**Основы разработки интерфейсов мобильных приложений**

**Цель лабораторной работы:**

Изучение основ разработки интерфейсов мобильных приложений

**Задачи лабораторной работы:**

* Изучить элементы интерфейса
* Практическим путём научиться размещать элементы и менять их свойства
* Разработать прототип интерфейса собственного приложения

**6.1 Введение**

Лабораторная работа посвящена разработке интерфейсов мобильных приложений. Работа содержит подробное описание построения гармоничного понятного пользовательского интерфейса для главной активности приложения и описание основных элементов интерфейса. Лабораторная работа поможет выбрать концепцию своего приложения и начать разработку его интерфейса.

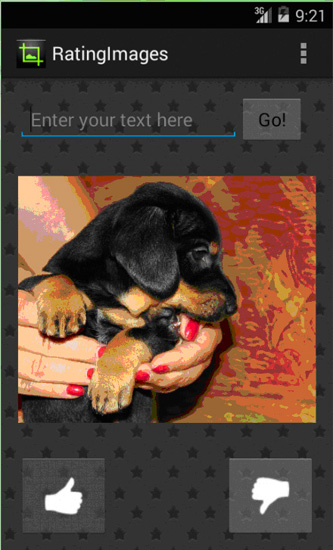
Более подробную информацию о разработке интерфейсов мобильных приложений под *Android* можно узнать на сайте [Design [Android Developers]](http://developer.android.com/design/index.html" \t "_blank).

**6.2 Создание прототипа интерфейса**

Рассмотрим пример разработки интерфейса приложения, которое ищет в сети *Интернет* изображения по запросу пользователя, позволяет оценивать их, скачивать, и посещать *интернет*-страницы сайтов, на которых было найдено изображение.

**6.2.1 Создание заготовки для приложения**

Выглядеть главное окно будет примерно так:



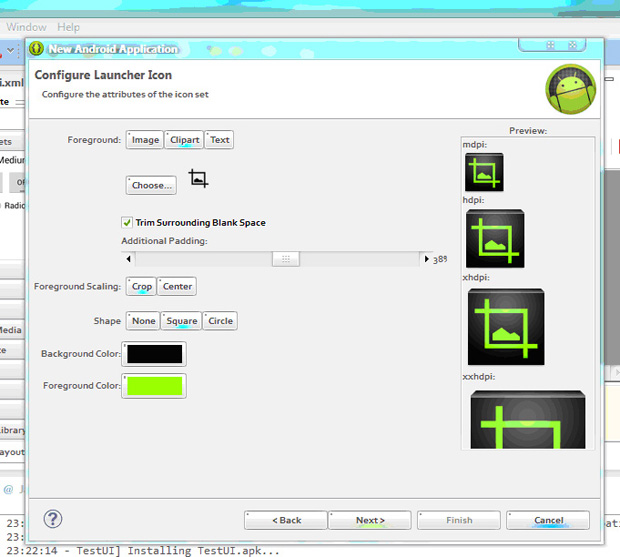
**Рис. 6.1.**Интерфейс главной активности

На нём присутствуют поле ввода текста для запроса пользователя и кнопка, начинающая поиск изображений. Внизу экрана две кнопки: "**like**" и "**dislike**", с их помощью пользователь сможет оценить изображение. После того, как пользователь сделает оценку изображения, текущее изображение закрывается и загружается следующее.

Итак, начнём с создания нового проекта. Назовём его "**RatingImages**" ("Рейтинг изображений").

На данном этапе изучения программирования под Android не обязательно менять иконку запуска, но эта лабораторная работа направлена на разработку дизайна интерфейса, и потому, приступим.

Создайте иконку на свой вкус.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_02.jpg)

**Рис. 6.2.**Создание иконки

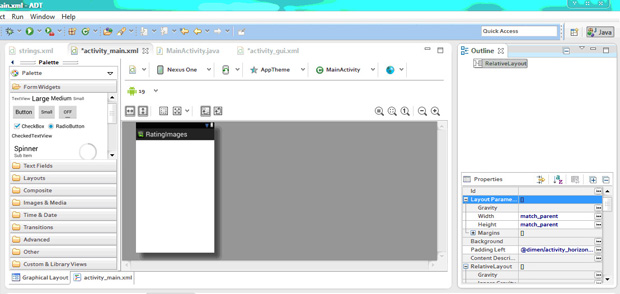
После создания проекта откройте **activity\_main.xml** из каталога **res/layout/**.

Когда вы откроете файл **activity\_main.xml**, вы увидите графический редактор макета. Благодаря этому редактору создание интерфейсов стало ещё интереснее, поскольку добавить элемент на форму можно при помощи перетаскивания мышью, к тому же, благодаря графическому редактору, не обязательно запускать эмулятор, чтобы увидеть результат своих трудов.

Теперь щелкните по вкладке **activity\_main.xml** в нижней части экрана. Открылся XML-редактор кода. Этот способ редактирования стандартный, но все изменения, вносимые в этот документ, можно так же ощутить визуально, перейдя на графический редактор.

Вернёмся на вкладку с графическим редактором. Во-первых, подготовим документ к началу работы, для этого удалите <TextView>.

Результат выглядит так:

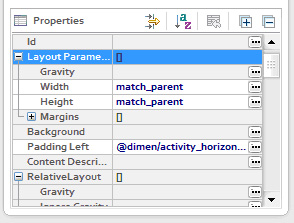
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_03.jpg)

**Рис. 6.3.**Проект, готовый к началу разработки

На рабочей области экрана остался один элемент. Это макет <RelativeLayout>. В нём позиция дочерних элементов может быть описана по отношению друг к другу или к родителю. Подробнее о макетах можно узнать [здесь](http://developer.android.com/guide/topics/ui/declaring-layout.html).

Два атрибута, ширина и высота (android:layout\_width и android:layout\_height), требуются для всех элементов для того, чтобы указать их размер.

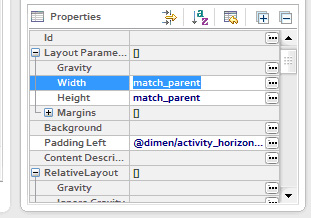
Так как <RelativeLayout> - это корень в макете, то нужно, чтобы он заполнял всю область экрана. Это достигается при помощи установки параметра "match\_parent" для ширины и высоты. Это значение указывает, что ширина и высота элемента будет равна ширине и высоте родителя.



**Рис. 6.4.**Свойства <RelativeLayout> элемента

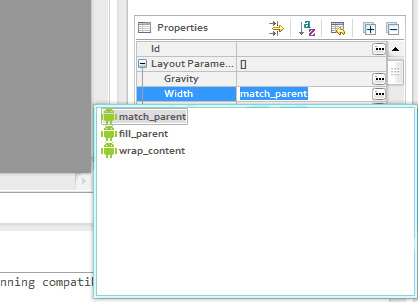
Как установить это значение? Будем разбираться.

