

Практика 9: Разработка калькулятора

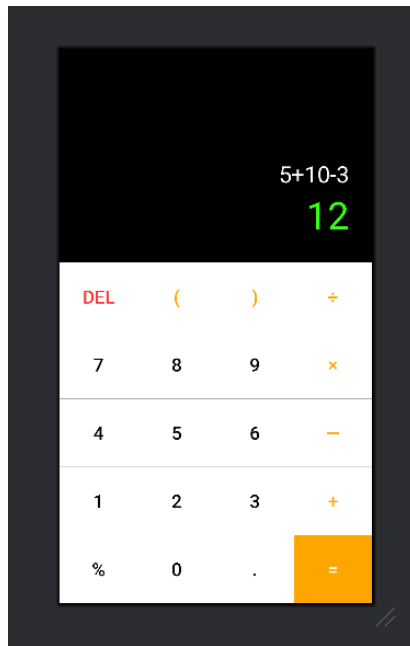
Согласно инструкции, добавим цвета и стиль для кнопки

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="red">#FF3131</color>
    <color name="pink">#FFC0CB</color>
    <color name="orange">#FFA500</color>
    <color name="neon_green">#39FF14</color>
    <style name="Button_Style" parent="Widget.AppCompat.Button.Colored">
        <item name="android:background">@color/white</item>
        <item name="android:textSize">24sp</item>
        <item name="android:textColor">@color/black</item>
        <item name="android:gravity">center</item>
    </style>
</resources>
```

Вставляем длинный код макета

```
<include layout="@layout/content_main" />
<include layout="@layout/content_main2" />
<include layout="@layout/content_main3" />
<include layout="@layout/content_main4" />
<include layout="@layout/content_main5" />
<include layout="@layout/content_main6" />
<include layout="@layout/content_main7" />
<include layout="@layout/content_main8" />
<include layout="@layout/content_main9" />
<include layout="@layout/content_main10" />
<include layout="@layout/content_main11" />
<include layout="@layout/content_main12" />
<include layout="@layout/content_main13" />
<include layout="@layout/content_main14" />
<include layout="@layout/content_main15" />
<include layout="@layout/content_main16" />
<include layout="@layout/content_main17" />
<include layout="@layout/content_main18" />
<include layout="@layout/content_main19" />
<include layout="@layout/content_main20" />
<include layout="@layout/content_main21" />
<include layout="@layout/content_main22" />
<include layout="@layout/content_main23" />
<include layout="@layout/content_main24" />
<include layout="@layout/content_main25" />
<include layout="@layout/content_main26" />
<include layout="@layout/content_main27" />
<include layout="@layout/content_main28" />
<include layout="@layout/content_main29" />
<include layout="@layout/content_main30" />
<include layout="@layout/content_main31" />
<include layout="@layout/content_main32" />
<include layout="@layout/content_main33" />
<include layout="@layout/content_main34" />
<include layout="@layout/content_main35" />
<include layout="@layout/content_main36" />
<include layout="@layout/content_main37" />
<include layout="@layout/content_main38" />
<include layout="@layout/content_main39" />
<include layout="@layout/content_main40" />
<include layout="@layout/content_main41" />
<include layout="@layout/content_main42" />
<include layout="@layout/content_main43" />
<include layout="@layout/content_main44" />
<include layout="@layout/content_main45" />
<include layout="@layout/content_main46" />
<include layout="@layout/content_main47" />
<include layout="@layout/content_main48" />
<include layout="@layout/content_main49" />
<include layout="@layout/content_main50" />
<include layout="@layout/content_main51" />
<include layout="@layout/content_main52" />
<include layout="@layout/content_main53" />
<include layout="@layout/content_main54" />
<include layout="@layout/content_main55" />
<include layout="@layout/content_main56" />
<include layout="@layout/content_main57" />
<include layout="@layout/content_main58" />
<include layout="@layout/content_main59" />
<include layout="@layout/content_main60" />
<include layout="@layout/content_main61" />
<include layout="@layout/content_main62" />
<include layout="@layout/content_main63" />
<include layout="@layout/content_main64" />
<include layout="@layout/content_main65" />
<include layout="@layout/content_main66" />
<include layout="@layout/content_main67" />
<include layout="@layout/content_main68" />
<include layout="@layout/content_main69" />
<include layout="@layout/content_main70" />
<include layout="@layout/content_main71" />
<include layout="@layout/content_main72" />
<include layout="@layout/content_main73" />
<include layout="@layout/content_main74" />
<include layout="@layout/content_main75" />
<include layout="@layout/content_main76" />
<include layout="@layout/content_main77" />
<include layout="@layout/content_main78" />
<include layout="@layout/content_main79" />
<include layout="@layout/content_main80" />
<include layout="@layout/content_main81" />
<include layout="@layout/content_main82" />
<include layout="@layout/content_main83" />
<include layout="@layout/content_main84" />
<include layout="@layout/content_main85" />
<include layout="@layout/content_main86" />
<include layout="@layout/content_main87" />
<include layout="@layout/content_main88" />
<include layout="@layout/content_main89" />
<include layout="@layout/content_main90" />
<include layout="@layout/content_main91" />
<include layout="@layout/content_main92" />
<include layout="@layout/content_main93" />
<include layout="@layout/content_main94" />
<include layout="@layout/content_main95" />
<include layout="@layout/content_main96" />
<include layout="@layout/content_main97" />
<include layout="@layout/content_main98" />
<include layout="@layout/content_main99" />
<include layout="@layout/content_main100" />
```

И вот, что получается



Для дальнейшей работы подключаем ViewBinding

