Министерство образования и науки Российской Федерации

ФГАОУ ВО «Севастопольский государственный университет»

Институт информационных технологий

и управления в технических системах

Лабораторная работа №3

«Работа с ресурсами в ОС Android»

по дисциплине «Мобильные информационные технологии»

для студентов всех форм обучения направления подготовки

09.03.02 «Информационные системы и технологии»

Севастополь

2018

Лабораторная работа №3

Работа с ресурсами в AndroidStudio

**1. Цель работы**

Изучить организацию хранения различных типов **ресурсов**приложения Android и исследовать возможности их использования.

**2. Краткие теоретические сведения**

Ресурсы – один из основных компонентов приложения, представляющий из себя файлы разметки, отдельные строки, звуковые файлы, файлы изображений и т.д. На рисунке 2.1 представлена часть существующих ресурсов. Стоит отметить, что система постоянно обновляется и добавляются новые типы.

Изображение выглядит как снимок экрана

Описание создано с очень высокой степенью достоверности

Рисунок 2.1 – Типы ресурсов Android

В Android принято держать ресурсы за пределами исходного кода. Система поддерживает хранение ресурсов во внешних файлах, что облегчает их поддержку, обновление и редактирование. Также использование внешних файлов позволяет создавать альтернативные ресурсы для разных языков или размеров экрана, что очень важно, поскольку появляется все больше устройств с различными конфигурациями, работающих на платформе Android.

Каждое приложение на Android содержит каталог для ресурсов res и каталог для активов assets. Реальное различие между ресурсами и активами заключается в следующем:

информация в каталоге ресурсов будет доступна в приложении через класс R, который автоматически генерируется средой разработки. То есть хранение файлов и данных в ресурсах (в каталоге res) делает их легкодоступными для использования в коде программы;

для чтения информации, помещенной в каталог активов assets (необработанный формат файла), необходимо использовать AssetManager для чтения файла как потока байтов.

То есть реальное основное различие между ресурсами и активами заключается в том, что в каталоге res каждому файлу предоставляется предварительно скомпилированный ID, по которому можно легко получить доступ через R.id. Это полезно для быстрого и простого доступа к изображениям, звукам, значкам и т.д.

Каталог assets больше похож на файловую систему и предоставляет больше свободы для размещения любого файла. Доступ к каждому из файлов в этой системе можно получить так же, как при доступе к файлу в любой файловой системе через Java. Этот каталог хорош для таких вещей, как детали игры, словари и т.д.

Ресурсы в Android являются декларативными. В основном ресурсы хранятся в виде XML-файлов в каталоге res.

Для удобства система создает идентификаторы ресурсов и использует их в файле R.java. Статический класс R генерируется на основе ваших заданных ресурсов и создается во время компиляции проекта. Так как файл R генерируется автоматически, то нет смысла его редактировать вручную, потому что все изменения будут утеряны при повторной генерации.

В общем виде ресурсы представляют собой файл (например, изображение) или значение (например, заголовок программы), связанные с создаваемым приложением. Имена файлов для ресурсов должны состоять исключительно из букв в нижнем регистре, чисел и символов подчеркивания.

В Android используются два подхода к созданию ресурсов:

ресурсы задаются в файле, при этом имя файла значения не имеет;

ресурс задается в виде самого файла.

Общая структура каталогов, содержащих ресурсы, представлена на рисунке 2.2. Перечисление основных ресурсов с кратким описанием приведено в Приложении Б.

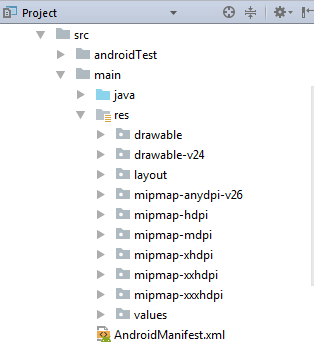


Рисунок 2.2 – Общая структура каталогов, содержащих ресурсы

**2.2 Применение ресурсов**

Существует два способа доступа к ресурсам: в файле исходного кода и в файле xml.

