**ЛАБОРАТОРНАЯ РАБОТА № 4**

**СТРОКОВЫЕ И ЦВЕТОВЫЕ РЕСУРСЫ**

*Изучение правил создания строковых и цветовых ресурсов проекта, создание приложения с использованием этих ресурсов.*

**Цель работы:** Создать приложение с применением строковых и цветовых ресурсов.

Ресурсы строк - один из важных компонентов приложения. Они используются для работы со всем текстом, содержащимся в окнах приложения, в том числе заголовки окон приложения и локализация на другие языки.

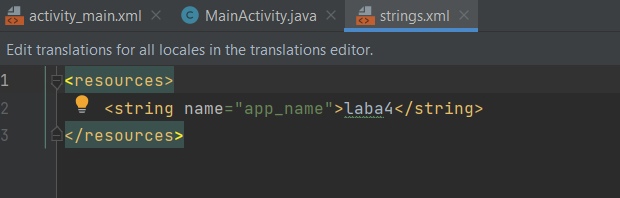


Рис. 4.1 – Ресурсы строк

XML-файлы, представляющие собой ресурсы строк, находятся в проекте в папке res/values. По умолчанию ресурсы строк находятся в файле strings.xml, который может выглядеть следующим образом:

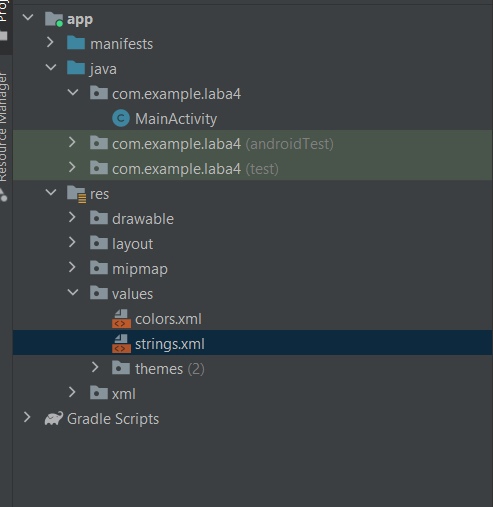


Рис. 4.2 – Расположение ресурсов строк в дереве проекта

В самом простом виде этот файл определяет один ресурс "app\_name", который устанавливает название приложения, которое мы видим в заголовке приложения на экране устройства. Но естественно мы можем определить любые строковые ресурсы. Каждый отдельный ресурс определяется с помощью элемента string, а его атрибут name содержит название ресурса. Затем в приложении в файлах кода мы можем ссылаться на эти ресурсы:

R.string.app\_name

Например, в коде Java:

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

Либо в xml-файле:

@string/app\_name

Например, изменим файл res/values/strings.xml следующим образом:

<resources>  
 <string name="app\_name">ViewApp</string>  
 <string name="message">Hello Android!</string>  
</resources>

Здесь добавлен ресурс message со значением "Hello Android!".

С помощью выражения @string/message передаем атрибуту android:text значение из ресурса.