Однократным щелчком левой кнопкой мыши по надписи "**Width**" активируйте строку с параметрами ширины:



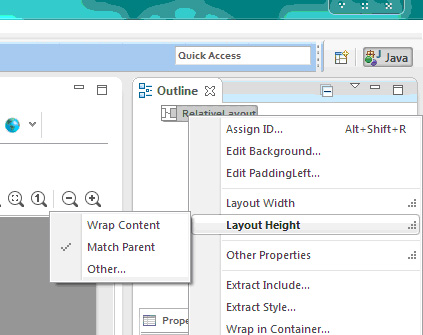
**Рис. 6.5.**Свойства <RelativeLayout> элемента, выбран атрибут android:layout\_width

Щелчком левой кнопки мыши по области ввода вызовите диалоговое окно, и двойным щелчком сделайте выбор параметра:



**Рис. 6.6.**Свойства <RelativeLayout> элемента, выбран параметр "match\_parent"

Или щелчком правой кнопки мыши по <RelativeLayout> в Outline:



**Рис. 6.7.**Контекстное меню <RelativeLayout> элемента

При выполнении первого способа вы увидели еще два возможных параметра: "fill\_parent" и "wrap\_content".

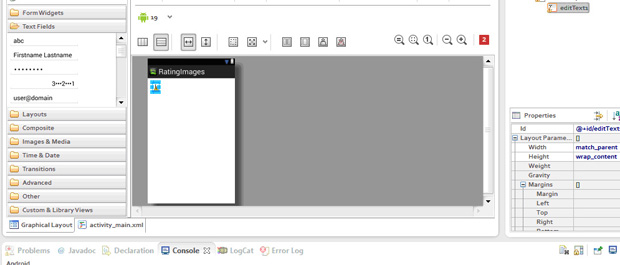
На самом деле, match\_parent = fill\_parent, но "fill\_parent" считается устаревшим, и к использованию в новых проектах предлагается"match\_parent".

Параметр "wrap\_content" указывает, что представление будет увеличиваться при необходимости, чтобы поддерживать соответствие содержанию экрана.

#### 6.2.2 Добавление текстового поля

Для начала добавьте элемент <LinearLayout> с горизонтальной ориентацией в <RelativeLayout>, и укажите для ширины и высоты параметр "wrap\_content". Теперь, для создания пользовательского редактируемого текстового поля, добавьте элемент <EditText> с параметром "wrap\_content" для ширины и высоты в <LinearLayout>.

Сейчас должно получиться примерно следующее:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_08.jpg)

**Рис. 6.8.**Добавление текстового поля

Возможно, появился желтый предупреждающий знак, но сейчас это не важно, со временем он исчезнет. Наличие таких предупреждений никак не влияет на компилируемость проекта.

Теперь переходим к настройке добавленных нами элементов.

Для многих элементов нужно назначать id, он обеспечивает уникальный идентификатор, который можно использовать как ссылку на объект из кода вашего приложения для управления им.

Откроем редактор кода XML-файла и обратим внимание на элемент

<EditText>:

<EditText

android:id="@+id/editText1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content">

<requestFocus/>

</EditText>

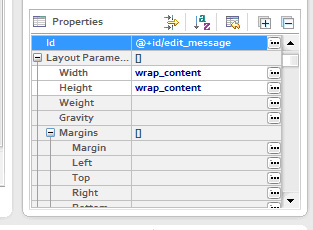
Строка <requestFocus/> появляется добавлением элемента requestFocus (папка Advanced), и позволяет установить фокус на нужном компоненте. Важно использовать этот элемент, когда у вас имеется, к примеру, три текстовых поля, и нужно, чтобы фокус был на втором из них.

При указании id, знак (@) требуется в том случае, если вы имеете в виду любой ресурс объекта из XML-файла. За ним следуют тип ресурса (в данном случае ID), косая черта (слеш) и имя ресурса (editText1).

Знак плюс (+) перед типом ресурсов необходим только тогда, когда вы впервые определяете идентификатор ресурса.

По сути, id, который создается автоматически, уже уникален, но грамотнее переименовывать id в соответствии со назначением элемента.

Зададим id для текстового поля. Для этого прямо в коде строку android:id="@+id/editText1" заменяем наandroid:id="@+id/edit\_message", жмём **CTRL+S** и открываем графический редактор. Если всё хорошо, то в свойствах текстового поля в графе **id** будет следующее:



**Рис. 6.9.**Идентификатор текстового поля

Добавим в код ещё две строки:

android:ems="10" - задает соответствия для симметричного отображения шрифтов,

android:hint="@string/edit\_message" - содержание тестового поля "по умолчанию", т.е. пока пользователь не начал вводить в поле текст. Вместо того, чтобы использовать просто слово (например android:hint="message"), что крайне не удобно при изменении основного языка приложения, используется ссылка на значение, хранящееся в файле **strings.xml**. Поскольку это относится к конкретному ресурсу (а не только к **id**), знак плюс не нужен.

Однако, мы ещё не определили строку ресурсов файле **strings.xml**, и потому вы получите следующее:

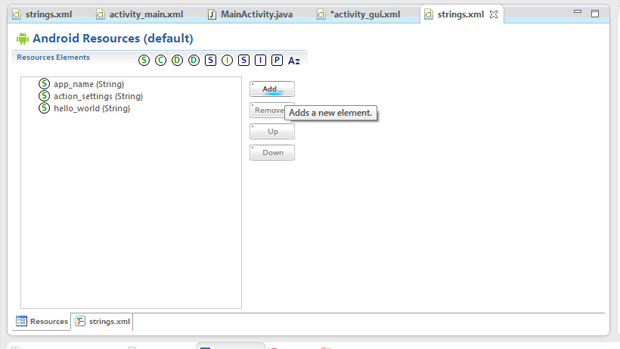


**Рис. 6.10.**Предупреждение: не найден ресурс, на который прописана ссылка

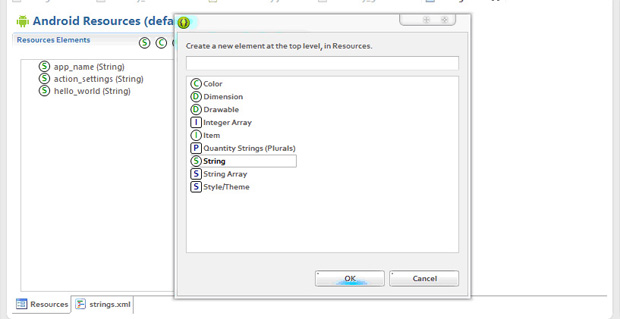
Для того, чтобы ссылка на ресурс начала работать, нужно этот ресурс создать.

Откройте файл **res/values/strings.xml**. Очевидно, что его тоже можно редактировать двумя способами: графически и вручную.

Выбирайте тот способ, который больше по душе.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_11.jpg)

**Рис. 6.11.**Редактирование файла ресурсов графическим способом; добавление нового ресурса

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_12.jpg)

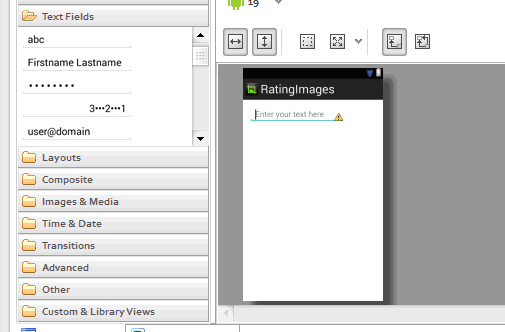
**Рис. 6.12.**Редактирование файла ресурсов графическим способом; выбор типа ресурса, выбран тип String

Заполняем поля "**Имя**" и "**Значение**":

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_13.jpg)

**Рис. 6.13.**Редактирование файла ресурсов графическим способом

Сохраняем и любуемся результатом:



**Рис. 6.14.**Отображение текста "По умолчанию" в поле ввода

Подробнее о ресурсах - [здесь](http://developer.android.com/guide/topics/resources/providing-resources.html).

