleTekstowe.setText(
        getResources().getStringArray(R.array.zawartoscSpinnera)[0]
);
```

ZADANIE PRAKTYCZNE:

Przelicznik walut obsługiwany bez pomocy klawiatury ekranowej



```
13      <android.support.constraint.Guideline
14          android:layout_width="wrap_content"
15          android:layout_height="wrap_content"
16          android:id="@+id/guideline"
17          app:layout_constraintGuide_percent="0.5"
18          android:orientation="vertical" />
19
20      <TextView
21          android:id="@+id/textView01"
22          android:text="0"
23          android:layout_width="0dp"
24          android:layout_height="wrap_content"
25          app:layout_constraintLeft_toLeftOf="parent"
26          app:layout_constraintRight_toRightOf="parent"
27          android:textSize="40dp"
28          android:gravity="right"
29          android:paddingBottom="20dp"
30          android:paddingRight="20dp"
31          android:background="@color/widok"
32          />
33
34      <TextView
35          android:id="@+id/textView02"
36          android:text="0"
37          android:layout_width="0dp"
38          android:layout_height="wrap_content"
39          app:layout_constraintLeft_toLeftOf="parent"
40          app:layout_constraintRight_toRightOf="parent"
41          app:layout_constraintTop_toBottomOf="@id/textView01"
42          android:textSize="40dp"
43          android:gravity="right"
44          android:layout_marginBottom="20dp"
45          android:paddingRight="20dp"
46          android:background="@color/widok"/>
```

Wygląd aplikacji (w pliku XML)

Pola TextView wyświetlające wartości

Aplikacje mobilne

Wygląd aplikacji (w pliku XML)

Pola wyboru waluty – waluta wejściowa

```
50      <RadioGroup
51          android:id="@+id/radioGroup01"
52          android:layout_width="0dp"
53          android:layout_height="wrap_content"
54          app:layout_constraintTop_toBottomOf="@id/textView02"
55          app:layout_constraintLeft_toLeftOf="parent"
56          app:layout_constraintRight_toLeftOf="@id/guideline"
57          android:background="@color/widok">
58
59          <RadioButton
60              android:id="@+id/radioButton01"
61              android:layout_width="wrap_content"
62              android:layout_height="wrap_content"
63              android:text="PLN"
64              android:textSize="25sp"
65              android:checked="true"
66              android:onClick="zmianaWaluty"/>
67          <RadioButton...>
74          <RadioButton...>
81      </RadioGroup>
```

Aplikacje mobilne

Wygląd aplikacji (w pliku XML)

Pola wyboru waluty – waluta docelowa

```
<RadioGroup
    android:id="@+id/radioGroup02"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    app:layout_constraintTop_toBottomOf="@id/textView02"
    app:layout_constraintLeft_toLeftOf="@id/guideline"
    app:layout_constraintRight_toRightOf="parent"
    android:background="@color/widok">

    <RadioButton
        android:id="@+id/radioButton04"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="PLN"
        android:textSize="25sp"
        android:onClick="zmianaWaluty"/>
    <RadioButton...>
    <RadioButton...>
</RadioGroup>
```

Aplikacje mobilne

Wygląd aplikacji (w pliku XML)

Suwak - SeekBar

```
116     <SeekBar
117         android:id="@+id/seekBar01"
118         android:layout_width="0dp"
119         android:layout_height="wrap_content"
120         app:layout_constraintTop_toBottomOf="@id/radioGroup02"
121         app:layout_constraintLeft_toLeftOf="parent"
122         app:layout_constraintRight_toRightOf="parent"
123         android:padding="20dp"
124         android:background="@color/widok"
125         android:max="200"/>
126
```