**Получение доступа в файле исходного кода**

Тип ресурса в данной записи ссылается на одно из пространств (внутренних классов), определенных в файле R.java, которые имеют соответствующие им типы в xml:

R.drawable (ему соответствует тип в xml drawable)

R.id (id)

R.layout (layout)

R.string (string)

R.attr (attr)

R.plural (plurals)

R.array (string-array).

Например, для установки ресурса activity\_main.xml в качестве графического интерфейса в коде MainActivity в методе onCreate() есть такая строка:

setContentView(R.layout.activity\_main);

Через выражение R.layout.activity\_main можно сослаться на ресурс activity\_main.xml, где layout – это тип ресурса, а activity\_main – имя ресурса.

**Получение доступа в файле xml**

Ссылки на ресурсы в файлах xml имеют следующую формализованную форму:

@[имя\_пакета:]тип\_ресурса/имя\_ресурса ,

где «имя\_пакета» представляет имя пакета, в котором ресурс находится (указывать необязательно, если ресурс находится в том же пакете);

«тип\_ресурса» представляет подкласс, определенный в классе R для типа ресурса;

«имя\_ресурса» – имя файла ресурса без расширения или значение атрибута android:name в XML-элементе (для простых значений).

**Метод getResources()**

Для получения ресурсов в классе Activity может быть использован метод getResources(), который возвращает объект android.content.res.Resources. Но чтобы получить сам ресурс, необходимо у полученного объекта Resources вызвать один из следующих методов (представлена только часть методов):

getString(): получает строку из файла strings.xml по числовому идентификатору;

getDimension(): получает числовое значение – ресурс dimen;

getDrawable(): получает графический файл

getBoolean(): получает значение Boolean.

**2.3 Типы ресурсов**

**Идентификаторы**

Этот тип ресурсов формируется, как правило, автоматически, и программисты даже не обращают на него внимания. Когда на форму добавляется новый элемент, с которым предполагается взаимодействовать в программе, то ему нужно присвоить идентификатор. Как правило, это происходит в виде @+id/editText1 (часто это происходит автоматически). Знак плюса обозначает, что если идентификатора не существует, то его нужно создать в классе R.

Можно также заранее создать ресурс типа item для задания id, не связанного ни с каким конкретным ресурсом. Обычно идентификаторы размещают в отдельном файле res/values/ids.xml.

**Псевдонимы (alias)**

Чтобы избежать дублирования ресурсов, можно использовать псевдонимы, которые будут ссылаться на один и тот же ресурс. Предположим, что было создано два файла res/layout-land/activity\_main.xml и res/layout-large/activity\_main.xml с одинаковым содержанием для разметки с альбомной ориентацией для смартфонов и планшетов. Вместо этих файлов можно создать два файла res/values-land/refs.xml и res/values-large/refs.xml, со ссылкой на файл с именем res/layout/activity\_main\_horizontal.xml, содержание которого идентично res/layout-land/activity\_main.xml (res/layout-large/activity\_main.xml):

<resources>

<item type="layout"

name="activity\_main">@layout/activity\_main\_horizontal</item>

</resources>

**Строковые ресурсы (strings)**

XML-файлы, представляющие собой ресурсы строк, находятся в проекте в папке res/values. По умолчанию ресурсы строк находятся в файле strings.xml. В самом простом виде этот файл определяет один ресурс "app\_name", который устанавливает название приложения. Данный файл можно редактировать, добавляя новые строковые ресурсы. Также можно создавать новые файлы, которые будут содержать строковые ресурсы, например, strings2.xml, catnames.xml и т.д. В большинстве случаев программисты используют для строковых ресурсов стандартное имя strings.xml.