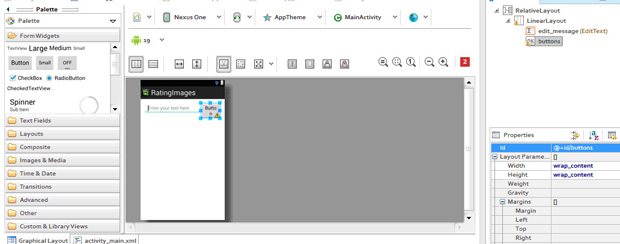
Аналогично создадим id для <LinearLayout>

android:id="@+id/linear1"

Сохраните изменения.

#### 6.2.3 Добавление кнопки

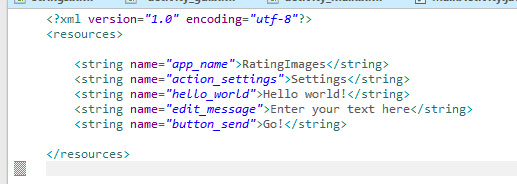
Теперь добавьте <Button> в макет после элемента <EditText>:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_15.jpg)

**Рис. 6.15.**Новая кнопка

Чтобы кнопка трансформировалась в соответствии с текстом кнопки, ширина и высота должны быть установлены во "wrap\_content".

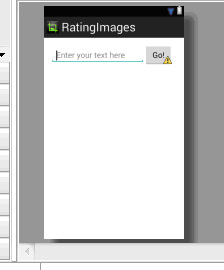
Теперь поменяем надпись на кнопке на "**Go!**" с помощью ссылки на ресурс в XML-коде главной активности и добавления одного ресурса в файл**strings.xml**:



**Рис. 6.16.**Редактор XML-кода файла strings.xml

Сохраните.

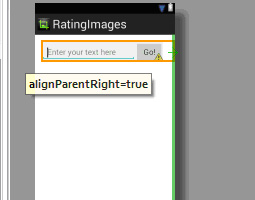
На графическом редакторе главной активности будут изменения:



**Рис. 6.17.**Кнопка приняла новую форму, в соответствии с надписью на ней

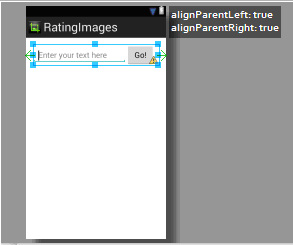
Теперь, когда мы поместили два главных представления на <LinearLayout> элемент, настало время добавить ещё два параметра для этого элемента.

Речь идет о "приращении" правого и левого краёв лейаута к правому и левому краям <RelativeLayout> элемента соответственно. А сделать это проще простого - просто потяните мышкой один край к другому!



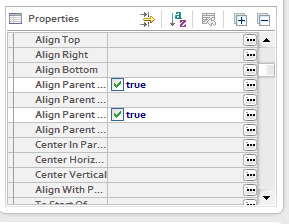
**Рис. 6.18.**"Приращение" правого края

Зелёные стрелки по краям дают понять, к какому элементу удалось прицепить край:



**Рис. 6.19.**<LinearLayout> после выравнивания

И, конечно, это отобразится в свойствах:



**Рис. 6.20.**Свойства <LinearLayout>

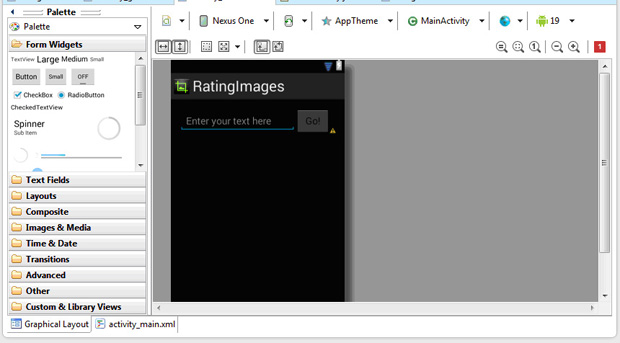
#### 6.2.4 Смена фона

Идём дальше - попробуем поменять фон.

Чтобы изменить цвет фона на чёрный, нужно в XML-коде главной активности написать одну строку в блоке <RelativeLayout> элемента:

android:background="#000000".

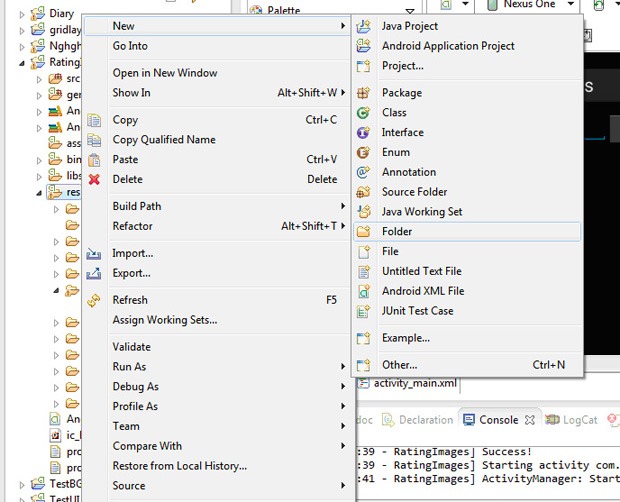
Сохраните и проверьте результат, открыв графический редактор.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_21.jpg)

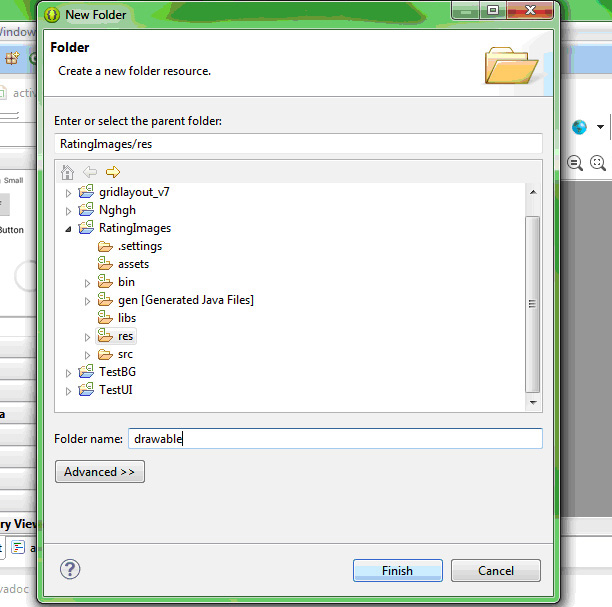
[увеличить изображение](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_21.jpg)  
**Рис. 6.21.**Фон стал чёрным

Красиво, но скучно. Как сделать фон ещё интереснее? Поместить на него рисунок!