```
buildFeatures{
    viewBinding = true
}
```

Привязываем действие каждой кнопки

```
binding.buttonClear.setOnClickListener{
    binding.input.text = " "
    binding.output.text = " "
}
binding.buttonBracketLeft.setOnClickListener {
    addToInputText("(")
}
binding.buttonBracketLeft.setOnClickListener {
    addToInputText(")")
}
```

И остальные по аналогии

Подключаем exp4j через dependencies

```
implementation ("net.objecthunter:exp4j:0.4.8")
```

Добавим функции для ввода символов и получения всего выражения

```
// Функция для добавления текста к полю ввода
private fun addToInputText(value: String) {
    binding.input.append(value)// Добавляем переданное значение в конец поля ввода
}
// Функция для получения строки ввода
private fun getInputExpression(): String {
    return binding.input.text.toString()// Возвращаем текст из поля ввода
}
```

И для знака равно вызывается функция showResult, которая вычисляет результат

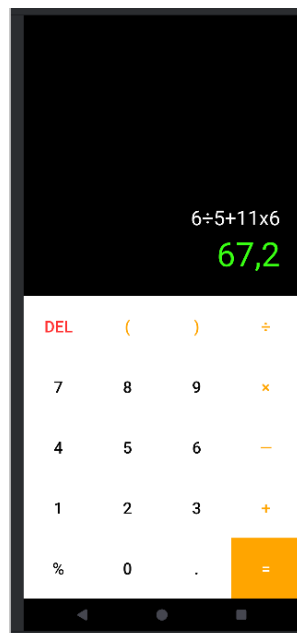
```
// Функция для показа результата вычислений
private fun showResult() {
    try {
        val expression = getInputExpression()
            .replace("%", "/100") // Замена %
            .replace("x", "*") // Замена x
            .replace("÷", "/") // Замена ÷
        val result = ExpressionBuilder(expression).build().evaluate()
        binding.output.text = DecimalFormat("0.#####").format(result).toString()
        binding.output.setTextColor(
            ContextCompat.getColor(
                this,
                R.color.neon_green
            )
        )
    } catch (e: Exception) {
        // Обработка исключений
    }
}
```

