МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра автоматизированных систем управления

Лабораторная работа №4

по дисциплине: «Современные технологии разработки программного обеспечения»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнил:** |  | **Проверил:** |  | |
| Студент | *Чибисов И.А* | Преподаватель | *Эстрайх И.В.*  *ст. преподаватель* | |
| Факультет | *АВТФ* |  |  | |
| Направление (специальность) подготовки | *09.04.03 – Прикладная информатика* | Балл: \_\_\_\_\_\_\_\_\_ | |  |
| Группа | *АПМ-22* | Оценка *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* | | |
| Шифр | *010594214* |  |  | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| подпись | | подпись | | |
| Дата сдачи: | «\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. | Дата защиты: | «\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г | |

Новосибирск

2023

Цель работы: научиться создавать приложения, учитывающие ориентацию устройства, освоить способы улучшения дизайна приложения с помощью тем и стилей.

**Задание на лабораторную работу**

1. Создайте приложение с кнопками и текстовым полем, учитывающее ориентацию устройства.

2. Создайте приложение или несколько приложений, использующих различные темы и стили.

3. Включите в отчет цель работы, коды программ, скриншоты с результатами и вывод.

**Выполнение:**

1. Ориентация устройства

В Android studio создадим новый проект на языке Java, назовем его «chooseYouOrientationJavaIVAN». Добавим на поле 6 кнопок (рисунок 1), после чего скомпонуем кнопки, чтобы в горизонтальном виде они не оказывались за пределами видимости.



Рисунок 1 – кнопки на поле

Для этого создадим новую подпапку в res с помощью мастера (рисунок 2) и внутри этой папки создадим также xml-файл (рисунок 3).

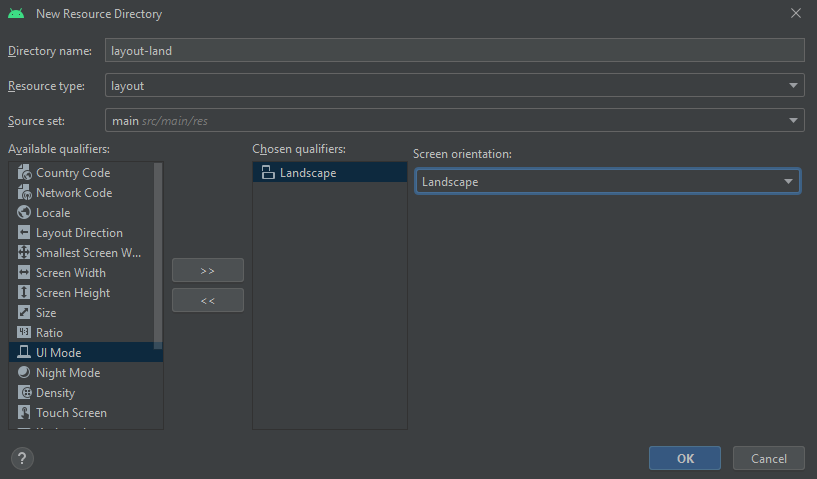


Рисунок 2 – создание подпапки

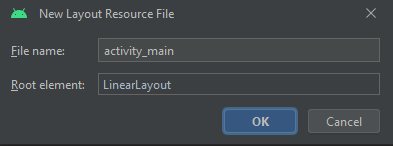


Рисунок 3 – создание xml-файла

В созданный файл добавим код, приведенный в методичке, после чего запустим программу и проверим работу приложения при повороте устройства на 90 градусов (рисунок 4).



Рисунок 4 – приложение при повороте устройства

Для проверки текущей ориентации устройства с помощью кода добавим функцию, представленную в методичке, после чего на поля портретной и альбомной ориентации добавим по одной дополнительной кнопке «Проверка», по нажатию которых в консоль будет выводиться результат, показывающий текущую ориентацию устройства (рисунки 5, 6).

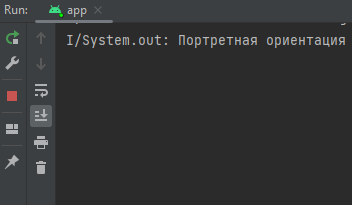


Рисунок 5 – нажатие кнопки в портретной ориентации и результат в консоли

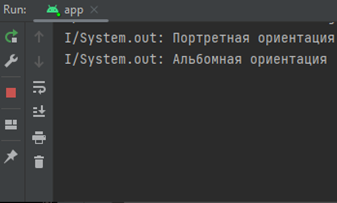


Рисунок 6 – нажатие кнопки в альбомной ориентации и результат в консоли

Код метода представлен далее:

public void onClick(View view) {  
 if (getResources().getConfiguration().orientation == Configuration.*ORIENTATION\_PORTRAIT*)  
// Вывод результата в консоль будем производить с помощью System.out.println  
 System.*out*.println ("Портретная ориентация");  
 else if (getResources().getConfiguration().orientation == Configuration.*ORIENTATION\_LANDSCAPE*)  
 System.*out*.println ("Альбомная ориентация");  
 else  
 System.*out*.println ("");  
 }