Для этого сначала в папке **res/** создадим папку **drawable/**

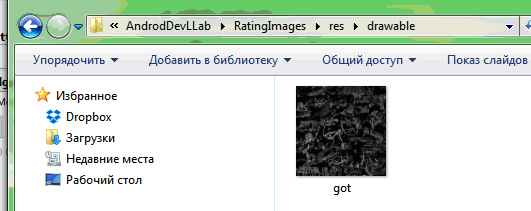
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_22.jpg)

**Рис. 6.22.**Создание папки



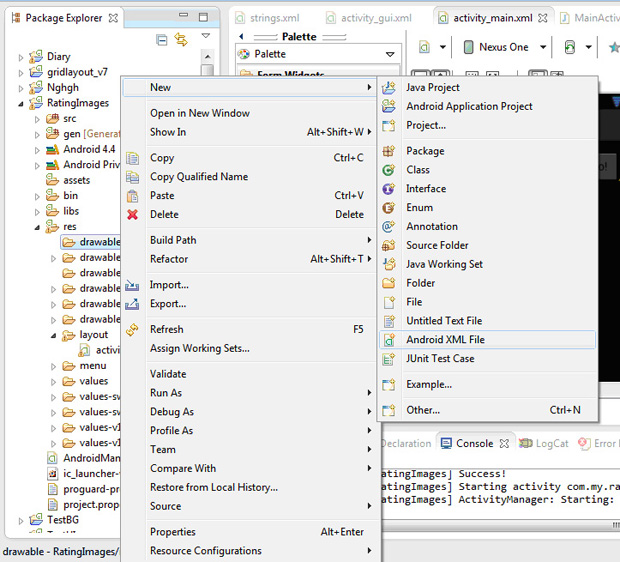
**Рис. 6.23.**Создание папки: имя папки

После того, как папка создана, нужно положить в эту папку изображение - картинка называется **got.png**:

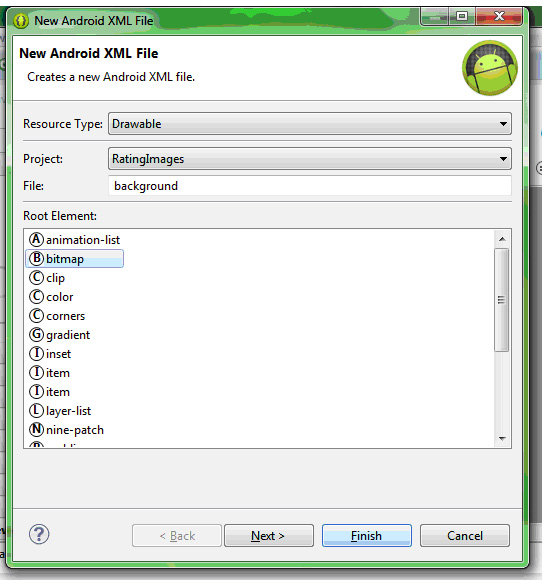


**Рис. 6.24.**Изображение в папке drawable/

После этого в папке **drawable/** нужно создать файл **background.xml**, важно при создании выбрать параметр bitmap.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_25.jpg)

**Рис. 6.25.**Создание нового XML-файла



**Рис. 6.26.**Создание нового XML-файла

Как только новый файл открылся, пропишем в него одну строчку, с указанием на то, откуда и какой файл использовать:

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

<bitmap xmlns:android="http://schemas.android.com/apk/res/android"

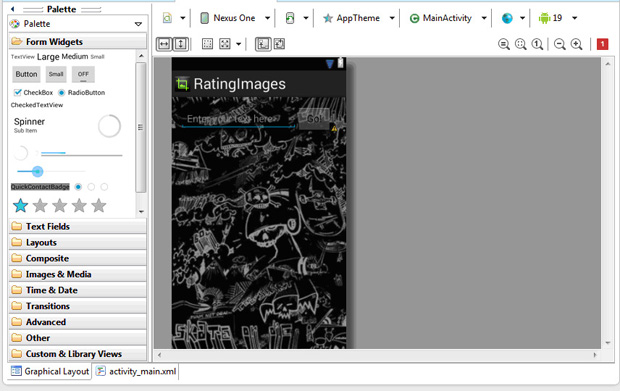
android:src="@drawable/got">

</bitmap>

Вернемся в редактор XML-кода, туда, где прописывали цвет фона.

Вместо строки android:background="#000000" напишем ссылку на XML-файл android:background="@drawable/background".

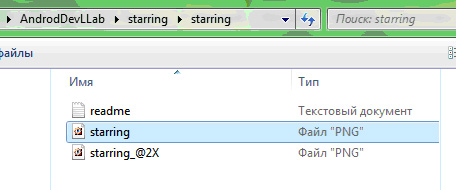
Сохраняем и видим результат:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_27.jpg)

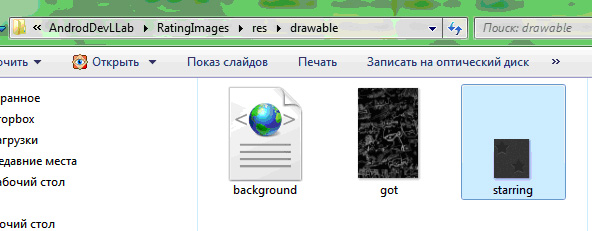
**Рис. 6.27.**Новый фон

Несомненно, фон смотрится хорошо, но очевидно, что кнопка и поле ввода просто затерялись, а это значит, что для этого приложения такой фон не подходит. Можно продолжить подбирать изображения на фон, но лучше создать черепичную заливку небольшим изображением. На[этом](http://subtlepatterns.com/) сайте можно найти узор на любой вкус!

Когда вы выбрали узор и скачали его, скопируйте изображение в папку **drawable/**.



**Рис. 6.28.**Копирование изображения



**Рис. 6.29.**Скопированное изображение

Теперь немного изменим файл **background.xml**.

Во-первых, нужно изменить имя изображение со старого на новое.

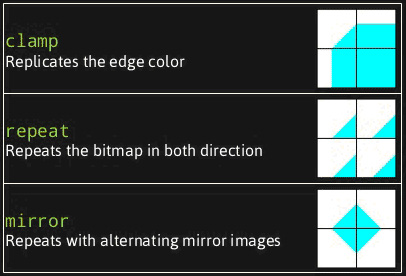
android:src="@drawable/starring"

Во-вторых, добавим такую строчку:

android:tileMode="repeat"

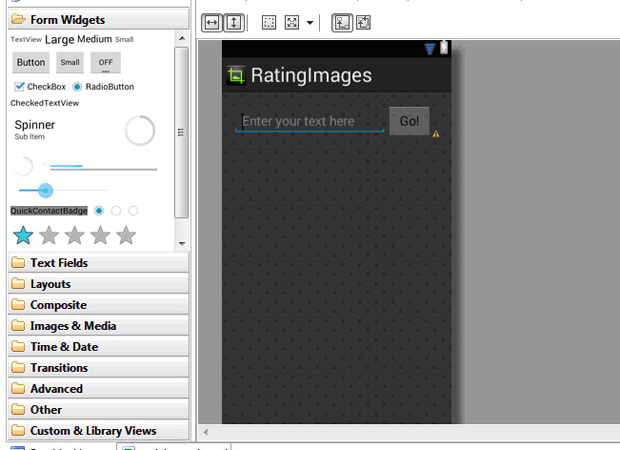
Сохраняем.

