

Используйте отладчик в Android Studio

Прежде чем начать

В этом руководстве вы узнаете, как использовать отладчик в Android Studio для проверки того, что происходит в приложении **Dice Roller** во время выполнения.

Отладчик - это важный инструмент, позволяющий контролировать выполнение кода, на котором основано ваше приложение для Android, чтобы вы могли исправить любые ошибки в нем. Он позволяет указывать точки, в которых следует приостановить выполнение кода, и вручную взаимодействовать с переменными, методами и другими аспектами кода.

Необходимые условия

- Базовое знакомство с Android Studio
- Возможность создать и запустить базовое приложение Jetpack Compose в Android Studio
- Завершение коделаба «Создание интерактивного приложения Dice Roller App».

Что вы узнаете

- Как подключить отладчик к приложению Android.
- Как запустить приложение с подключенным отладчиком.
- Как использовать некоторые основные функции отладчика.
- Для чего обычно используется отладчик.

Что вам понадобится

- Компьютер с установленной Android Studio
- Код решения для приложения Dice Roller в Compose

Запуск отладчика

Есть два способа запустить отладчик вместе с вашим приложением:

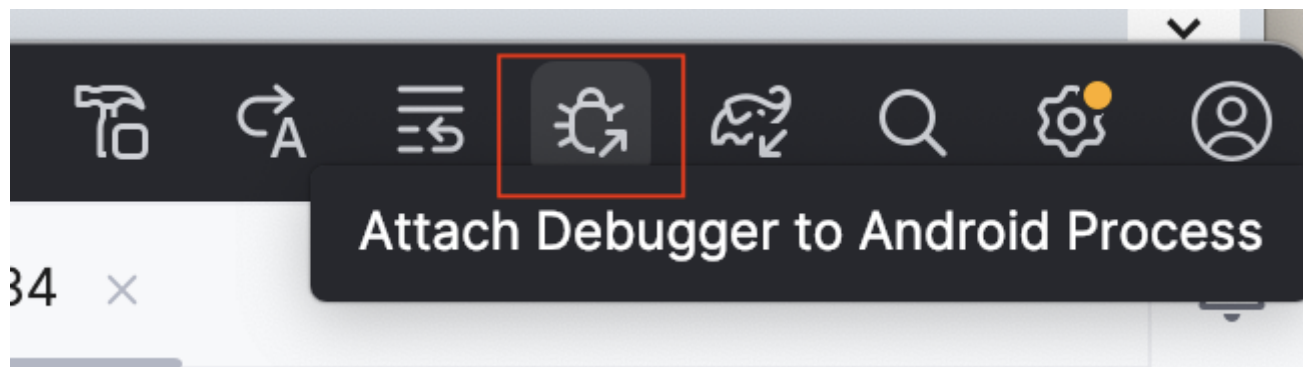
- Прикрепить отладчик к существующему процессу приложения, запущенному на устройстве или эмуляторе.
- Запустить приложение с отладчиком.

Оба способа в определенной степени выполняют одну и ту же задачу. Когда вы ознакомитесь с обоими способами, вы сможете выбрать тот, который вам больше нравится, или тот, который необходим.

Прикрепить отладчик к процессу приложения Если ваше приложение уже запущено, вы можете прикрепить отладчик к нему.

