Пора опять написать какое-нибудь полезное приложение (напоминаю, что до этого мы написали хорошую программу Счётчик ворон). На этот раз напишем конвертер, который позволит нам узнать длину длиннокота не только в метрах, но и в попугаях. Если вы не знаете, кто такой длиннокот, то вам прямая дорога на мой блог Субкота-5. Надеюсь, что такое субкота, вы знаете?

Изучив данную технику, вы сможете написать собственные конвертеры. Например, вы сможете конвертировать доллары в тугрики, футы в метры, градусы Цельсия в градусы по Фаренгейту, свинец в золото, ну и так далее...

Подготовка

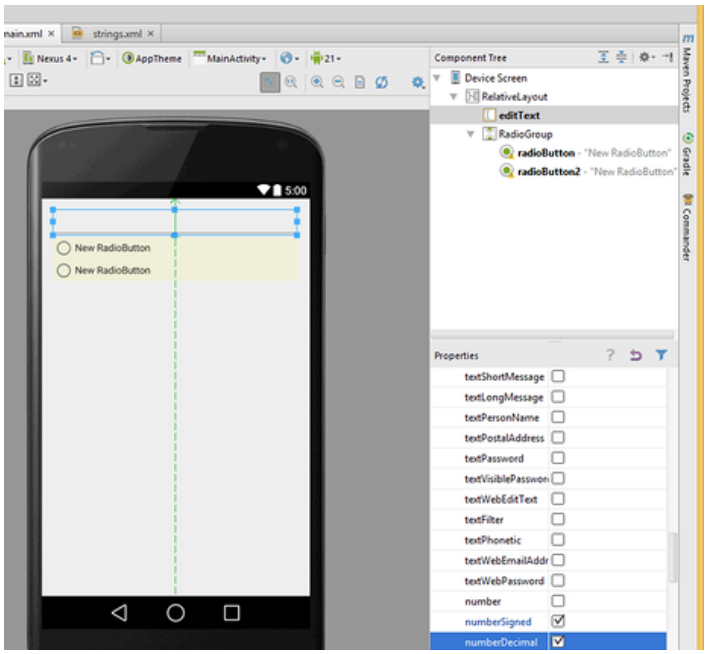
Создаём новый проект **Converter** и добавляем строковые ресурсы в файл **res/values/strings.xml**.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="action_settings">Settings</string>
    <string name="app_name">38 попугаев</string>
    <string name="radiobutton_meters">В метрах</string>
    <string name="radiobutton_parrots">В попугаях</string>
    <string name="buttonConvert_text">Конвертировать</string>
    <color name="activity_color">#3399cc</color>

</resources>
```

Открываем файл **activity_main.xml** в папке **res/layout/** и настраиваем внешний вид экрана приложения. Удаляем глупую надпись **Hello World!**. Далее с панели инструментов из раздела **Text Fields** перетаскиваем элемент **Plain Text**. Затем в разделе **Containers** находим элемент **RadioGroup** и также перетаскиваем на экран. Теперь из раздела **Widgets** дважды переносим элемент **RadioButton** и размещаем их внутри **RadioGroup**. Завершающий штрих - добавляем кнопку.

Мы подготовили каркас приложения. Теперь сделаем небольшой тюнинг. Назначим необходимые свойства добавленным элементам. Свойства можно менять через отдельную панель **Properties**.

Давайте сделаем так, чтобы при вводе текста в текстовом поле по умолчанию появлялась цифровая клавиатура. Так удобнее будет пользователю вводить длину кота. Выделяем компонент **EditText**, находим у него свойство **InputType**, раскрываем его и ставим флажки у свойств **numberSigned** и **numberDecimal**.



Далее присвоим текст переключателям **RadioButton**. Мы заранее уже заготовили строковые ресурсы для переключателей. Поэтому нам нужно просто назначить нужные ресурсы в свойствах **Text**: **radiobutton_meters** и **radiobutton_parrots**. Убедитесь, что у первого переключателя у свойства **Checked** установлено значение **true**.

У кнопки для свойства **Text** мы используем строковый ресурс **buttonConvert_text**, а для свойства **onClick** - **onClick**.

Чтобы приложение не выглядело стандартным, присвоим свойству **background** у корневого элемента значение **activity_color** (цвет морской волны).

Чтобы приложение выглядело профессиональнее, изменим также идентификаторы по умолчанию на более понятные: **editText**, **radioButtonMeter**, **radioButtonParrot**, **buttonConverter**.

Побудьте дизайнером, придумайте свои варианты. У меня получилось следующее.

Код разметки выглядит следующим образом:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:tools="http://schemas.android.com/tools"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:background="@color/activity_color"
                android:paddingBottom="@dimen/activity_vertical_margin"
                android:paddingLeft="@dimen/activity_horizontal_margin"
                android:paddingRight="@dimen/activity_horizontal_margin"
                android:paddingTop="@dimen/activity_vertical_margin"
                tools:context=".MainActivity">

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:inputType="numberDecimal|numberSigned"/>
```

```
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/editText">

        <RadioButton
            android:id="@+id/radioButtonMeter"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="@string/radiobutton_meters"/>

        <RadioButton
            android:id="@+id/radioButtonParrot"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/radiobutton_parrots"/>
    </RadioGroup>

    <Button
        android:id="@+id/buttonConverter"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/radioGroup"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="28dp"
        android:onClick="onClick"
        android:text="@string/buttonConvert_text"/>
</RelativeLayout>
```