Можно установить нужную ориентацию через манифест приложения в элементе <activity> параметр «android:screenOrientation:». Таким образом можно задать принудительную альбомную или портретную ориентацию, либо же задать вариант, когда устройство полагается на сенсор.

Доработаем программу, чтобы при нажатии на кнопку «Проверка» текущая ориентация устройства отображалась в текстовом поле, а не в консоли. Для этого добавим на экран элемент TextView и пропишем для него все необходимое в файле MainActivity.

Код метода onClick представлен далее:

public void onClick(View view) {

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

if (getResources().getConfiguration().orientation == Configuration.ORIENTATION\_PORTRAIT) {

//Результат выводится на элемент textView

textView.setText("Портретная ориентация");

}

else if (getResources().getConfiguration().orientation == Configuration.ORIENTATION\_LANDSCAPE) {

textView.setText("Альбомная ориентация");

}

else System.out.println ("");

}

Запустим приложение и посмотрим на результаты (рисунки 7,8).

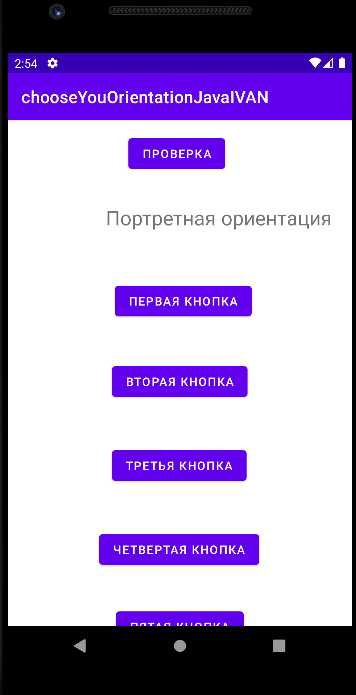


Рисунок 7 – устройство при портретной ориентации

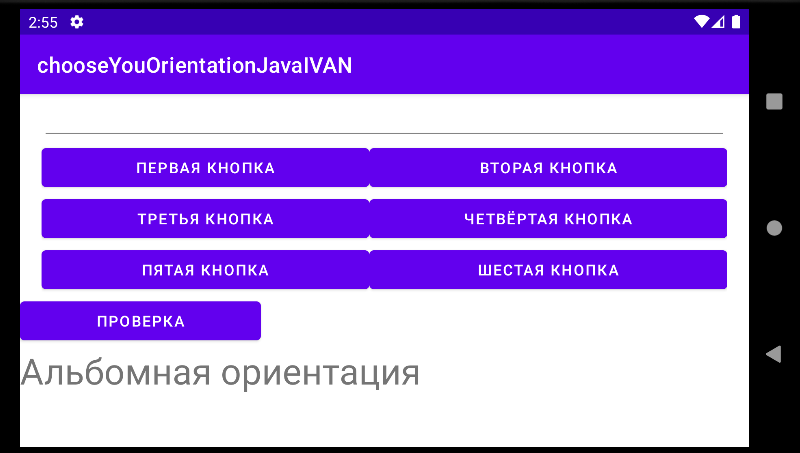


Рисунок 8 – устройство при альбомной ориентации

Как видим, приложение правильно отображает ориентацию устройства.

1. Темы и стили приложений

Откроем приложение, разработанное в предыдущей лабораторной работе, внутри которого сделаем окно «О программе» диалоговым. Для этого необходимо в манифесте для второй активности дописать строчку с указанием на тему:

android:theme="@style/Theme.AppCompat.Light.Dialog".

Запустим приложение и сравним результаты до и после добавления темы (рисунок 9).