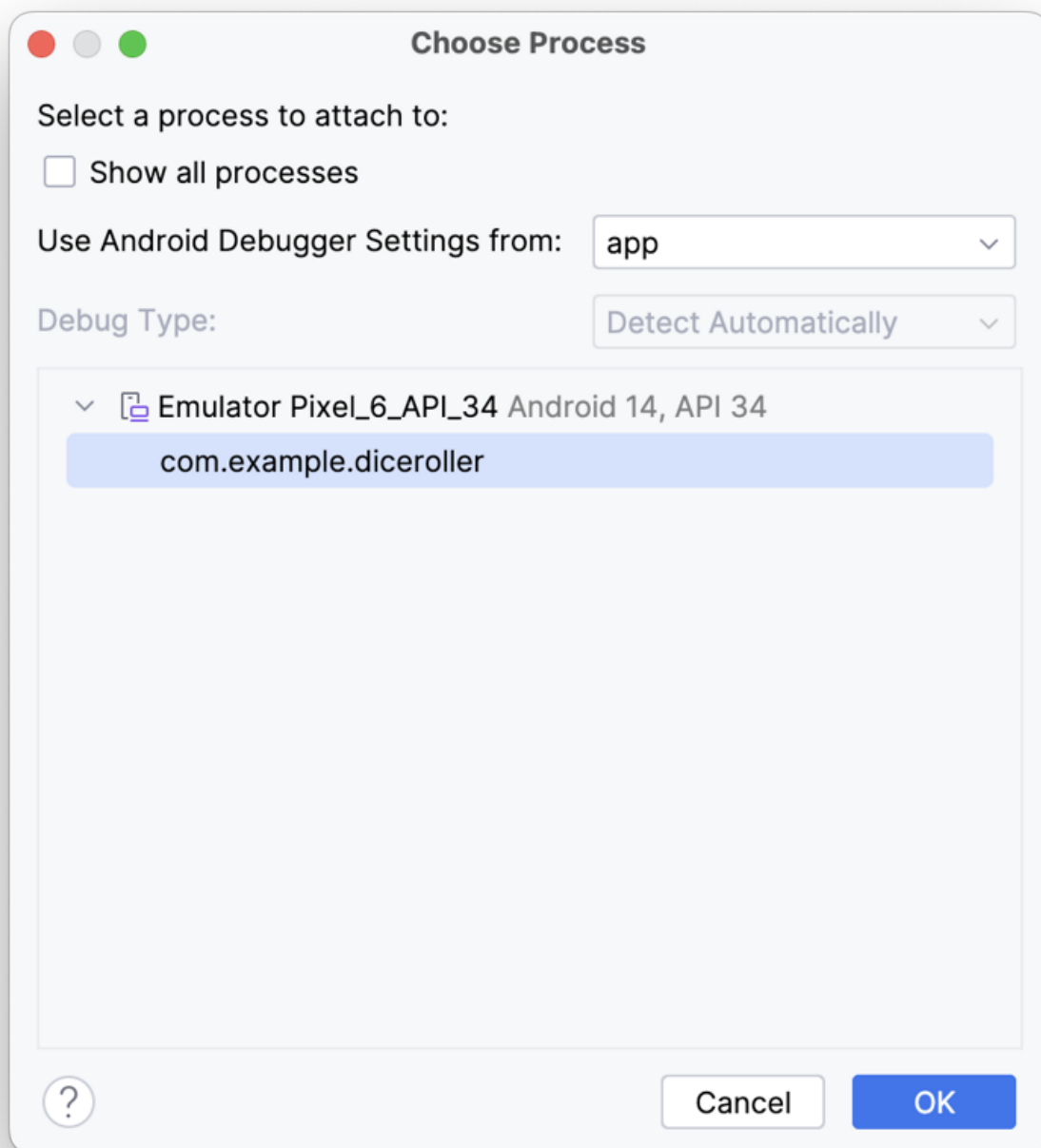
Чтобы прикрепить отладчик к процессу приложения, выполните следующие действия:

- Нажмите Прикрепить отладчик к процессу Android.