Шкодим

Ну, давайте писать код, что-ли. Открываем файл активности **MainActivity.java**. Нам понадобятся два метода для конвертации величин и обработчик щелчка кнопки. При пустом значении текстового поля будем выводить Toast-сообщение. Начнём с двух методов.

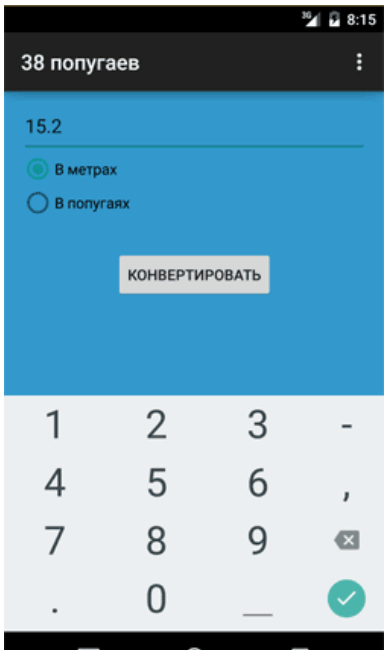
Методы просты. Указываем количество попугаев и делим на магическое число 7.6 - получаем метры. И наоборот, указываем число метров и получим количество попугаев. Откуда взялось число 7.6 вы узнаете позже.

При щелчке кнопки получаем текст из текстового поля и в зависимости от того, какой переключатель активен, применяем нужный метод конвертации. Полный код.

```
package ru.alexanderklimov.converter;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    // Конвертируем в метры
    private float convertParrotToMeter(float parrot) {
```

```
    // Конвертируем в попугаи
    private float convertMeterToParrot(float meter) {
        return (float) (meter * 7.6);
    }

    public void onClick(View view) {

        RadioButton meterRadioButton = (RadioButton) findViewById(R.id.radioButtonMeter);
        EditText inputEditText = (EditText) findViewById(R.id.editText);

        if (inputEditText.getText().length() == 0) {
            Toast.makeText(getApplicationContext(), "Введите длину кота",
                    Toast.LENGTH_LONG).show();
            return;
        }

        float inputValue = Float.parseFloat(inputEditText.getText().toString());
        if (meterRadioButton.isChecked()) {
            inputEditText.setText(String
                    .valueOf(convertParrotToMeter(inputValue)));
        } else {
            inputEditText.setText(String
                    .valueOf(convertMeterToParrot(inputValue)));
        }
    }
}
```

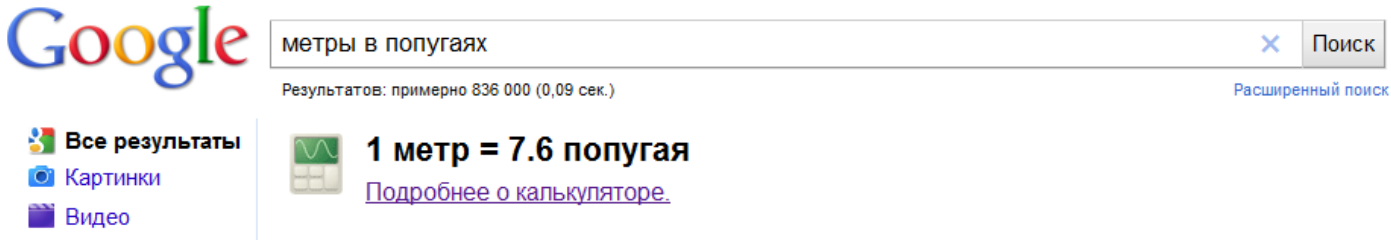
Запускаем проект и любуемся новым приложением.



попугаях" и нажимаем на кнопку. Получаем результат. В программе нет защиты от дурака и вам следует самому позаботиться, чтобы при вводе недопустимых символов программа не завершалась с ошибкой, а предупреждала пользователя.

Заключение

Чтобы быть уверенным, что мы применяли научные методы при написании программы, обратимся к Google с поисковым запросом метры в попугаях



Таким образом, вам осталось поймать длиннокота и измерить его. Результаты необходимо показать коту и, тогда вас ждет сюрприз! Кот от удивления произнесет знаменитую фразу: А в попугаях я гораздо длиннее!

Дополнительное чтение

Обсуждение статьи на форуме.