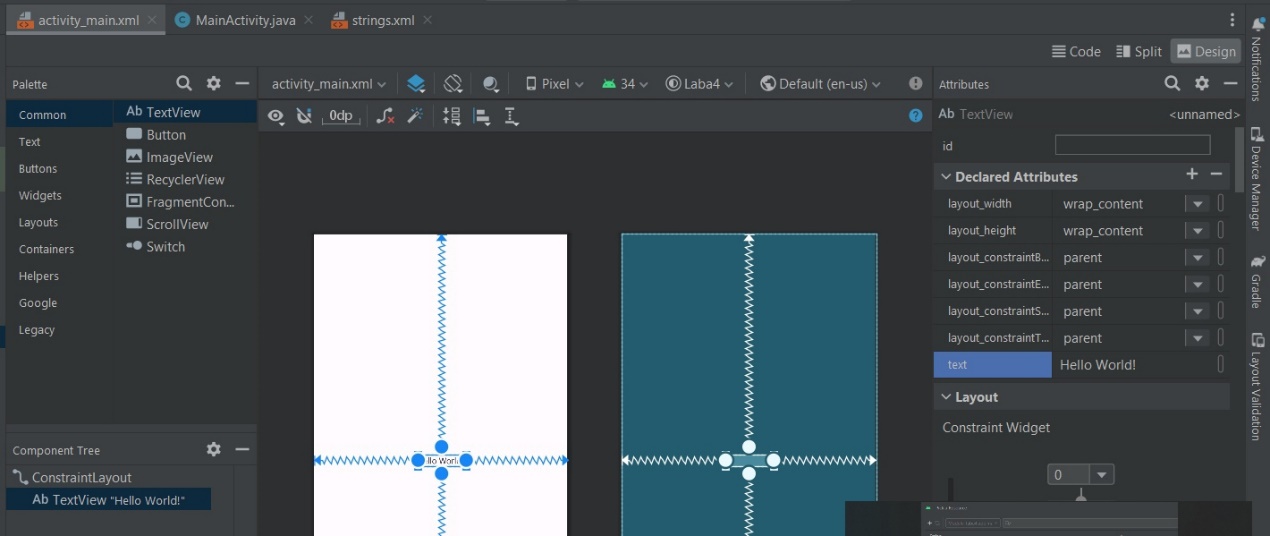


Рис. 4.3 - Передача текста строкового ресурса

Выберем текстовый ресурс:

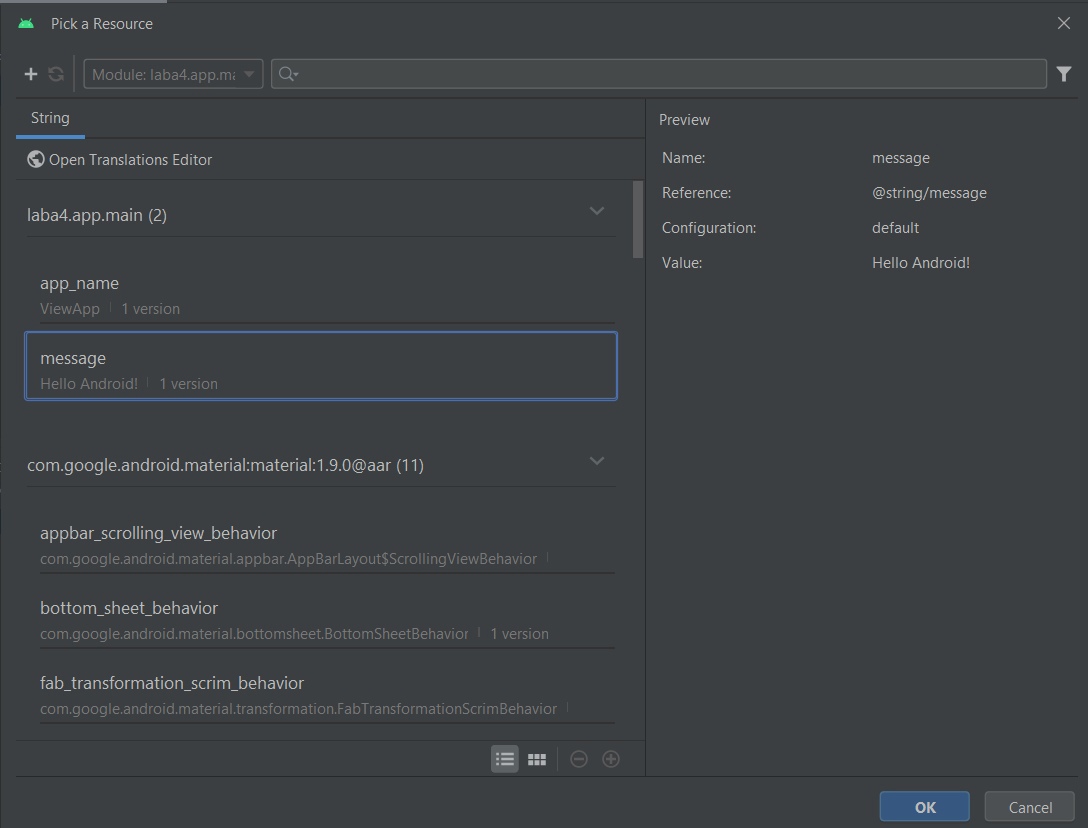


Рис. 4.4 - Передача текста строкового ресурса

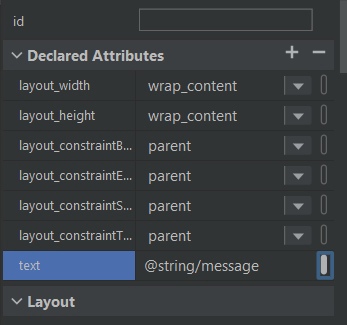


Рис. 4.5 - Передача текста строкового ресурса

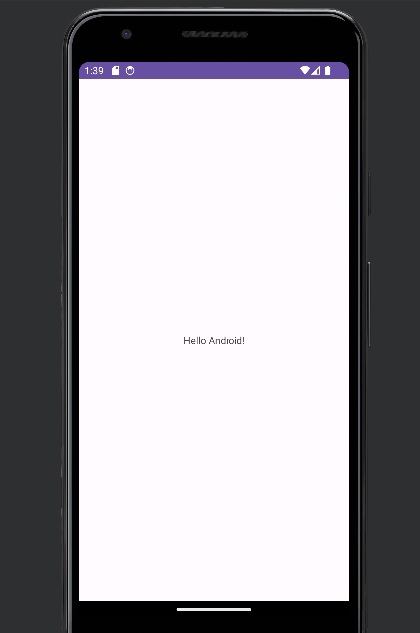


Рис. 4.6 - Файл Activity после изменения строкового ресурса

Аналогично мы могли бы использовать ресурс в коде Activity:

// получаем элемент textView

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

// переустанавливаем у него текст

textView.setText(R.string.message);

Если необходимо получить ресурс в коде Java (необязательно для установки текста в TextView), то в этом случае можно использовать метод getResources().getString(идентификатор\_ресурса)

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

Хотя, по умолчанию, для ресурсов строк применяется файл strings.xml, но разработчики могут добавлять дополнительные файлы ресурсов в каталог проекта res/values. При этом достаточно соблюдать структуру файла: он должен иметь корневой узел и иметь один или несколько элементов .

Так, нажмем на папку res/values правой кнопкой мыши и в появившемся списке выберем пункт New -> Value Resource File:

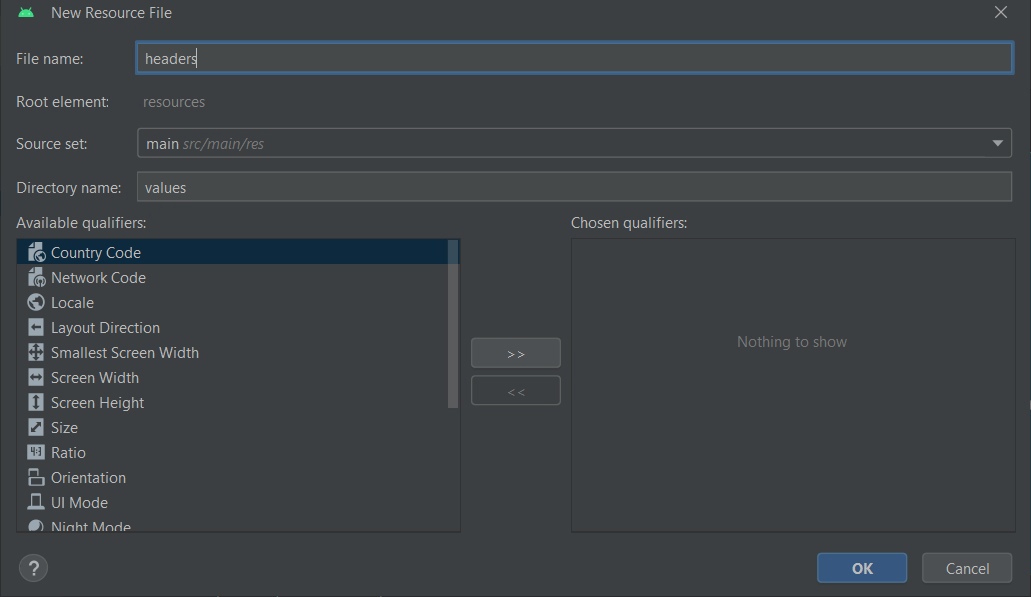


Рис. 4.7 - Создание нового ресурса

Причем следует отметить, что данный тип файлов будет характерен для любого типа ресурсов, который добавляется в папку res/values.

