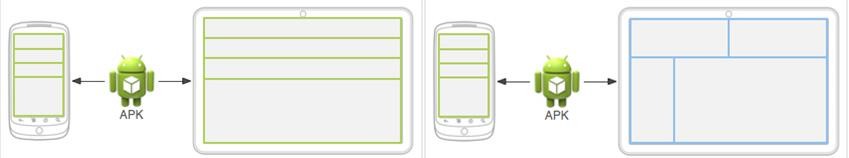
**ЛЕКЦІЯ 12. РЕСУРСИ ПРОЕКТУ**

**7.1 Поняття ресурсу. Типи ресурсів проекту**

При розробці мобільних додатків прийнято підхід, який полягає у відокремленні ресурсів, які використовує додаток, від коду. До ресурсів можуть належати: зображення, текстові стрічки, кольори, компонування елементів інтерфейсу користувача тощо. Відокремлення ресурсів від коду дозволяє використовувати альтернативні ресурси для різних конфігурацій пристроїв: мова, розширення екрану і т. д. Для забезпечення сумісності з різними конфігураціями, ресурси необхідно згрупувати в директорії за типом ресурсів і конфігурації пристрою, отримані директорії поміщуються в папку *res*/ проекту.

Для будь-якого типу ресурсів можна визначити дві групи (рис. 7.1).



а) б)

Рисунок 7.1 Варіант компонування інтерфейсу користувача, параметри якого зберігаються в ресурсах: а) компонування по замовчуванню;

б) специфічне компонування для певного пристрою

Перша визначає ресурси, які будуть використовуватися незалежно від конфігурації пристрою або в тому випадку, коли під конфігурацію немає відповідних альтернативних ресурсів (ресурси за замовчуванням). Друга група визначає ресурси, які призначені для певної конфігурації фізичного пристрою (альтернативні ресурси). Такі ресурси розміщуються в директорії із назвою, що позначає цю конфігурацію (рис. 7.2).

Для різних типів ресурсів, визначених у проекті, в каталозі *res/* створюються свої підкаталоги. Файли ресурсів не можна розміщувати в папку *res/* безпосередньо, оскільки буде отримано помилку компіляції. Підтримувані підкаталоги:

 *animator*/: xml-файли, що визначають анімацію;

 *anim*/: xml-файли, що визначають tween-анімацію;

 *color*/: xml-файли, що визначають кольори;

 drawable/: графічні файли (.png, .jpg, .gif);

 *mipmap*/: графічні файли, які використовуються для іконок додатку під різні розширення екрану;

 *layout*/: xml-файли, що визначають інтерфейс користувача програми;

 *menu*/: xml-файли, що визначають меню програми;

 *raw*/: файли, які зберігаються в бінарному («сирому») вигляді;

 *values*/: xml-файли, які містять різні використовувані в додатку значення, наприклад, ресурси стрічок;

 *xml*/: довільні xml-файли.



Рисунок 7.2 Можливий варіант розміщення ресурсів проекту, специфікованих для різних розширень екрану фізичного пристрою

Загалом в проекті можна використовувати такі типи ресурсів (табл. 7.1):

**Типи ресурсів Android-проекту**

**Таблиця 7.1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ресурс** | **Каталог проекту** | **Файл** | **Елемент у файлі** |
| **1** | **2** | **3** | **4** |
| Стрічки | /res/values/ | strings.xml | <string> |
| Plurals | /res/values/ | strings.xml | <plurals> |
| Масиви стрічок | /res/values/ | strings.xml або arrays.xml | <string-array> |