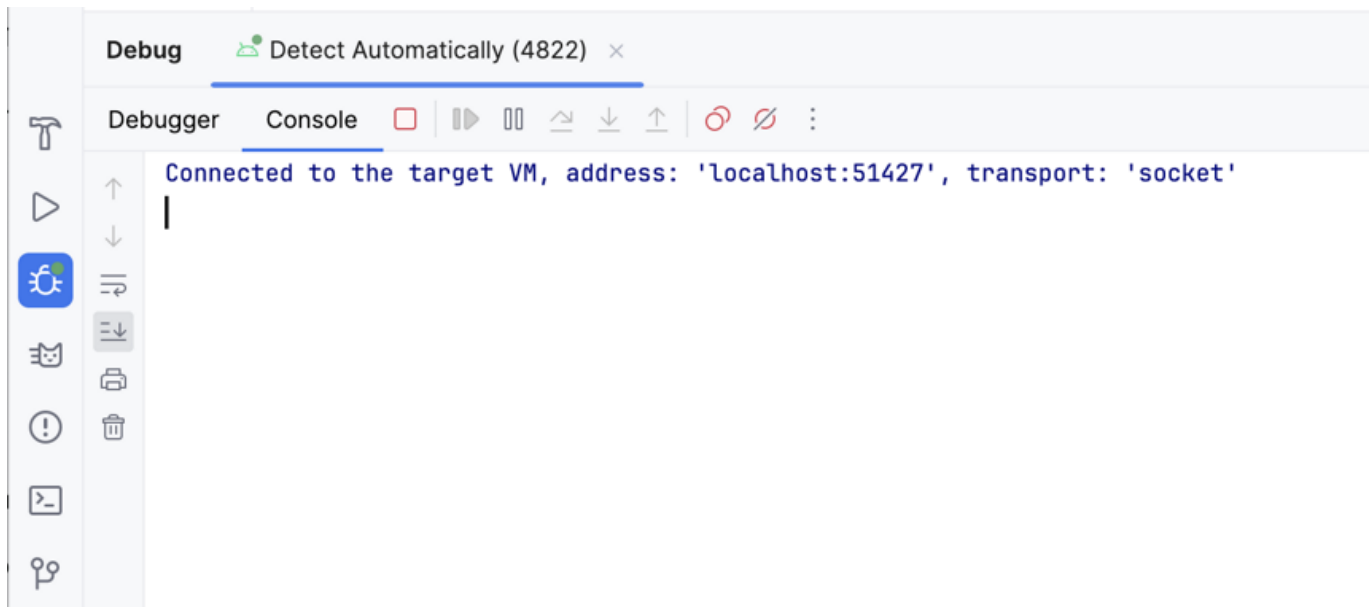


Откроется диалоговое окно Choose Process, в котором вы можете выбрать процесс, к которому хотите прикрепить отладчик.

- Выберите com.example.diceroller и нажмите OK.



В нижней части Android Studio появляется панель Debug с сообщением о том, что отладчик подключен к целевому устройству или эмулятору.