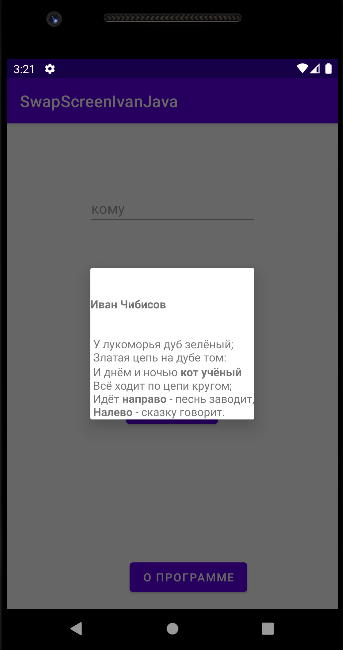
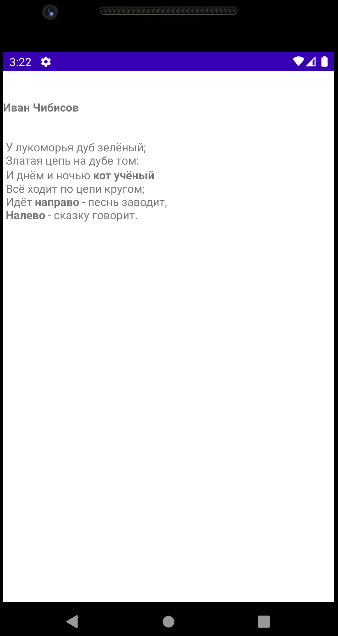


Рисунок 9 – сравнение окна до и после установления темы

После добавлении темы вместо отдельного окна открывается диалоговое окно поверх рабочего пространства, чего мы и добивались.

Поработаем со стилями, создадим новый проект на языке Java, назовем его «StylesJavaIvan» и заменим код для TextView на код из лабораторной, после чего в папке «res/values» создадим новый xml-файл styles.xml. В него вставим код из лабораторной, в котором указаны цвет и размер шрифта. В файле разметки TextView присвоим ему созданный ранее стиль, прописав «style="@style/MyTextStyle"» и получим следующее (рисунок 10).

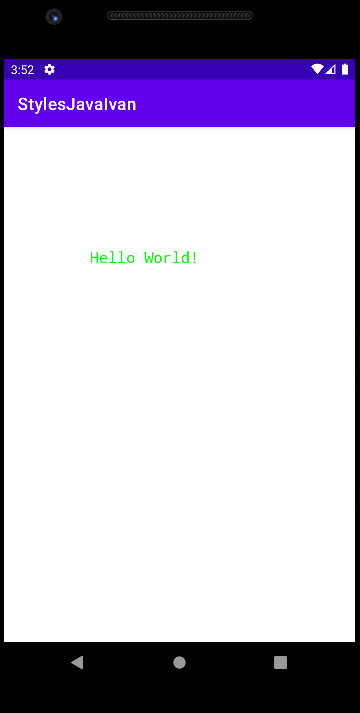


Рисунок 10 – TextView со стилем

Можем воспользоваться функцией наследования, чтобы использовать уже готовые стили в качестве наработок. Механизм работает с помощью атрибута «parent», указав существующий стиль, а затем переопределить или добавить некоторые свойства. При наследовании от собственных стилей атрибут parent не нужен, можно просто использовать префикс имени наследуемого стиля перед создаваемым новым стилем, разделяя имена стилей точкой. Создадим таким образом стиль с красным цветом текста, для этого напишем следующее:

<style name="MyTextStyle.Red">

<item name="android:textColor">#FF0000</item>

</style>

После чего изменим в файле activity\_main свойство style на «MyTextStyle.Red» и запустим приложение (рисунок 11).



Рисунок 11 – приложение при смене стиля

Теперь изменим стандартную тему. Изменять и добавлять темы нужно в папке «res/values/themes». Изменим тему, поменяв в ней цвета, после чего получим следующее (рисунок 12).

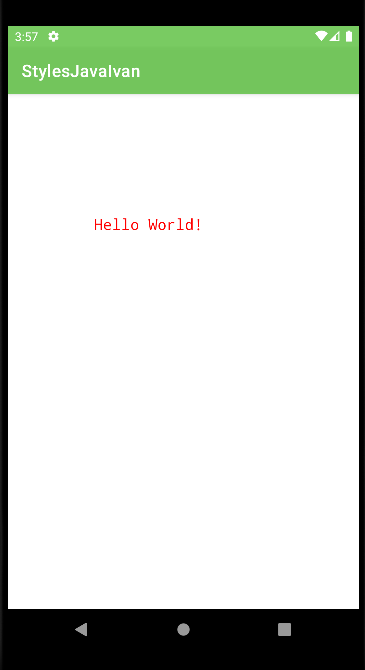


Рисунок 12 – измененная тема приложения

Вывод: Мы создали приложение с кнопками и текстовым полем, которое по нажатию кнопки пишет текущую ориентацию устройства. Затем доработали приложение, для вывода второй активности в качестве диалогового окна, а также создали приложение, в котором использовали разные стили и темы.