Типичный файл выглядит следующим образом:

<resources>

<string name="hello">Здравствуй, Мир!</string>

<string name="app\_name">Hello, World</string>

</resources>

При создании или обновлении файла со строковыми ресурсами среда разработки автоматически создает или обновляет класс R.java, сообщая уникальные ID для определенных в файле строковых ресурсов. Если открыть данный файл, то можно найти там определенные ранее ресурсы в следующем виде:

public static final class string {

public static final int hello = 0x7f040000;

public static final int app\_name = 0x7f040001;

}

Значения констант app\_name и hello представляют собой числовой идентификатор ресурса. В исходном коде можно ссылаться на эти ресурсы, используя следующий формат: R.string.hello.

 ОС Android самостоятельно сопоставляет числовые идентификаторы с соответствующими ресурсами строк. Например:

String application\_name =  getResources().getString(R.string.app\_name);

Либо в xml-файле:

@string/app\_name

Если планируется использование нескольких файлов с ресурсами, то необходимо следить за уникальностью создаваемых имен. Не выйдет ничего хорошего, если в двух файлах будет, к примеру, одна и та же переменная app\_name.

Также стоит отметить, что пробелы в начале и в конце строк обрезаются. Если все же нужно использовать пробелы, то можно поместить строку в кавычках, либо же использовать код \u0020 вместо пробела.

Рассмотрим пример использования строковых ресурсов.

Изменим содержимое файла res/values/strings.xml:

<resources>

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

    <string name="message">Hello Android Nougat!</string>

</resources>

Теперь используем ресурс в файле activity\_main.xml:

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

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

    android:id="@+id/activity\_main"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    <TextView

        android:id="@+id/welcome"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="@string/message" />

</RelativeLayout>

Тут атрибуту android:text с помощью выражения @string/message передается значение из ресурса.

Аналогично можно использовать ресурс в коде Activity:

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        // получаем элемент textView

        TextView textView = (TextView) findViewById(R.id.welcome);

        // получаем ресурс

        String message = getResources().getString(R.string.message);

        // переустанавливаем у него текст

        textView.setText(message);

    }

}

Некоторые особенности использования строковых ресурсов:

строки, находящиеся в кавычках, необходимо явно помещать в еще одни кавычки;

возможно использование стандартных последовательностей Jаvа, предназначенных для форматирования строк;

нельзя использовать символы « & », « < » в явном виде. Задать эти символы можно используя специальную последовательность «&lt», «&amp» соответственно.

если текст содержит html-теги, и в нем встречается неразрывный пробел «&nbsp;», то его надо заменить на «&#160;»;

возможно использование простых элементы HTML, предназначенных для форматирования (теги <b> (полужирный шрифт), <i> (наклонный шрифт) и т.д.).

Пример использования строки форматирования:

<resources>

<string name="java\_format\_string">Привет %2$s. Здравствуй %1$s</string>

</resources>

//считывание строки форматирования Java

String javaFormatString =

getResources().getString(R.string.java\_format\_string);

// преобразование отформатированной строки при помощи данных,

// передаваемых в аргументах

String substitutedString = String.format(javaFormatString,

"Рыжик", "Барсик" );

// помещение вывода в текстовый вид

TextView tvText;

tvText = (TextView) findViewById(R.id.textView1);

tvText.setText(substitutedString);

Пример использования строки НТМL для оформления текста:

<resources>

<string name="tagged\_string"><b>Рыжик</b> - <i>мой любимый кот</i></string>

</resources>

// считывание строки html-строки из ресурса и помещение ее в текстовый вид

String htmlTaggedString = getResources().getString(R.string.tagged\_string);

// преобразование строки во фрагмент текста,

// который может быть помещен в текстовом виде;

// класс android.text.Html допускает рисование строк «html» (но не все теги)

Spanned textSpan = android.text.Html.fromHtml(htmlTaggedString);

TextView tvText;

tvText = (TextView) findViewById(R.id.textView2);

tvText.setText(textSpan);

**Ресурсы цветов (colors)**

Цвет определяется с помощью элемента <color>. Его атрибут name устанавливает название цвета, которое будет использоваться в приложении, а шестнадцатеричное число - значение цвета.