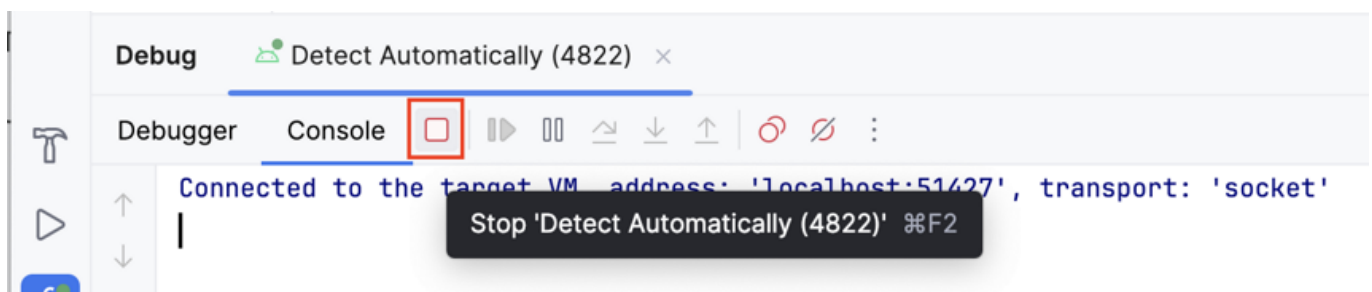


Запуск приложения с отладчиком

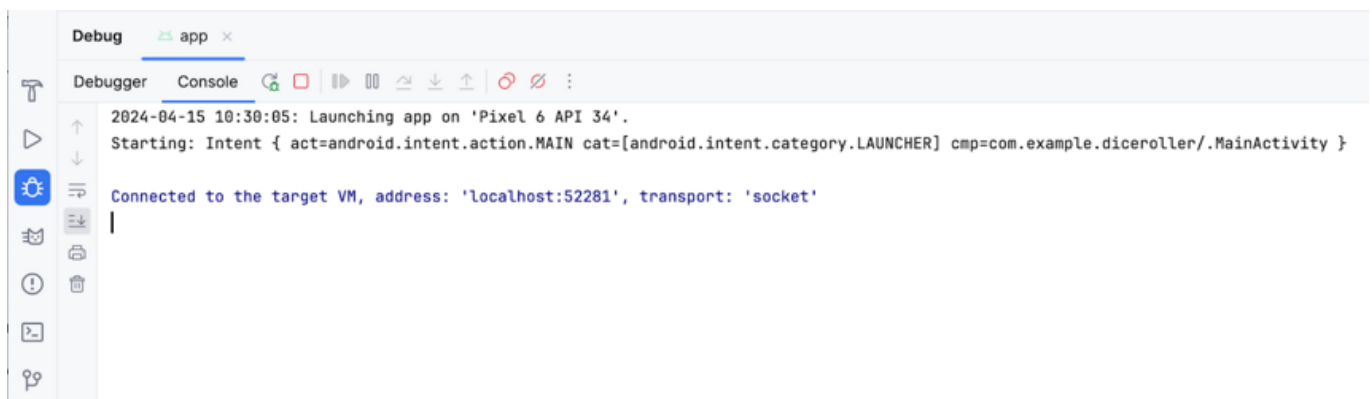
Если вы с самого начала знаете, что хотите использовать отладчик, вы можете сэкономить время, запустив приложение с отладчиком. Кроме того, если вы хотите отладить код, который запускается только при запуске приложения, вам нужно запустить приложение с уже подключенным отладчиком.

Чтобы запустить приложение с отладчиком, выполните следующие действия:

- В панели отладки нажмите Остановить, а затем закройте приложение на устройстве или эмуляторе.



Та же панель Debug появляется в нижней части Android Studio с некоторым консольным выводом.



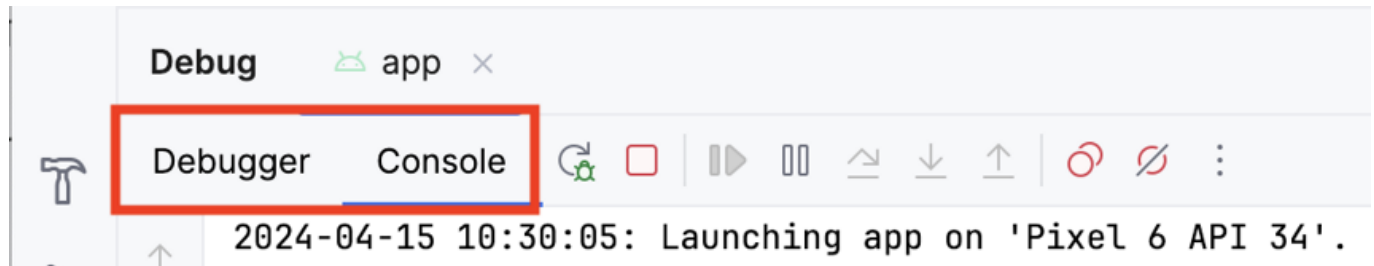
Теперь вы знаете, как запустить отладчик! Далее вы узнаете, как им пользоваться.

Использование отладчика

Панель отладки

Вы, вероятно, заметили, что в верхней части панели отладки находится довольно много кнопок, но сейчас они мало что значат, и большинство из них выделены серым цветом и не доступны для нажатия. В этом разделе рассматриваются наиболее часто используемые функции отладчика. В этом учебном пособии объясняются другие кнопки по мере их появления.

При первом запуске отладчика вы видите ряд кнопок в панели Debug. В верхней части панели отладки находятся кнопки Отладчик и Консоль.



Кнопка Console отображает вывод logcat приложения. Если в вашем коде есть операторы журнала, вывод будет отображаться по мере выполнения этого фрагмента кода.

