Лабораторная работа 3.

Работа с элементами экрана из кода. Обработка событий. Использование ресурсов приложения и всплывающих сообщений.

Чтобы обратиться к элементу экрана из кода, нам нужен его **ID**. Он прописывается либо в **Properties**, либо в **layout-файлах**, как вам удобнее.

Для ID существует четкий формат - **@+id/name**, где **+** означает, что это новый ресурс и он должен добавиться в **R.java** класс, если он там еще не существует.

Чтобы к нему обратиться, надо написать **R.id.name**, т.к. все элементы прописываются в классе **id** подкласса **R.**

Для того, чтобы обратиться к элементу программно, нам понадобится метод **findViewById**. Он по ID возвращает View.

Папка res/values. Используем ресурсы приложения.

В подпапках res хранятся различные **ресурсы** приложения. Мы уже отлично знаем про **layout-файлы** в папке **res/layout**. В папке **res/drawable** с density-суффиксами хранятся **картинки**. Теперь обратим внимание на папку **res/values**. Она предназначена для хранения ресурсов (констант) различных типов. Мы рассмотрим типы **String** и **Color**.

Например, в проекте, созданном по умолчанию, мы видим два элемента типа String:

hello – по умолчанию он использован в свойстве Text в TextView в main.xml. И соответственно TextView отображает значение этого элемента.

app_name – по умолчанию используется как заголовок для приложения и Activity. Это указывается в манифест-файле.

На эти элементы можно кликнуть и увидеть справа, что они собой представляют: **имя** (Name) и **значение** (Value).

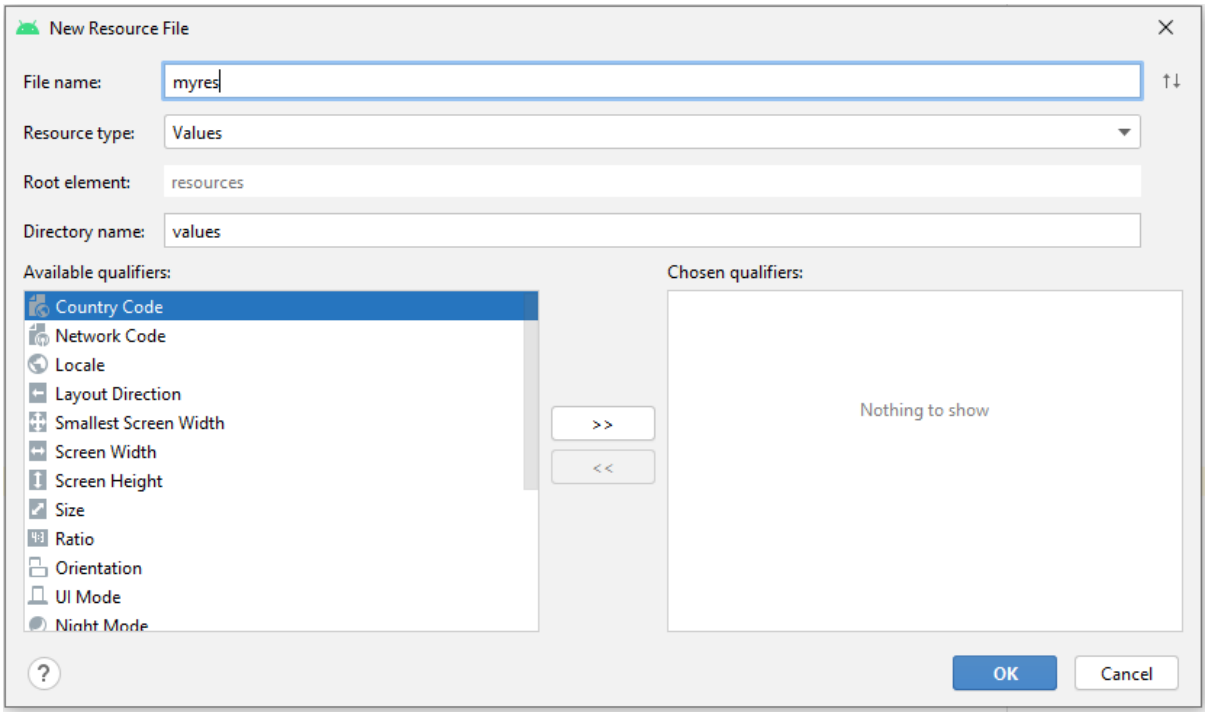
Name – это **ID**. Оно должно быть уникальным, и для него в **R.java** создается константа, чтобы мы могли иметь доступ к этому String-элементу.

Если мы посмотрим XML-содержимое файла strings.xml (вкладка снизу – аналогично как для main.xml), то видим, что там все прозрачно и просто. Попробуем и мы использовать ресурсы.

Например:

Экран разделен на две одинаковые половины, содержащие LinearLayout, Button и TextView. Для **LinearLayout** мы укажем **фоновый цвет**, а для **TextView** и **Button** – изменим **текст**. Реализуем это с помощью ресурсов. Причем View-элементы **верхней** части мы будем настраивать вручную через **properties**, а **нижнюю** часть попробуем настроить **программно**.