Для задания цветовых ресурсов можно использовать следующие форматы:

#RGB (#F00 - 12-битное значение);

#ARGB (#8F00 - 12-битное значение с добавлением альфа-канала);

#RRGGBB (#FF00FF - 24-битное значение);

#AARRGGBB (#80FF00FF - 24-битное значение с добавлением альфа-канала);

Обычно для цветовых ресурсов используют файл colors.xml в подкаталоге res/values. Но можно использовать любое произвольное имя файла, или даже вставить их в файл вместе со строковыми ресурсами strings.xml. Android прочтет все файлы, а затем обработает их, присвоив им нужные ID.

Для получения цвета применяется метод getColor(), который в качестве параметра принимает идентификатор цветового ресурса.

Пример использования:

<resources>

<color name="red">#f00</color>

<color name="yellow">#FFFF00</color>

<color name="green">#FF00FF00</color>

</resources>

//программное использование

Resources res = getResources(); // получим объект для работы с ресурсами

int title\_color = res.getColor(R.color.red); // получим значение цвета,

// заданного в ресурсах

//использование в xml-файле

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/anytext"

android:color="@color/red" />

**Ресурсы размеров (dimens)**

В Android используются следующие единицы измерения: пикселы, дюймы, точки. Все они могут входит в состав ХМL-шаблонов и кода Jаvа. Данные единицы измерения также можно использовать как ресурсы при помощи тега <dimen> (обычно используют файл dimens.xml), который в качестве значения принимает некоторый размер в одной из принятых единиц измерения (dp, sp, pt, px, mm, in).

Ресурсы dimension используются для таких атрибутов визуальных элементов, которые в качестве значения требуют указание числового значения. Например, атрибут android:layout\_height или android:textSize.

Для получения ресурсов в коде java применяется метод getDimension() класса Resources.

Пример использования:

<resources>

<dimen name="activity\_horizontal\_margin">16dp</dimen>

<dimen name="activity\_vertical\_margin">16dp</dimen>

<dimen name="text\_size">16sp</dimen>

</resources>

//программное использование

float textSize = getResources().getDimension(R.dimen.text\_size);

int leftPadding =

(int)getResources().getDimension(R.dimen.activity\_horizontal\_margin);

int topPadding =

(int)getResources().getDimension(R.dimen.activity\_vertical\_margin);

//использование в xml-файле

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textSize="@dimen/text\_size"/>

Если необходимо динамически перейти от одних единиц измерения к другим, например, от dip к пикселям, то можно использовать класс android.util.TypedValue. С помощью метода TypedValue.applyDimension() осуществляется перевод из одних единиц в другие. Первый параметр – тип исходных единиц измерения (в данном случае dip), второй параметр – количество для конвертации, а третий – параметр DisplayMetrics, который позволяет произвести конвертацию.

Информацию об иных типах ресурсов можно найти на официальном сайте разработчика или из других источников.

**3. Постановка задачи**

3.1. Изучить теоретические сведения об использовании ресурсов в ОС Android.

3.2. Модифицировать проект приложения, разработанного в предыдущих лабораторных работах (либо же создать новое произвольное приложение), задействовав нем ресурсы в соответствии с вариантом задания, представленном в Приложении А.

**4. Вопросы для самоподготовки**

4.1. Что представляют из себя ресурсы в приложении Android? Перечислите известные Вам типы ресурсов.

4.2. Зачем использовать ресурсы? Есть ли в их использовании какие-либо плюсы?

4.3. Поясните различие между ресурсами и активами.

4.4. Каким образом можно получить доступ к ресурсам в файле исходного кода?

4.5. Каким образом можно получить доступ к ресурсам в файле xml?

4.6. В каких случаях стоит использовать псевдонимы?

4.7.