Атрибут android:tileMode задает тип заполнения, в данном случае простое повторение исходного изображения. Кроме repeat возможны варианты clamp и mirror. Помните, что данный приём применим только к bitmap, к фигурам, созданным при помощи XML, применить данную операцию нельзя.



**Рис. 6.30.**Варианты заполнения

Настало время посмотреть, что из этого получилось:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_31.jpg)

**Рис. 6.31.**Фон из звездочек

В <RelativeLayout> стоит добавлять android:background только в том случае, если вы хотите неподвижный фон, а в <ScrollView>чтобы фон прокручивался вместе с контентом.

Если вы выбрали тёмный фон, то стоит поменять цвет текста, вводимого в поле ввода, например на белый.

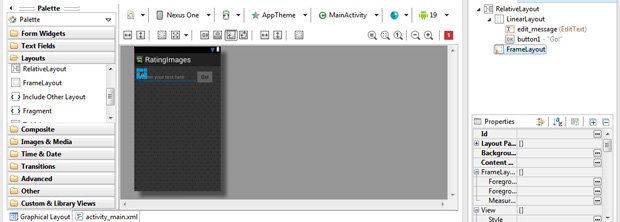
Для этого в блок <EditText> добавим строчку

android:textColor="#ffffff"

#### 6.2.5 Область просмотра изображений

Теперь займемся созданием области отображения изображений, которые пользователь будет оценивать.

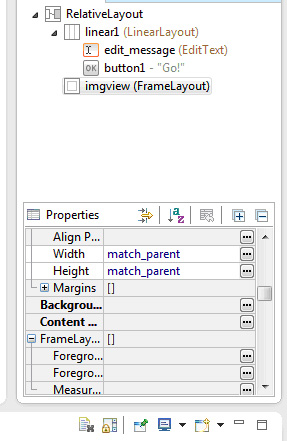
Добавьте "рамку" <FrameLayout> в <RelativeLayout>:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_32.jpg)

**Рис. 6.32.**Добавление "рамки"

Обратите внимание, что предупреждающий жёлтый треугольник исчез, но появился у нового элемента, и это значит, что всё идёт по плану.

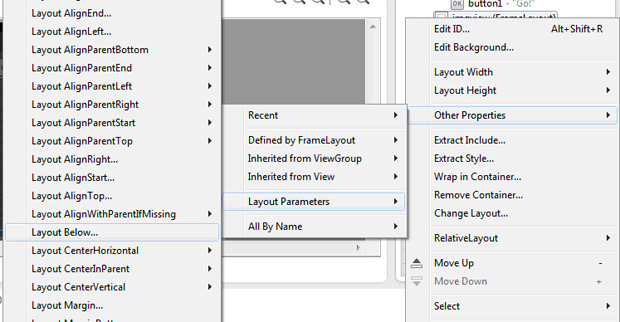
Укажем этому элементу ширину, высоту и id. Предупреждение должно пропасть.



**Рис. 6.33.**Свойства "рамки"

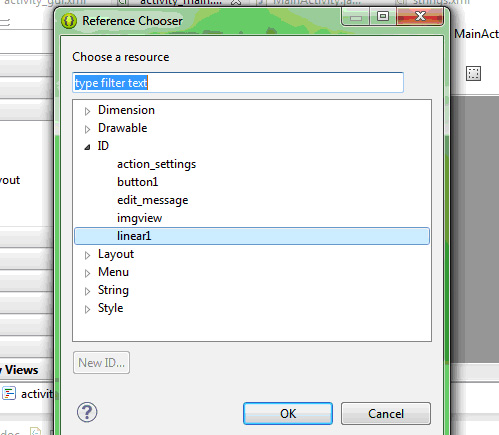
Сейчас **FrameLayout** "заезжает" на поле ввода и кнопку. Чтобы исправить это, нужно задать для "рамки" параметр, который будет следить, чтобы "рамка" находилась ниже, чем поле ввода и кнопка.

Выберите из списка **Outline** "рамку" и кликом правой кнопки мыши вызовите контекстное меню: **Other Properties -> Layout Parametrs -> Layout Below…**

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_34.jpg)

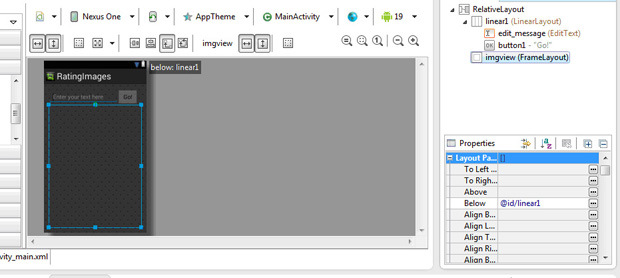
**Рис. 6.34.**Контекстное меню "рамки"

Появится окно. В нём выбираем из списка "**ID**" имя лейаута, на котором находятся поле ввода и кнопка:



**Рис. 6.35.**Выбор элемента, ниже которого должна быть "рамка"

Жмём **ОК** и "рамка" примет вид:

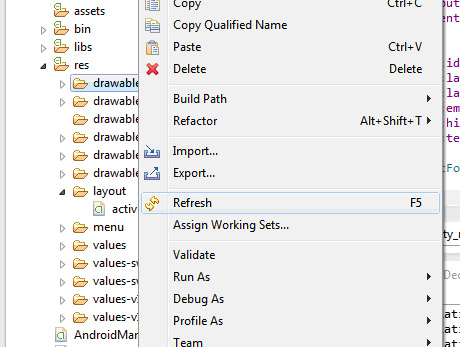
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_36.jpg)

**Рис. 6.36.**Новое расположение "рамки"

Теперь для наглядности поместим туда изображение.

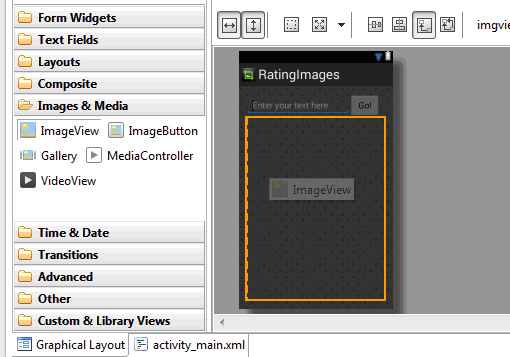
Сначала поместим само изображение в папку **res/drawable/** и обновим её.

Это нужно для того, чтобы новые данные загрузились в проект.