Кнопка Debugger отображает три отдельные панели, которые сейчас пусты, потому что вы не используете отладчик:

- Отображение фреймов
- Ввод оценок и часовых выражений
- Панель переменных



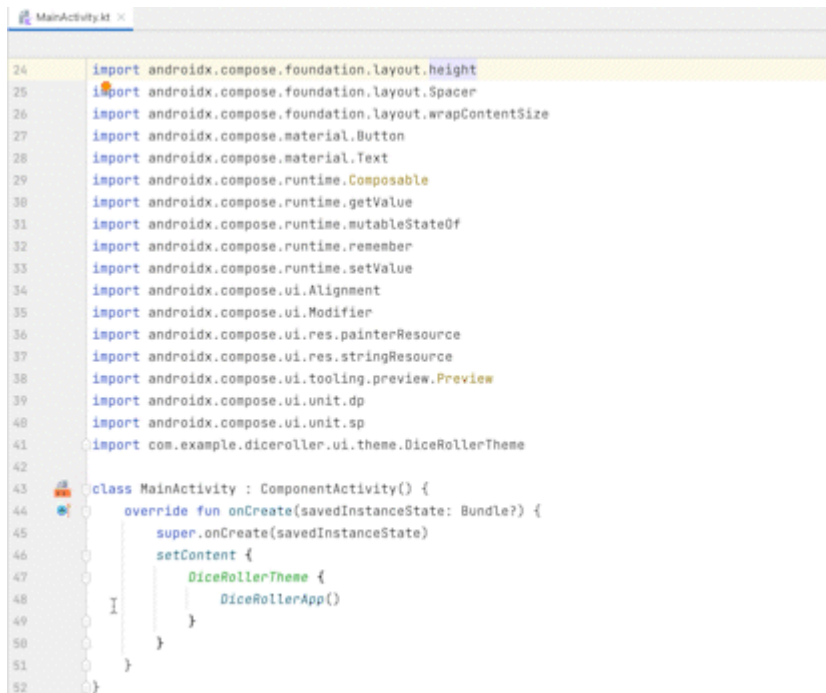
Используйте общие функции отладчика

Установка точки останова

Одна из главных особенностей отладчика - возможность остановить выполнение на определенной строке кода с помощью точки останова.

Чтобы установить точку останова в Android Studio, нужно перейти к определенной строке кода, а затем щелкнуть в желобе рядом с номером строки. Чтобы снять точку останова, нужно щелкнуть по существующей точке останова в желобе, чтобы она исчезла.

Чтобы попробовать это самостоятельно, установите точку останова в том месте, где задана переменная `imageResource`.