```

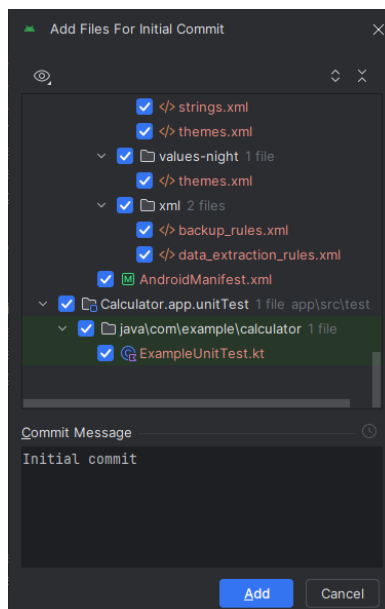
    )
    )
} catch (e: Exception) {
    binding.output.text = "Ошибка"
    binding.output.setTextColor(
        ContextCompat.getColor(
            this,
            R.color.red
        )
    )
}
}
}

```

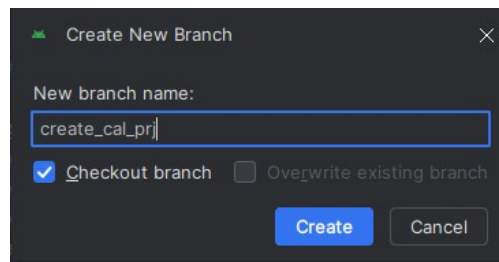
Тестируем



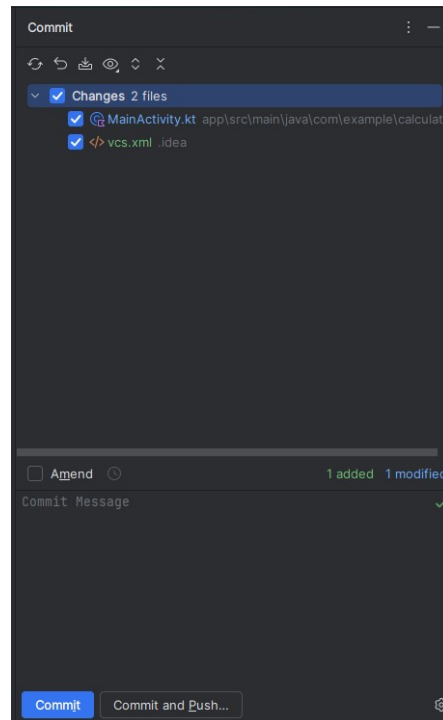
Выкладываем проект на GitHub



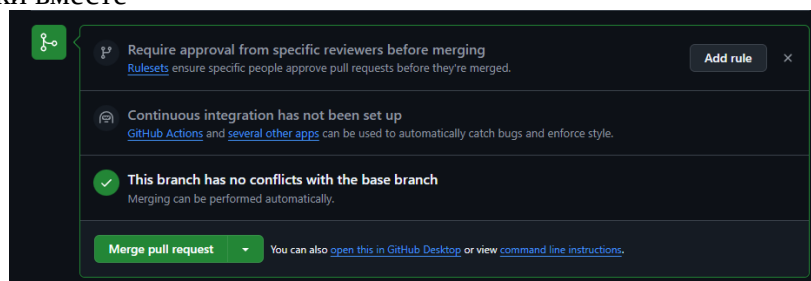
Создаем новую ветку



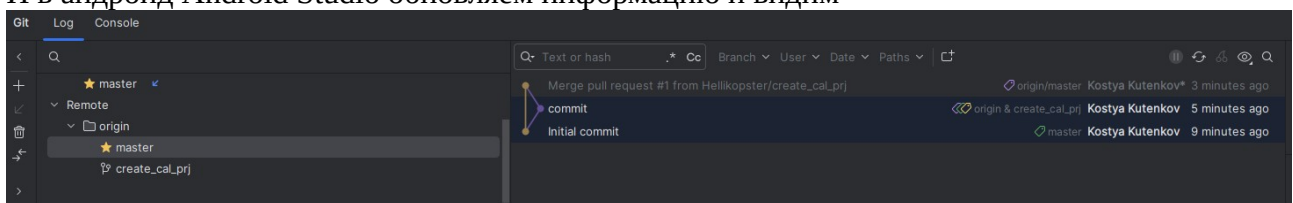
Коммитим изменения и пушим



Сливаем две ветки вместе



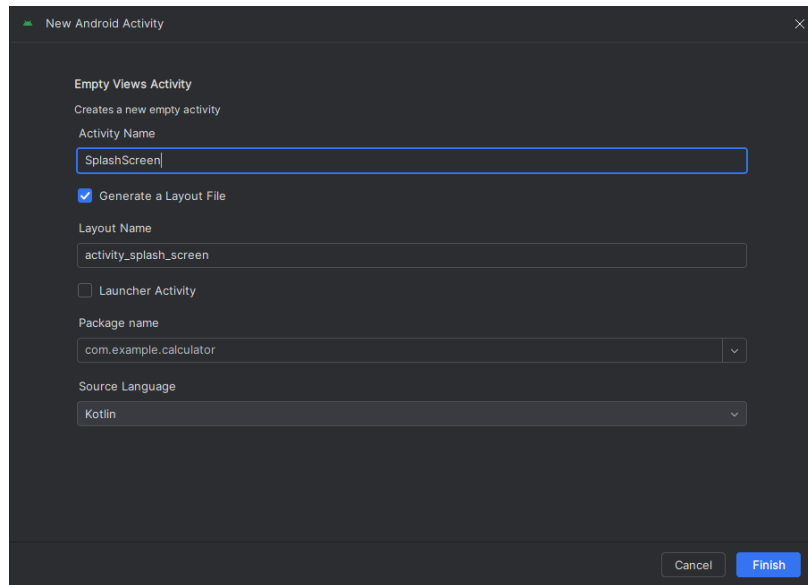
И в андроид Android Studio обновляем информацию и видим



Ссылка на [GitHub](#)

Заставка приложения

Добавим новую активность



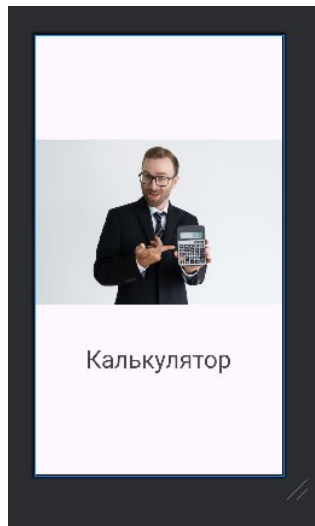
Оформляем

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SplashScreen">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="400dp"
        android:layout_marginTop="112dp"
        android:src="@drawable/splash_screen"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.495"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/title"
        android:textSize="40sp"
        app:layout_constraintTop_toBottomOf="@+id/imageView"
        tools:layout_editor_absoluteX="0dp" />

</androidx.constraintlayout.widget.ConstraintLayout>
```



В функции onCreate класса SplashScreen следующий код

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContentView(R.layout.activity_splash_screen)  
    android.os.Handler(Looper.getMainLooper()).postDelayed({  
        val intent = Intent(this, MainActivity::class.java)  
        startActivity(intent)  
    }, 2000)  
}
```

Немного подправляем manifest

```
<activity  
    android:name=".MainActivity"  
    android:exported="false" />  
<activity  
    android:name=".SplashScreen"  
    android:exported="true">  
    <intent-filter>  
        <action android:name="android.intent.action.MAIN" />  
  
        <category android:name="android.intent.category.LAUNCHER" />  
    </intent-filter>  
</activity>
```

И теперь каждый появляется новая заставка