Продовження Таблиці 7.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Ресурс** | **Каталог**  **проекту** | **Файл** | **Елемент у файлі** |
| **1** | **2** | **3** | **4** |
| Логічні значення | /res/values/ | bools.xml | <bool> |
| Кольори | /res/values/ | colors.xml | <color> |
| Списки кольорів | /res/color/ | довільне найменування | <selector> |
| **1** | **2** | **3** | **4** |
| Розміри (Dimensions) | /res/values/ | dimens.xml | <dimen> |
| Ідентифікатори ID | /res/values/ | ids.xml | <item> |
| Цілі числа | /res/values/ | integers.xml | <integer> |
| Масиви цілих чисел | /res/values/ | integers.xml | <integer-array> |
| Графічні файли | /res/drawable/ | файли із розширенням  jpg та png |  |
| Tween-анімація | /res/anim/ | файл xml із довільним  найменуванням | <set>, <alpha>,  <rotate>, <scale>,  <translate> |
| Покадрова анімація | /res/drawable/ | файл xml із довільним  найменуванням | <animation-list> |
| Анімація  властивостей | /res/animator/ | файл xml із довільним  найменуванням | <set>,  <objectAnimator>,  <valueAnimator> |
| Меню | /res/menu/ | файл xml із довільним  найменуванням | <menu> |
| XML-файли | /res/xml/ | файл xml із довільним  найменуванням |  |
| Бінарні ресурси | /res/raw/ | файли мультимедіа  (mp3, mp4), текстові та інші файли |  |
| Розмітка графічного  інтерфейсу | /res/layout/ | файл xml із довільним  найменуванням |  |
| Стилі та теми | /res/values/ | styles.xml, themes.xml | <style> |

98

По замовчуванню в каталозі *res/* є підкаталоги не для всіх типів ресурсів, які можна використовувати в додатку, проте при необхідності ми можемо додати в папку *res/* потрібний каталог, а в нього потім помістити відповідний ресурс. Коли відбувається компіляція проекту відомості про всі ресурси додаються в спеціальний файл *R*.*java* проекту. У цьому файлі зберігаються всі оголошення ресурсів у вигляді числових констант. Наприклад, у проекті за замовчуванням є ресурс розмітки інтерфейсу – файл *activity\_main.xml* в папці *res*/*layout*. Для цього ресурсу в класі *R* буде створено приблизно таку константу:

public final class R {

................................

public static final class layout {

public static final int activity\_main=0x7f030001;

}

**7.2 Застосування ресурсів під час розробки додатків**

Існує два способи доступу до ресурсів: в файлі вихідного коду і в файлі xml: в коді програми та в тексті xml-файлів.

**7.2.1 Посилання на ресурси в коді програми**

Тип ресурсу в даному випадку посилається на один із просторів (внутрішніх класів), визначених у файлі R.java:

 *R.drawable (йому відповідає тип в xml drawable);*

 *R.id (id);*

 *R.layout (layout);*

 *R.string (string);*

 *R.attr (attr);*

 *R.plural (plurals);*

 *R.array (string-array).*

Наприклад, для встановлення ресурсу *activity\_main.xml* в якості графічного інтерфейсу в коді методу *onCreate*() класу *MainActivity* є такий рядок: setContentView(R.layout.activity\_main);

З допомогою виразу *R.layout.activity\_main* ми посилаємося на ресурс

*activity\_main.xml*, де *layout* - тип ресурсу, а *activity*\_main – ім’я ресурсу.

Подібним чином ми можемо посилатися на інші ресурси. Наприклад, у файлі

*res/values/strings.xml* визначено ресурс *app*\_*name*:

<resources>

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

</resources>

Щоб отримати посилання на даний ресурс в програмному коді ми можемо використовувати вираз *R.string.app\_name*.

**7.2.2 Посилання на ресурси у файлі xml**

Посилання на ресурси в файлах xml мають формат:

*@[ім’я\_пакета:]тип\_ресурсу/ім’я\_ресурсу*. Тут:

 *ім’я\_пакета* – імєя пакета, в якому знаходиться ресурс (вказувати необов’язково, якщо ресурс знаходиться в тому ж пакеті);

 *тип\_ресурсу* – представляє підклас, визначений у класі R для типу ресурсу;

 *ім’я\_ресурсу* – ім’я файлу ресурсу без розширення або значення атрибуту

*android:name* в XML-елементі (для простих значень).

Наприклад, ми хочемо вивести в елементі інтерфейсу *TextView* рядок, який визначено у вигляді ресурсу в файлі *strings.xml*:

<TextView android:id=["@+id](mailto:@+id)/welcome" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:tex[t="@strin](mailto:@string)g/app\_name" />

В даному випадку властивість *text* елементу *TextView* в якості значення буде отримувати значення стрічкового ресурсу *app\_name*.

**7.2.3 Метод getResources()**

Для отримання ресурсів в класі *Activity* ми можемо використовувати метод *getResources()*, який повертає об’єкт *android.content.res.Resources*. Але щоб отримати сам ресурс, слід в отриманого об’єкта *Resources* викликати один із наступних методів:

 *getString*(): повертає стрічку з файлу strings.xml по числовому ідентифікатору;

 *getDimension*(): повертає числове значення - ресурс dimension;

 *getDrawable*(): повертає графічний файл;

 *getBoolean*(): повертає значення boolean.