ПРИЛОЖЕНИЕ А

Варианты заданий

|  |  |
| --- | --- |
| № варианта | Тип ресурса, который требуется использовать в проекте |
| 1 | [Property](http://developer.alexanderklimov.ru/android/theory/resources.php#bool) Animation;  Plurals |
| 2 | [Colors](http://developer.alexanderklimov.ru/android/theory/resources.php#color);  Dimensions |
| 3 | [Layer List;](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/381-urok-163-grafika-drawable-bitmap-layer-list-state-list.html)  Strings |
| 4 | [Tween](http://developer.alexanderklimov.ru/android/theory/resources.php#dimen) Animation;  Transition Drawable |
| 5 | [Level List;](http://startandroid.ru/ru/uroki/vse-uroki-spiskom/392-urok-164-grafika-drawable-level-list-transition-inset-clip-scale.html)  Styles |
| 6 | Inset Drawable;  Nine Patch Images |
| 7 | Shape Drawable;  Integers |
| 8 | [Typed Array](https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/resources/more-resources.html#TypedArray);  Bool |
| 9 | Style;  Menu |
| 10 | State List;  String Array |
| 11 | Raw resources;  Mipmap |
| 12 | Integer Array;  Clip Drawable |

**Примечание:**

Обязательно требуется использование ресурсов как в коде приложения, так и в xml-файлах.

ПРИЛОЖЕНИЕ Б

Основные ресурсы Android

Таблица Б.1 – Основные ресурсы Android

|  |  |  |
| --- | --- | --- |
| Тип ресурса | Размещение | Описание |
| Цвета | res/values/имя\_файла | Идентификатор цвета. Указывает на цветовой код.  ID таких ресурсов выражаются в R.java как R.color.\*.  XML-узел: /resources/color |
| Строки | res/values/имя\_файла | Строковые ресурсы. В их число также входят строки в формате java и html.  ID таких ресурсов выражаются в R.java как R.string.\*.  XML-узел: resources/string |
| Меню | /res/menu/имя\_файла | Меню в приложении также можно задать как XML-ресурсы |
| Параметры | /res/values/имя\_файла | Параметры или размеры различных элементов. Поддерживает пикселы, дюймы, миллиметры, не зависящие от плотности экрана пикселы (dip) и пикселы, не зависящие от масштаба.  ID таких ресурсов выражаются в R.java как R.dimen.\*.  XML-узел: resources/dimen |
| Изображения | /res/drawable/ваши\_файлы | Ресурсы-изображения.  Поддерживает форматы JPG, GIF, PNG (самый предпочтительный) и др.  Каждое изображение является отдельным файлом и получает собственный идентификатор, который формируется по имени файла без расширения.  ID таких ресурсов представлены в файле R.java как R.drawable.\* |
| Отрисовываемые цвета | /res/values/ваш\_файл или /res/drawable/ваши\_файлы | Представляет цветные прямоугольники, которые используются в качестве фона основных отрисовываемых объектов, например точечных рисунков.  ID таких ресурсов выражаются в файле R.java как R.drawable.\*.  ХМL-узел для такого файла: /resources/drawable |

Таблица Б.2 – Продолжение таблицы Б.1

|  |  |  |
| --- | --- | --- |
| Тип ресурса | Размещение | Описание |
| Анимация | /res/anim/ваш\_файл | Android может выполнить простую анимацию на графике или на серии графических изображений.  Анимация включает вращения, постепенное изменение, перемещение и протяжение |
| Произвольные XML-файлы | /res/xml/\*.xml | В Android в качестве ресурсов могут использоваться произвольные XML-файлы. Они компилируются в aapt.  ID таких ресурсов выражаются в файле R.java как R.xml.\* |
| Произвольные необработанные ресурсы | /res/raw/\*.\* | Любые нескомпилированные двоичные или текстовые файлы, например, видео.  ID таких ресурсов выражаются в файле R.java как R.raw.\* |
| Произвольные необработанные активы | /assets/\*.\*/\*.\* | Можно использовать произвольные файлы в произвольно названных каталогах, которые находятся в подкаталоге /assets. Это не ресурсы, а просто необработанные файлы.  Для таких файлов не создаются идентификаторы ресурсов. При работе с ними нужно использовать относительное имя пути, начиная с /assets |