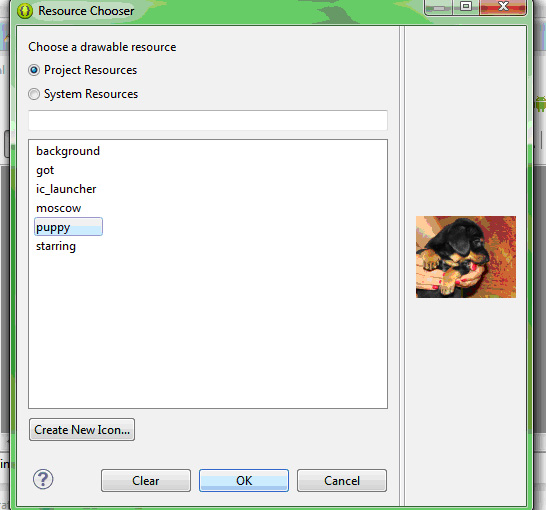
**Рис. 6.37.**Обновление папки

Теперь поместим внутрь "рамки" <ImageView>



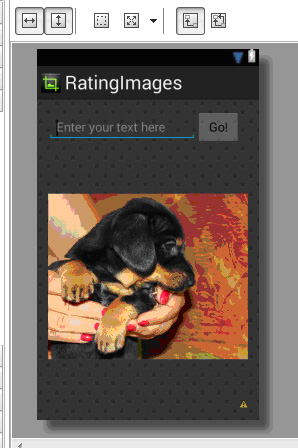
**Рис. 6.38.**Перемещение <ImageView>

В появившемся окне выбираем нужное нам изображение и жмём **OK**



**Рис. 6.39.**Выбор файла

Изображение должно появится на активности:



**Рис. 6.40.**Визуализация изображения

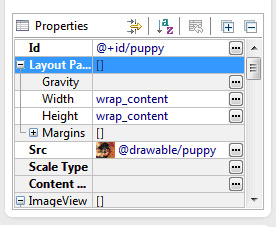
Вместе с изображением появился новый предупреждающий знак.

В описании предупреждения сказано: "**[Accessibility] Missing contentDescription attribute on image**". Давайте разберёмся, что это означает.

Атрибут android:contentDescription используется в программах, которые поддерживают специальные возможности для людей с нарушениями зрения и слуха. То есть, текст, прописанный в этом атрибуте, не будет показан на экране, но при включенной в "Специальных возможностях" услуге "звуковые подсказки" этот текст будет проговариваться, когда пользователь переходит к этому элементу управления.

Этот атрибут можно задать для ImageButton, ImageView и CheckBox, и задавать его или нет - дело каждого.

Свойства у изображения должны быть следующими:



**Рис. 6.41.**Свойства изображения

#### 6.2.6 Кнопки "like" и "dislike"

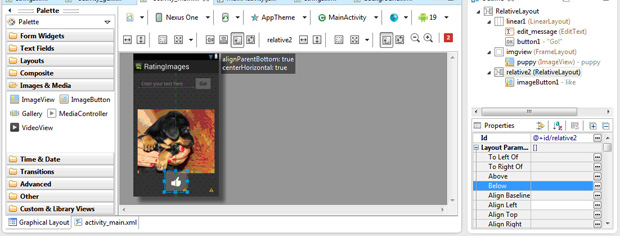
Пришло время создать кнопки оценивания.

Для этого добавьте на форму <RelativeLayout>, задайте для него ширину и высоту wrap\_content, и укажите id.

Снова в папку **res/drawable/** нужно добавить файлы. Найдите изображения "Палец вверх" и "Палец вниз", и поместите их в эту папку, после чего обновите её.

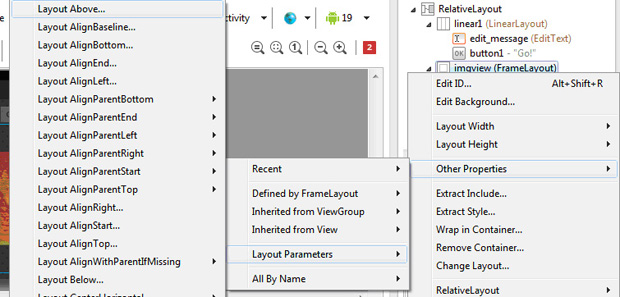
Изображения и другие полезные файлы можно скачать [здесь](http://developer.android.com/design/downloads/index.html).

Добавьте <ImageButton>, выберите изображение "Палец вверх", и переместите <RelativeLayout> в такое положение:

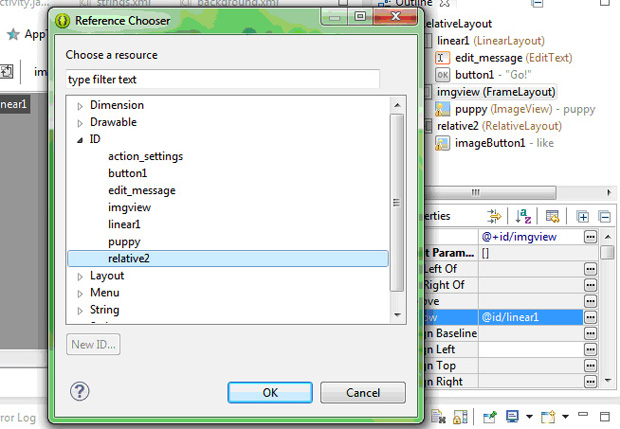
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_42.jpg)

**Рис. 6.42.**Новая кнопка

Укажите "рамке" быть выше, чем поле с кнопкой оценки:

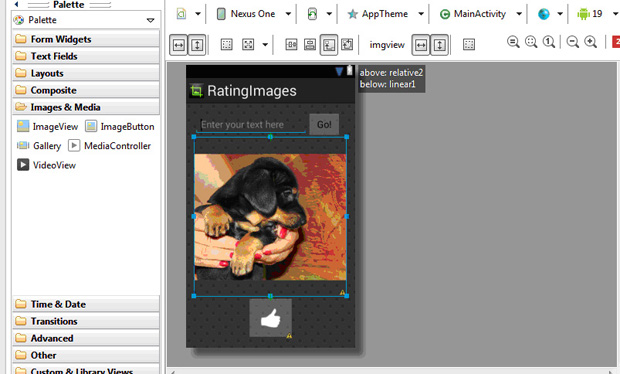
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_43.jpg)

**Рис. 6.43.**Контекстное меню "рамки"

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_44.jpg)

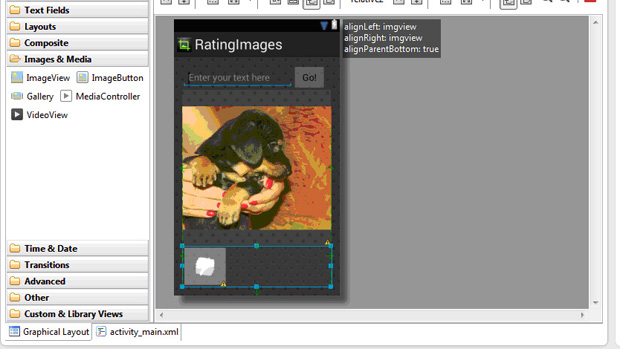
**Рис. 6.44.**Выбор элемента, выше которого должна быть "рамка"

В результате должно получиться следующее:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_45.jpg)

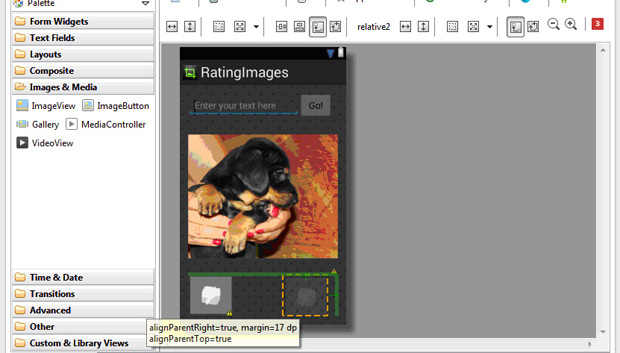
**Рис. 6.45.**Новое расположение элементов