После этого нам будет предложено определить для файла имя:

Назовем файл headers(заголовки для страниц). Для всех остальных полей оставим значения по умолчанию. И в папку res/values будет добавлен новый файл headers.xml. Определим в нем пару ресурсов:

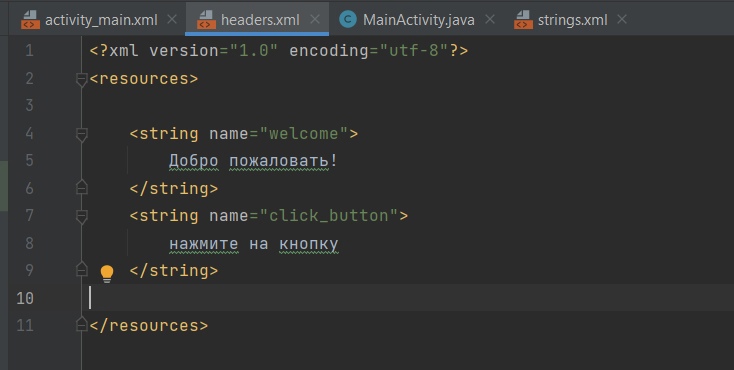


Рис. 4.8 - Заполнение файла ресурсами

После этого мы сможем использовать определенные здесь ресурсы в коде Activity или в файле layout.

В приложении Android также можно определять ресурсы цветов (Color). Они должны храниться в файле по пути res/values и, также, как и ресурсы строк, заключены в тег . Так, по умолчанию при создании самого простого проекта в папку res/values добавляется файл colors.xml:

Цвет определяется с помощью элемента . 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 альфаканала)
* #RRGGBB (#FF00FF - 24-битное значение)
* #AARRGGBB (#80FF00FF - 24-битное значение с добавлением альфа-канала)

Чтобы не трогать и не портить данный файл, определим свой новый файл ресурсов и для этого добавим в папку res/values новый файл ресурсов, который назовем my\_colors.xml.

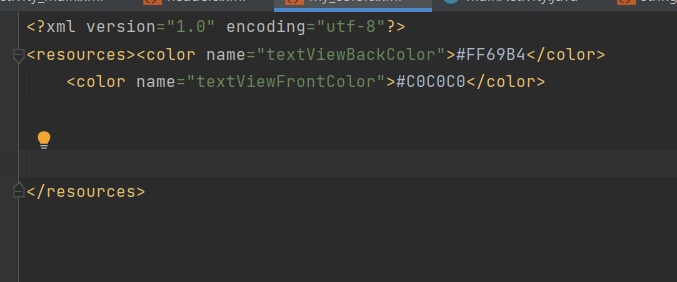


Рис. 4.9 - Заполнение файла ресурсами

С помощью атрибута android:textColor устанавливается цвет текста в TextView, а атрибут android:background устанавливает фон TextView. В качестве значения они используют цвет, например, в том же шестнадцатеричном формате. Для получения самого цвета после "@color/" указывается имя ресурса.