Наведемо простий приклад роботи з ресурсами із застосуванням методів об’єкта

*Resources*:

public class MainActivity extends AppCompatActivity {

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// отримання ресурсів із файлу values/strings.xml

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

TextView textView = new TextView(this); textView.setText(app\_name); setContentView(textView);

}

}

Тут в коді методу *onCreate()* використовуючи метод *getResources*().*getString()*

отримуємо найменування проекту із файлу *strings.xml,* яке надалі використовуємо для встановлення напису графічного елемента *textView*. При запуску програми ми побачимо, наприклад, таке:

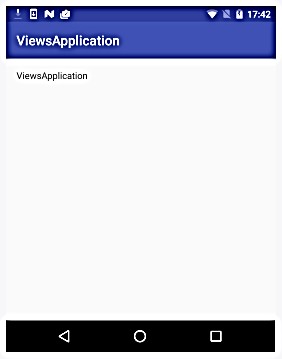


Рисунок 7.3 Результат роботи із текстовим ресурсом із застосуванням методів об’єкта *Resources*

**7.3 Робота із ресурсами основних типів**

**7.3.1 Стрічкові ресурси**

Стрічкові ресурси – один з найважливіших компонентів програми. Ми використовуємо їх при виведенні назви програми, різного тексту, наприклад, тексту кнопок і т. д. XML-файли, що представляють собою стрічкові ресурси, знаходяться в проекті в папці *res/values*. За замовчуванням ресурси даного типу знаходяться в файлі *strings.xml*, який може виглядати наступним чином:

<resources>

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

<string name="ButtonOk\_value">Вхід</string>

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

</resources>

Тобто, кожен окремий ресурс визначається за допомогою елемента *string*, а його атрибут *name* містить назву ресурсу. В коді додатку ми можемо посилатися на ці ресурси так: R.string.app\_name

ОС Android сама співставить дані числові ідентифікатори з відповідними ресурсами рядків. Наприклад в коді Java:

public class MainActivity extends AppCompatActivity {

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

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

// отримуємо ресурс

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

// встановлюємо текст елемента textView textView.setText(message);

}

}

А в xml-файлі можна отримати доступ до стрічкового ресурсу message так:

<TextView android:id=["@+id](mailto:@+id)/welcome" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content" android:tex[t="@strin](mailto:@string)g/message" />

**7.3.2 Ресурси типу dimension**

Оголошення ресурсів даного типу повинне знаходитися в папці res /values в

файлі з будь-яким довільним ім’ям. Загальний синтаксис:

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

<resources>

<dimen name="ім’я\_ресурса">використовуваний\_розмір</dimen>

</resources>

Тег *<dimen>* позначає ресурс і включає деяке значення розміру в одній з прийнятих одиниць виміру (dp, sp, pt, px, mm, in). Наприклад, якщо додати в Android Studio в папку *res/values* новий файл *dimens.xml* такого вмісту:

<resources>

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

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

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

</resources>

то ми визначимо два ресурси для відступів *activity\_horizontal\_margin* і *activity\_vertical\_margin*, які зберігають значення 16dp, і ресурс *text\_size*, який зберігає висоти шрифту в 16sp.

В файлі *activity\_main.xml* їх можна використати так:

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

<RelativeLayout xmlns:android="[http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:id=["@+id](mailto:@+id)/activity\_main"

android:layout\_width="match\_parent" android:layout\_height="match\_parent" android:paddingBotto[m="@dim](mailto:@dimen)en/activity\_vertical\_margin" android:paddingLeft="@dimen/activity\_horizontal\_margin" android:paddingRight=["@dimen/](mailto:@dimen)activity\_horizontal\_margin" android:paddingTop="@dimen/activity\_vertical\_margin">

<TextView android:textSize="@dimen/text\_size" android:layout\_width="wrap\_content"

android:text="Hello Android Nougat!" />

</RelativeLayout>

Ресурси типу *dimension* використовуються для таких атрибутів візуальних елементів, які в якості значення вимагають вказання конкретного числа.