Добавьте еще одну кнопку - кнопку "**Палец вниз**". Она "наложилась" на первую кнопку. Чтобы это исправить, проделайте с<RelativeLayout> то же самое, что и с <LinearLayout>: растяните элемент влево и вправо, до получения такого результата:

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_46.jpg)

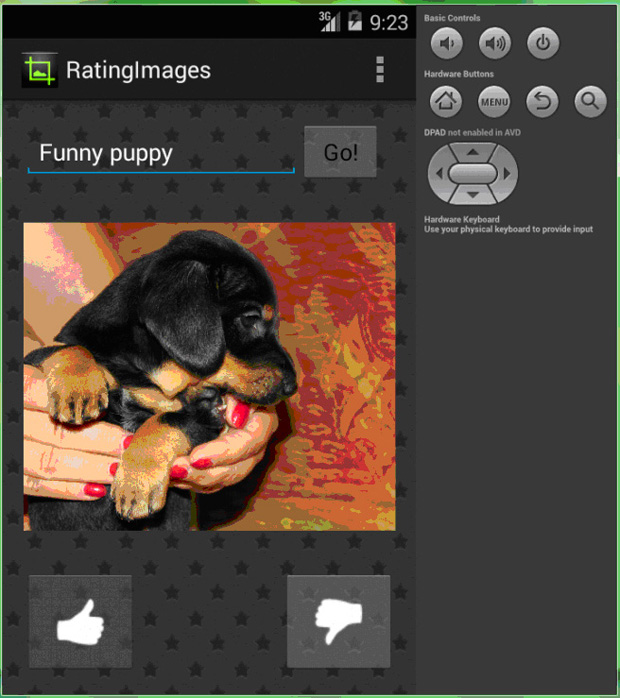
**Рис. 6.46.**Выравнивание области с кнопками

Теперь расставьте кнопки по краям так, чтобы они "прикрепились" к краям.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_47.jpg)

**Рис. 6.47.**Выравнивание кнопок

Готово! Теперь можно запустить эмулятор и посмотреть, что получилось.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_48.jpg)

**Рис. 6.48.**Главная активность

#### 6.2.7 Листинги

1)

<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: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"

android:background="@drawable/background">

<LinearLayout

android:id="@+id/linear1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:layout\_alignParentRight="true"

android:orientation="horizontal" >

<EditText

android:id="@+id/edit\_message"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="10"

android:hint="@string/edit\_message"

android:textColor="#ffffff" >

<requestFocus />

</EditText>

<Button

android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/button\_send"

/>

</LinearLayout>

<FrameLayout

android:id="@+id/imgview"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_above="@+id/relative2"

android:layout\_below="@id/linear1" >

<ImageView

android:id="@+id/puppy"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:src="@drawable/puppy"

/>

</FrameLayout>

<RelativeLayout

android:id="@+id/relative2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/imgview"

android:layout\_alignParentBottom="true"

android:layout\_alignRight="@+id/imgview" >

<ImageButton

android:id="@+id/imageButton2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:layout\_alignParentTop="true"

android:src="@drawable/dislike"

/>

<ImageButton

android:id="@+id/imageButton1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:layout\_alignParentTop="true"

android:src="@drawable/like"

/>

</RelativeLayout>

</RelativeLayout>

2)

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

<resources>

<string name="app\_name">RatingImages</string>

<string name="action\_settings">Settings</string>

<string name="hello\_world">Hello world!</string>

<string name="edit\_message">Enter your text here</string>

<string name="button\_send">Go!</string>

</resources>

3)

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

<bitmap

xmlns:android="http://schemas.android.com/apk/res/android"

android:src="@drawable/starring"

android:tileMode="repeat" >

</bitmap>

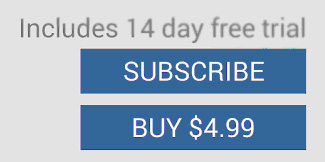
### 6.3 BuildingBlocks или элементы для построения интерфейса

*Среда разработки* "**AndroidIDE**" открывает широкий выбор готовых к использованию элементов для создания выдающихся приложений.

**Кнопка (Button)** состоит из текста и/или изображения, которые дают понять пользователю, что произойдёт, если нажать на эту кнопку. Важно помнить, что человек по своей природе привык взаимодействовать с объектами, поэтому фон у кнопки абсолютно не обязателен.



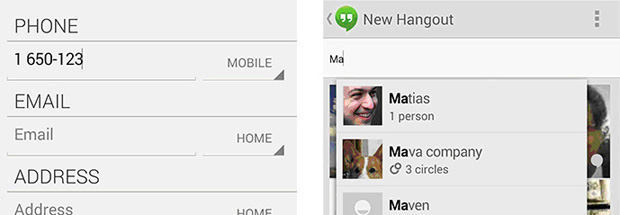
**Рис. 6.49a.**Кнопка-изображение



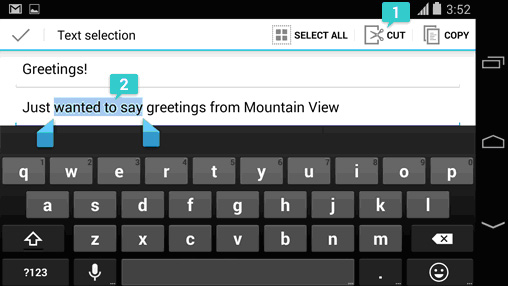
**Рис. 6.49b.**Кнопка-текст

**Текстовые поля (TextFields)**

Текстовые поля позволяют пользователю вводить текст в приложения. Они могут быть однострочными или многострочными. Одно касание текстового поля помещает *курсор* на *поле ввода* и автоматически выводит на экран клавиатуру. В *дополнение* к набору текста на клавиатуре текстовые поля позволяют выделять текст (вырезать, копировать, вставить). *Поиск* вариантов завершения слова помогает правописанию слов и упрощает *поиск* контактов в списке.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_50.jpg)

**Рис. 6.50.**50 Текстовые поля, поиск контактов



**Рис. 6.51.**Выделение текста

**Ползунки и слайдеры (Seek Bars and Sliders)**

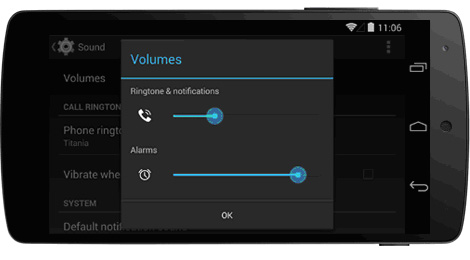
Интерактивные ползунки позволяют выбирать *значение* из непрерывного или дискретного диапазона значений путем перемещения ползунка. Наименьшее *значение* находится слева, наибольшее справа.



**Рис. 6.52.**Ползунки в светлой и темной темах

Интерактивный характер слайдера делает его удобным для настроек, которые отражают уровни интенсивности, такие как громкость, яркость, или насыщенность цвета.