```
12      TextView dane, wynik;
13      RadioButton danePLN, daneDolar, daneEuro, wynikPLN, wynikDolar, wynikEuro;
14      SeekBar suwak;
15      private double kursDolara=3.8, kursEuro=4.3;
16      @Override
17      protected void onCreate(Bundle savedInstanceState) {
18          super.onCreate(savedInstanceState);
19          setContentView(R.layout.activity_main);
20          dane = (TextView) findViewById(R.id.textView01);
21          wynik = (TextView) findViewById(R.id.textView02);
22          danePLN = (RadioButton) findViewById(R.id.radioButton01);
23          daneDolar = (RadioButton) findViewById(R.id.radioButton02);
24          daneEuro = (RadioButton) findViewById(R.id.radioButton03);
25          wynikPLN = (RadioButton) findViewById(R.id.radioButton04);
26          wynikDolar = (RadioButton) findViewById(R.id.radioButton05);
27          wynikEuro = (RadioButton) findViewById(R.id.radioButton06);
28          suwak = (SeekBar) findViewById(R.id.seekBar01);
29          SeekBar.OnSeekBarChangeListener l1 = new SeekBar.OnSeekBarChangeListener() {
30              @Override
31              public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
32                  dane.setText(String.valueOf(i));
33                  wynik.setText(
34                          (String.valueOf(przelicz((double) i))));
35              }
36              @Override
37              public void onStartTrackingTouch(SeekBar seekBar) {
38              }
39              @Override
40              public void onStopTrackingTouch(SeekBar seekBar) {
41              }
42          };
43          suwak.setOnSeekBarChangeListener(l1);
44      }
```

Kod aplikacji

Przypisanie kontrolek do zmiennych w programie.

Stworzenie suwaka i oprogramowanie metody wykonywanej w trakcie jego przesuwania

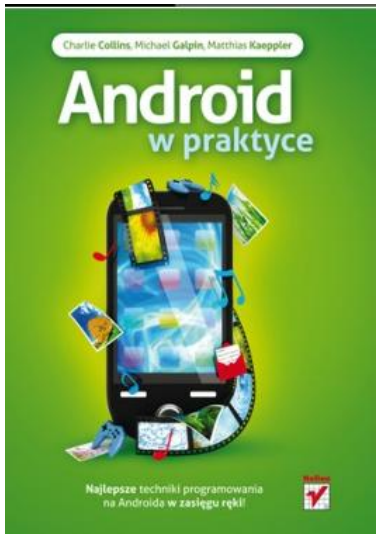
Aplikacje mobilne

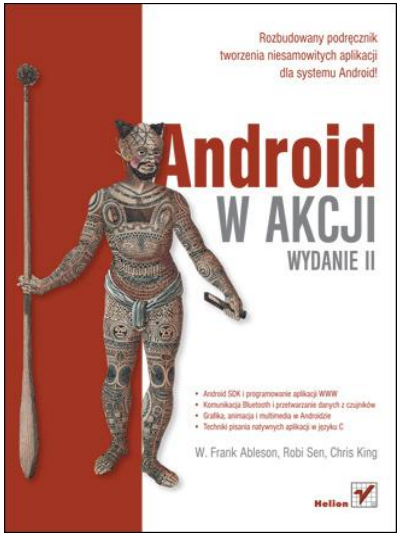
```
46      private double przelicz(double x) {
47          double kwotaPLN, wynik=0;
48          if (danePLN.isChecked()) kwotaPLN = x;
49          else if (daneDolar.isChecked()) kwotaPLN = x * kursDolara;
50          else kwotaPLN = x * kursEuro;
51
52          if (wynikPLN.isChecked()) wynik = kwotaPLN;
53          else if (wynikDolar.isChecked()) wynik= kwotaPLN / kursDolara;
54          else wynik = kwotaPLN / kursEuro;
55          return Math.round(wynik*100.0)/100.0;
56      }
57
58      public  void zmianaWaluty (View view) {
59          wynik.setText(
60                  (String.valueOf(
61                          przelicz(Double.parseDouble(
62                                  dane.getText().toString()))));
63      }
64
65  }
```

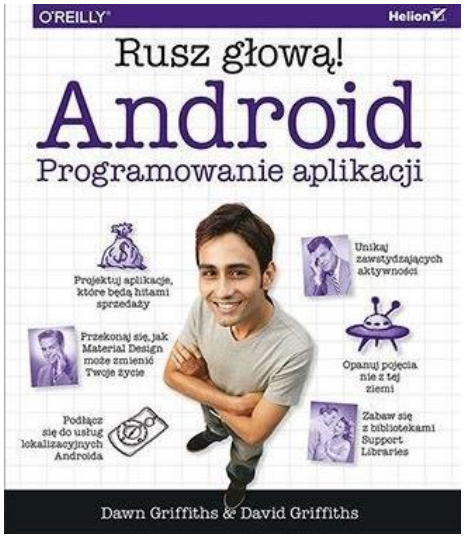
Metoda przeliczająca waluty

Metoda wypisująca wynik - korzysta z wyżej zdefiniowanej metody przelicz()

Literatura


Charlie Collins, Michael Galpin, Matthias Kaeppler
Android
w praktyce
Najlepsze techniki programowania na Androida w zasięgu ręki!


Rozbudowany podręcznik tworzenia niesamowitych aplikacji dla systemu Android!
Android
W AKCJI
WYDANIE II
W. Frank Ableson, Robi Sen, Chris King
Helion


O'REILLY
Helion
Rusz głową!
Android
Programowanie aplikacji
Projektuj aplikacje, które będą hitami sprzedaży
Przekonaj się, jak Material Design może zmienić Twoje życie
Podłącz się do usług lokalizacyjnych Androida
Unikaj zawstydzających aktywności
Opanuj pojęcia nie z tej ziemi
Zabaw się z bibliotekami Support Libraries
Dawn Griffiths & David Griffiths