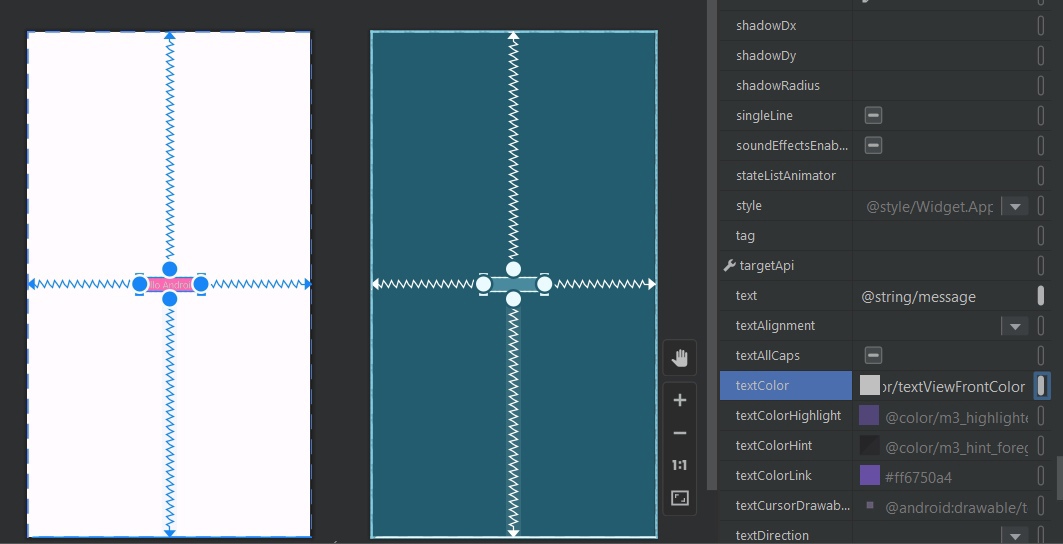


Рис. 4.10 - Выбор цвета

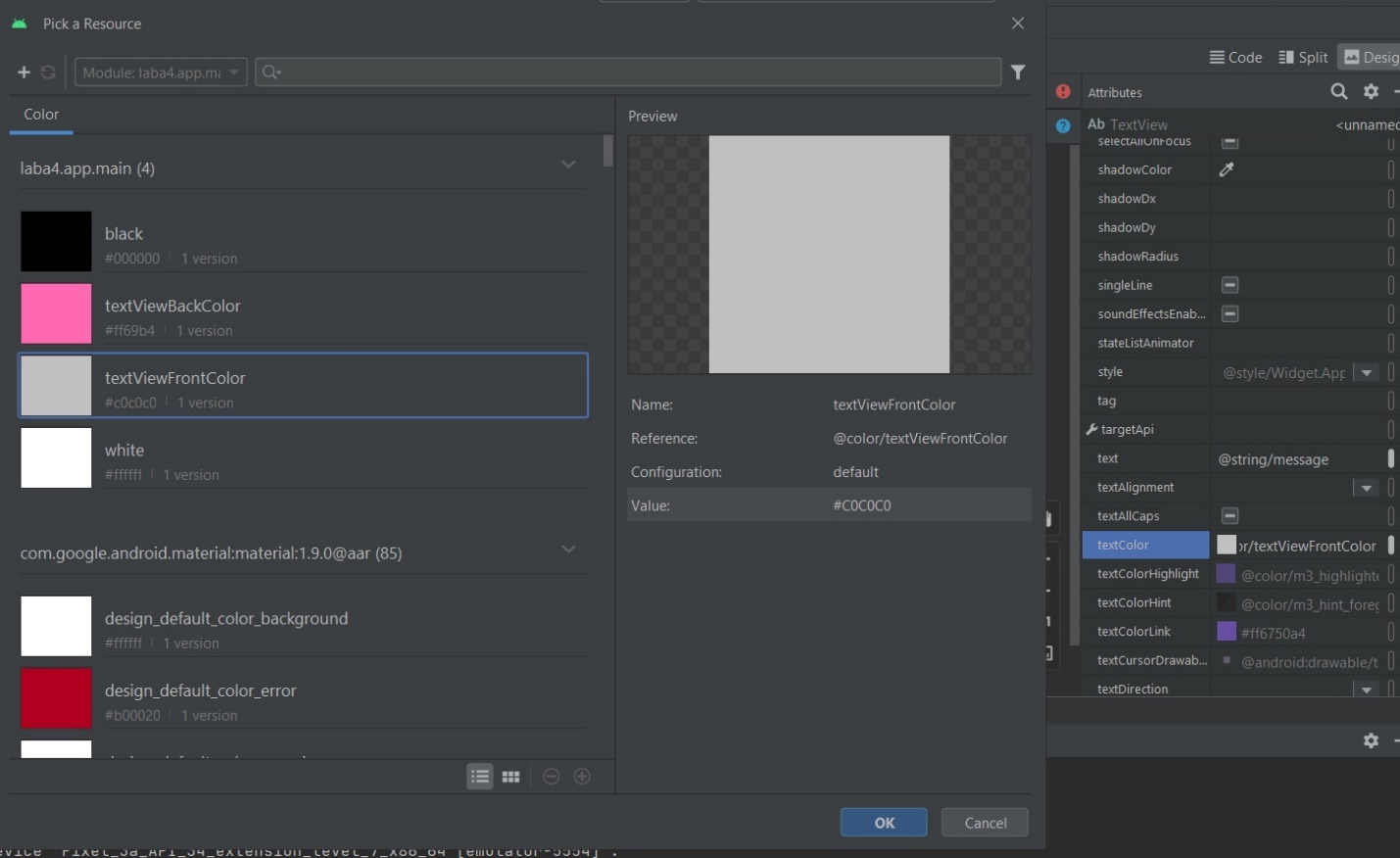


Рис. 4.11 - Выбор цвета

Все цвета из папки res проекта подгрузятся автоматически и будут доступны в окне выбора цвета.