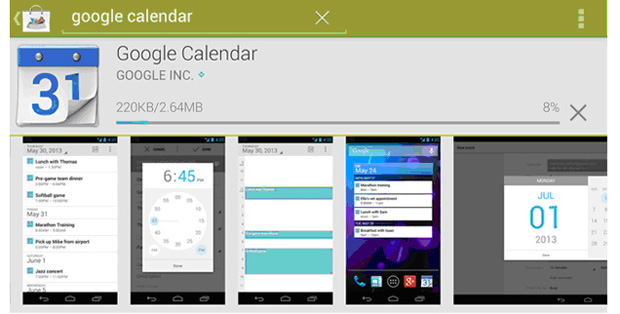
Так, например, уровень громкости можно регулировать кнопками на корпусе устройства или при помощи жеста пальцем по экрану.



**Рис. 6.53.**Настройка звука

**Прогресс-бары и активности (Progress&Activity)**

Прогресс-бары и показатели активности сигнализируют пользователям о происходящем в данный момент времени длительном действии, что означает для пользователя подождать завершения процесса некоторое время.

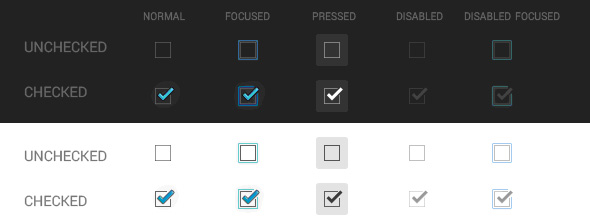
[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_54.jpg)

**Рис. 6.54.**Процесс загрузки приложения

**Переключатели (Switches)** позволяют пользователю выбирать параметры.

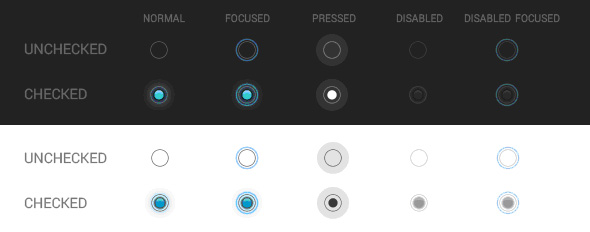
Есть три вида переключателей: флажки, радио-кнопки, и включение/выключение выключателей.

Флажки используются в том случае, если из предлагаемого списка можно выбрать одновременно несколько вариантов.



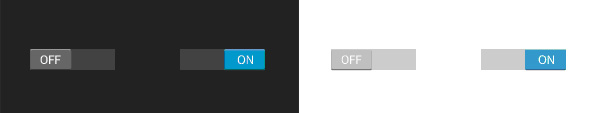
**Рис. 6.55.**Флажки

Радио-кнопки позволяют выбрать только один вариант из списка. Радио-кнопки формируются в группы.



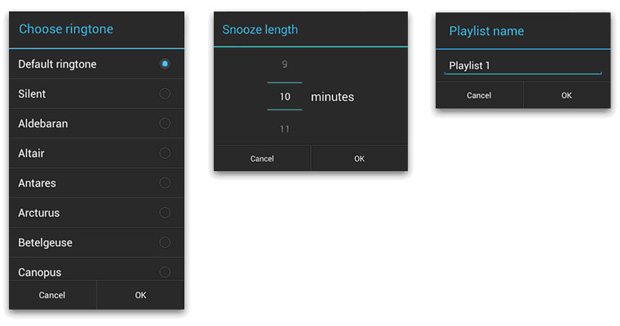
**Рис. 6.56.**Радио-кнопки

Выключатели дают возможность сделать флажок более наглядным, применив в качестве основы кнопку-значок, которая может фиксироваться в нажатом состоянии.



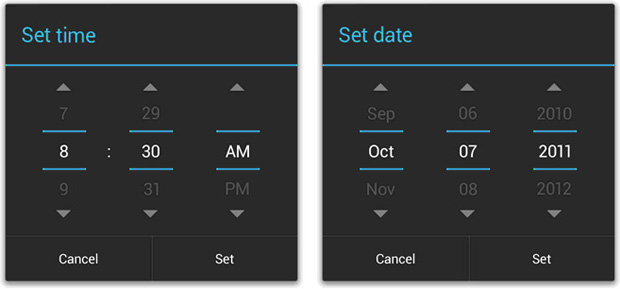
**Рис. 6.57.**Выключатели

**Диалоговые окна (Dialogs)** помогают приложению взаимодействовать с пользователем. Это может быть как банальный вопрос с двумя вариантами ответа (OK и Отмена), так и сложные окна с множеством настраиваемых пользователем параметров.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_58.jpg)

**Рис. 6.58.**Примеры диалоговых окон

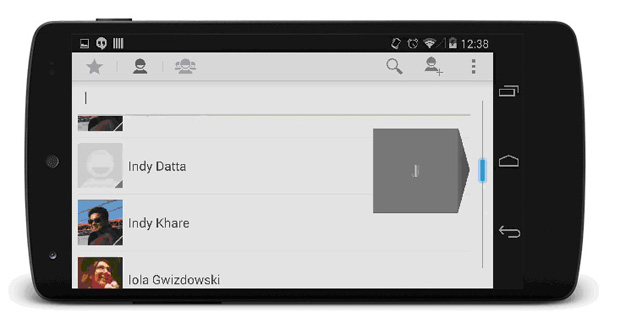
**Средство выбора (Pickers)** - это простой способ выбрать одно *значение* из набора путём последовательного перебора. Удобный перебор предлагаемых значений при помощи клика по стрелочкам или прокрутки, так же можно ввести *значение* с клавиатуры.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_59.jpg)

**Рис. 6.59.**Примеры пикеров

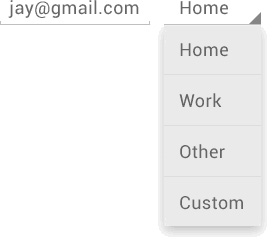
**Прокрутка (Scrolling)**, как интуитивно ожидаемый элемент, уже давно плотно сидит в головах у разработчиков и пользователей. Плавное или быстрое перемещение по содержимому экрана и контенту, который "вылез" за рамки экрана, осуществляется прокруткой, достаточно лишь провести пальцем по экрану в нужную сторону с желаемой скоростью.

Для того чтобы *индикатор* прокрутки не занимал *место* у рабочей области экрана, после приостановки пользования прокруткой, *индикатор*исчезает, и появляется снова, стоит только начать "прокручивать" пальцем. Для удобства поиска чего-либо в алфавитном списке, рядом с индикатором прокрутки возникает *индекс* с указанием на то, в каком разделе находится *индикатор*.

[](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_60.jpg)

**Рис. 6.60.**Индикатор и индекс прокрутки

**Выпадающий список (Spinner)** обеспечивает удобный способ выбора одного значения из набора.



**Рис. 6.61.**Спиннер для выбора типа e-mail адреса

**Вкладки (Tabs)** позволяют легко просматривать и переключаться между различными окнами или функциональными элементами приложения. Вкладки, прокручиваемые движением пальца (ScrollableTabs), делают этот процесс еще прощё и приятнее.

Основные вкладки (FixedTabs) удобны при отображении трех и менее вкладок, поскольку ширина вкладок фиксируется по самому длинному названию вкладки.

[Вкладки](http://www.intuit.ru/EDI/14_06_16_2/1465856498-31733/tutorial/1258/objects/6/files/06_62.jpg)

**Рис. 6.62.**Вкладки

**Списки (Lists)** удобны для представления нескольких позиций информации в вертикальном расположении.