Создадим свой файл с ресурсами в папке values (правой кнопкой по папке res), название пусть будет myres.



Далее добавим элементы в файл: пишем **имя** и **значение**. Создадим 4 String-элемента и 2 Color-элемента:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="tvTopText">Верхний текст</string>
    <string name="btnTopText">Верхняя кнопка</string>
    <string name="tvBottomText">Нижний текст</string>
    <string name="btnBottomText">Нижняя кнопка</string>
    <color name="llTopColor">#336699</color>
    <color name="llBottomColor">#339966</color>
</resources>
```

Не забудьте сохранить.

Теперь настроим View-элементы на их использование. Сначала верхние:

llTop – в Properties находим свойство **Background**, жмем кнопку выбора (три точки), в ветке **Color** выделяем **llTopColor** и жмем **OK**

tvTop – для свойства Text откройте окно выбора и найдите там **tvTopText**.
btnTop - для свойства Text откройте окно выбора и найдите там **btnTopText**.

Цвет верхней части изменился и тексты поменялись на те, что мы указывали в myres.xml.

Чтобы изменить нижнюю часть, будем «кодить». Сначала находим элементы, потом присваиваем им значения.

```
public class MainActivity extends Activity {
  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    LinearLayout llBottom = (LinearLayout)findViewById(R.id.llBottom);
    TextView tvBottom = (TextView)findViewById(R.id.tvBottom);
    Button btnBottom = (Button)findViewById(R.id.btnBottom);

    llBottom.setBackgroundResource(R.color.llBottomColor);
    tvBottom.setText(R.string.tvBottomText);
    btnBottom.setText(R.string.btnBottomText);

  }
}
```

Обратите внимание на то, что для смены текста используется метод **setText**. Только это не тот же setText, что мы использовали, когда задавали текст напрямую. Этот метод **на вход** принимает **ID** и мы используем R.java, который хранит ID всех наших ресурсов. Т.е. методы называются одинаково, но на вход принимают разные параметры.

Сохраняем, запускаем и проверяем. Теперь тексты и цвета взяты из файла ресурсов. Вы можете изменить содержимое myres.xml (например текст для верхней кнопки), сохранить, запустить приложение и увидите изменения.

Иногда необходимо в коде получить не ID ресурса, а его значение. Это делается следующим образом:

```
getResources().getString(R.string.tvBottomText);
```

Выражение вернет текст «Нижний текст», соответствующий String-ресурсу с name = **tvBottomText**.

- Напоследок пару слов об организации файлов для хранения ресурсов: Мы сейчас создали String и Color ресурсы **в одном файле** myres.xml, но рекомендуется их разделять **по разным файлам** (например strings.xml, colors.xml ...).
- Имена ресурсов сквозные для всех файлов в папке res/values. Т.е. вы не можете в разных файлах создать ресурс с одним именем и типом.
- Имена файлов ресурсов могут быть произвольными и файлов можно создавать сколько угодно. В R.java попадут все ресурсы из этих файлов.

Для лабораторной работы создайте новый модуль.

Задания (по вариантам).

Используйте Activity, как обработчик, для создания событий. На нажатия должны появляться всплывающие сообщения. Используемые в сообщениях и названиях строки и цвета прописать в ресурсах.

1. Добавьте на экран кнопки (красный, синий, зеленый, желтый, белый), по нажатию на которые будет меняться фоновый цвет экрана.
2. Программа загадывает число от 1 до 3, а пользователь пытается его угадать, нажимая на нужную кнопку с цифрой.
3. Добавьте на экран три кнопки и текстовое поле. По нажатию на каждую из кнопок текст, написанный на копке, попадает в текстовое поле. Добавьте кнопку для очистки текстового поля.
4. Добавьте на экран текстовое поле и две кнопки. По нажатию на первую кнопку вторая кнопка меняет свое положение на экране, а в текстовом поле отображается количество нажатий. Нажатие на вторую кнопку сбрасывает подсчет и возвращает ее на место.
5. Добавьте текстовое поле и две кнопки «Next» и «Back», по нажатию на которые, текстовое поле циклично выводит сообщения.
6. Добавьте на экран две кнопки. Нажатие на первую кнопку меняет цвет второй кнопки на случайный, нажатие на вторую, делает тоже самое с первой кнопкой.
7. Добавьте на экран три кнопки. Нажатие на одну из них меняет тексты на двух других кнопках между собой.
8. Добавьте на экран кнопки (красный, синий, зеленый) и текстовое поле, по нажатию на кнопки будет меняться цвет текста.
9. Программа загадывает цвет, а пользователь пытается его угадать, нажимая на кнопку нужного цвета. Достаточно четырех цветов.
10. Сделайте «клавиатуру». Добавьте текстовое поле и несколько кнопок с буквами и пробелом. В текстовом поле будет отображаться набранный текст. (функция `append()`)

Рекомендации:

Для изменения цвета используйте функции:

```
setTextColor(Color.BLUE)
setBackgroundColor(Color.RED)
```

Для смены цвета экрана нужно сначала найти Layout по id.

Обратите внимание, переменную для Layout объявлять глобально нельзя. Придется его поднимать в каждой функции отдельно.