Для отримання ресурсів в коді Java застосовується метод *getDimension*() класу

*Resources*. Наприклад:

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

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);

TextView textView=new TextView(this); textView.setText("Hello world!"); textView.setTextSize(textSize);

textView.setPadding(leftPadding, topPadding, leftPadding, topPadding);

setContentView(textView);

}

}

**7.3.3 Ресурси типу color**

Ресурси типу *color* повинні зберігатися в файлах каталогу *res/values* і так само як і ресурси стрічок, повинні бути вкладеними в тег *<resources>*. За замовчуванням при створенні найпростішого проекту в папку *res/values* додається файл *colors.xml*:

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

<resources>

<color name="colorPrimary">#3F51B5</color>

<color name="colorPrimaryDark">#303F9F</color>

<color name="colorAccent">#FF4081</color>

</resources>

Колір задається за допомогою елемента *<color>*. Його атрибут *name* задає назву кольору, яка буде використовуватися в додатку, а шістнадцяткове число – значення кольору. Для задання колірних ресурсів можна використовувати такі формати:

1) #RGB (#F00 – 12-бітне значення);

2) #ARGB (#8F00 – 12-бітне значення із додаванням альфа-каналу);

3) #RRGGBB (#FF00FF – 24-бітне значення);

4) #AARRGGBB (#80FF00FF – 24-бітне значення із додаванням альфа-каналу). Приклад застосування колірних ресурсів у файлі *activity\_main.xml:*

<TextView android:textSize="20dp" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:textColor=["@color/](mailto:@color)colorPrimary" android:background=["@color/t](mailto:@color)extViewColor" android:text="Hello world!" />

Тут за допомогою атрибута *android:textColor* встановлюється колір тексту в *TextView*, а атрибут *android:background* встановлює фон *TextView*. Для отримання значення кольору після ["*@color*](mailto:@color) */*" вказується ім’я ресурсу.

В коді *MainActivity* колірні ресурси можна застосовувати так:

public class MainActivity extends AppCompatActivity {

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

int textColor = ContextCompat.getColor(this, R.color.colorPrimary);

int backgroundColor = ContextCompat.getColor(this, R.color.textViewColor);

TextView textView = new TextView(this); textView.setText("Hello world!"); textView.setTextSize(20); textView.setPadding(16, 16, 16, 16); textView.setTextColor(textColor); textView.setBackgroundColor(backgroundColor);

setContentView(textView);

}

}

Тут застосовується метод *ContextCompat*.getColor(), який в якості першого

параметра приймає поточний об’єкт *Activity*, а в якості другого – ідентифікатор колірного ресурсу.

**7.3.4 Ресурси зображень**

Android підтримує такі формати файлів: .*png* (бажаний), .*jpg* (допустимий),

.*gif* (небажаний). Для графічних файлів в проекті за замовчуванням створюється папка *res/drawable*. При додаванні графічних файлів в цю папку для кожного з них Android створює ресурс *Drawable*. Після цього ми можемо звернутися до ресурсу в такий спосіб в коді Java:

R.drawable.ім’я\_файла або в коді Java:

@[ім’я\_пакету:]drawable/ім’я\_файла

При додаванні графічних файлів у проект варто враховувати, що кожен такий файл буде збільшувати розмір проекту. Крім того, великі зображення негативно впливають на продуктивність. Тому слід використовувати невеликі і оптимізовані (стиснуті) графічні файли.

Для роботи із зображеннями в Android можна використовувати різні елементи, але безпосередньо для виведення зображень призначений ImageView:

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

<RelativeLayout xmlns:android="[http://schemas.android.com/apk/res/android"](http://schemas.android.com/apk/res/android) android:id=["@+id](mailto:@+id)/activity\_main"

android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<ImageView android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:src=["@drawable/](mailto:@drawable)flowers" />

</RelativeLayout>

Тут для відображення файлу в *ImageView* встановлюється атрибут *android:src*. В його значенні вказується ім’я графічного ресурсу, яке повинно збігатися з ім’ям файлу без розширення. Якби ми створювали *ImageView* в коді Java із відповідним застосуванням ресурсу, то activity могла б виглядати так:

…

import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

[@Override](mailto:@Override)

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

ImageView imageView=new ImageView(this);

// застосовуємо ресурс

imageView.setImageResource(R.drawable.flowers);

//setContentView(R.layout.activity\_main);

setContentView(imageView);

}

}