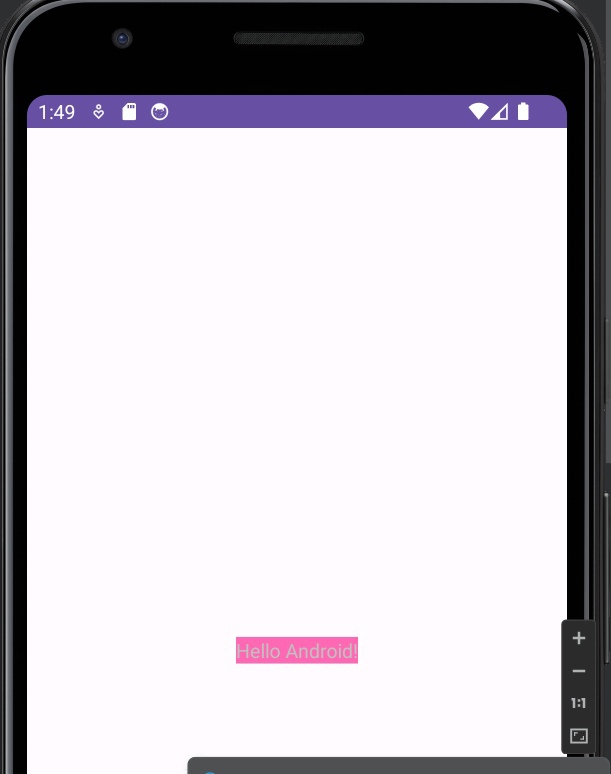


Рис. 4.12 - Результат выбора цвета

Также можно использовать цветовые ресурсы в коде MainActivity:

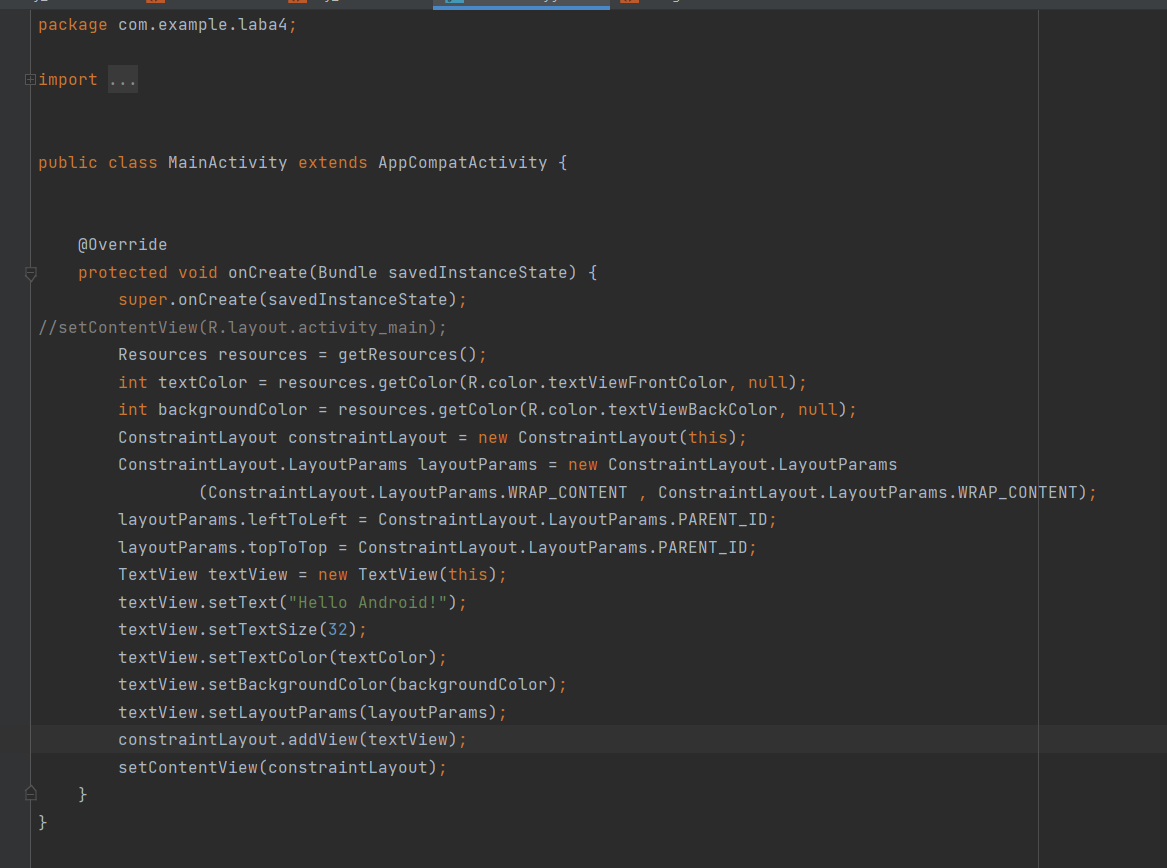


Рис. 4.13 - Выбор цветовых ресурсов в MainActivity.java

Для получения цвета из ресурсов применяется метод resources.getColor(), который принимает два параметра. Первый параметр - идентификатор ресурса, цвет которого надо получить. Второй параметр представляет тему. Но поскольку в данном случае тема не важна, для этого параметра передаем значение null.

Следует учитывать, что метод resources.getColor() с двумя параметрами, который использован выше, доступен, если для минимальная версия Android не ниже Android 6.0 (или Android 23).

Итоговый файл activity\_main.xml:

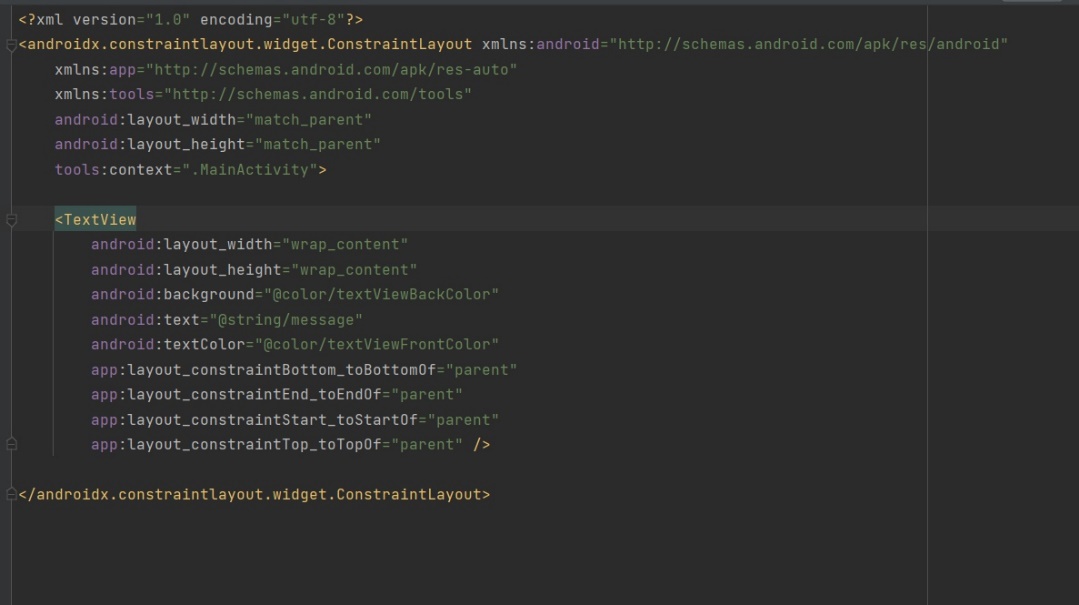


Рис. 4.14 - Файл activity\_main.xml

**Вывод**: в результате выполнения лабораторной работы было создано приложение с применением строковых и цветовых ресурсов