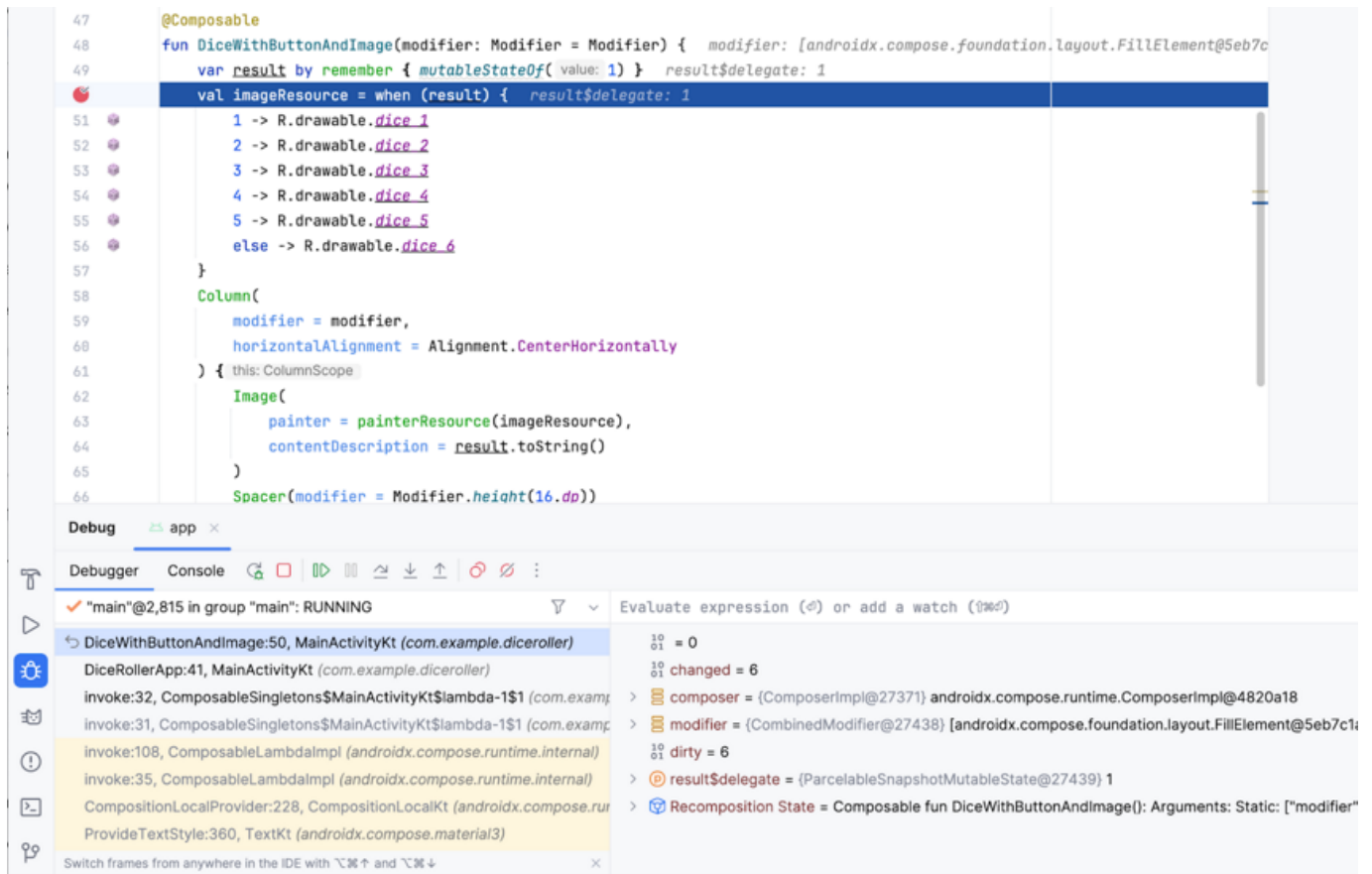
```
24 import androidx.compose.foundation.layout.height
25 import androidx.compose.foundation.layout.Spacer
26 import androidx.compose.foundation.layout.wrapContentSize
27 import androidx.compose.material.Button
28 import androidx.compose.material.Text
29 import androidx.compose.runtime.Composable
30 import androidx.compose.runtime.getValue
31 import androidx.compose.runtime.mutableStateOf
32 import androidx.compose.runtime.remember
33 import androidx.compose.runtime.setValue
34 import androidx.compose.ui.Alignment
35 import androidx.compose.ui.Modifier
36 import androidx.compose.ui.res.painterResource
37 import androidx.compose.ui.res.stringResource
38 import androidx.compose.ui.tooling.preview.Preview
39 import androidx.compose.ui.unit.dp
40 import androidx.compose.ui.unit.sp
41 import com.example.diceroller.ui.theme.DiceRollerTheme
42
43 class MainActivity : ComponentActivity() {
44     override fun onCreate(savedInstanceState: Bundle?) {
45         super.onCreate(savedInstanceState)
46         setContent {
47             DiceRollerTheme {
48                 DiceRollerApp()
49             }
50         }
51     }
52 }
```

Используйте кнопку Возобновить работу программы.

В предыдущем разделе вы установили точку останова в месте установки переменной `imageResource`. Эта точка останова приводит к приостановке выполнения по этой инструкции. Когда выполнение кода приостановлено с помощью отладчика, вам, скорее всего, потребуется продолжить выполнение, чтобы продолжить работу приложения. Самый прямой способ сделать это - использовать кнопку **Resume Program**.

Чтобы возобновить выполнение программы, выполните следующие действия:

- Нажмите Отладка 'app'. После запуска приложения вы должны увидеть что-то вроде этого изображения:



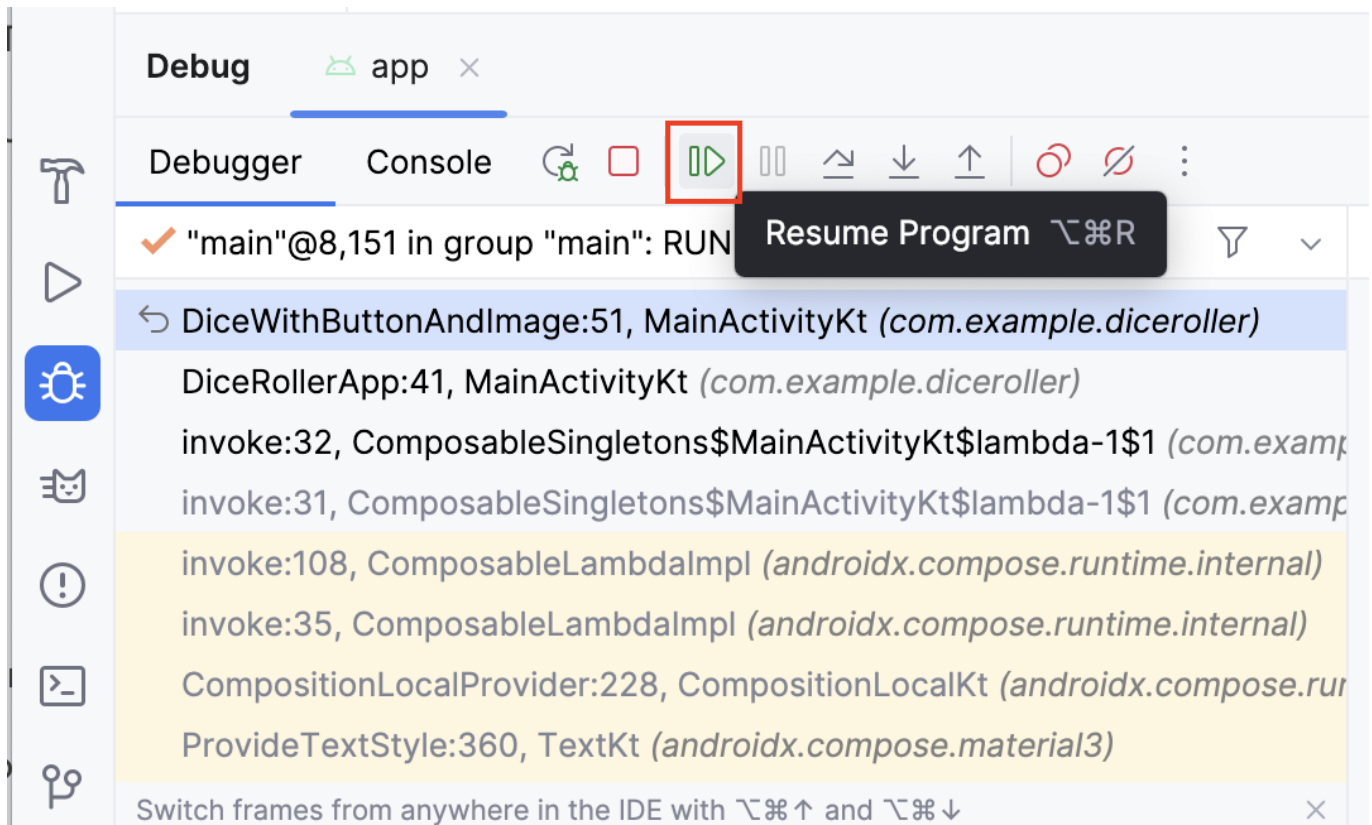
Прежде чем возобновить работу программы, важно объяснить некоторые моменты, которые вы видите на экране, когда отладчик приостанавливает выполнение:

- Многие кнопки на панели Debug теперь можно нажимать.
- На панели Frames отображается много информации, в том числе выделенная ссылка на строку, в которой была установлена точка останова.
- В панели Variables отображается ряд элементов, но в этом приложении не так много переменных, поэтому в данный момент в ней не так много информации, которая актуальна в рамках данного кодаба. Однако возможность просмотра переменных является важной особенностью отладчика, поскольку позволяет понять, что происходит в коде во время выполнения. Более подробно о том, как инспектировать переменные, мы поговорим позже.
- Если вы посмотрите на приложение на своем устройстве или в эмуляторе, то заметите, что экран пуст, потому что приложение приостановилось на строке кода. Точнее, выполнение остановилось на точке останова, и пользовательский интерфейс еще не отрисован.

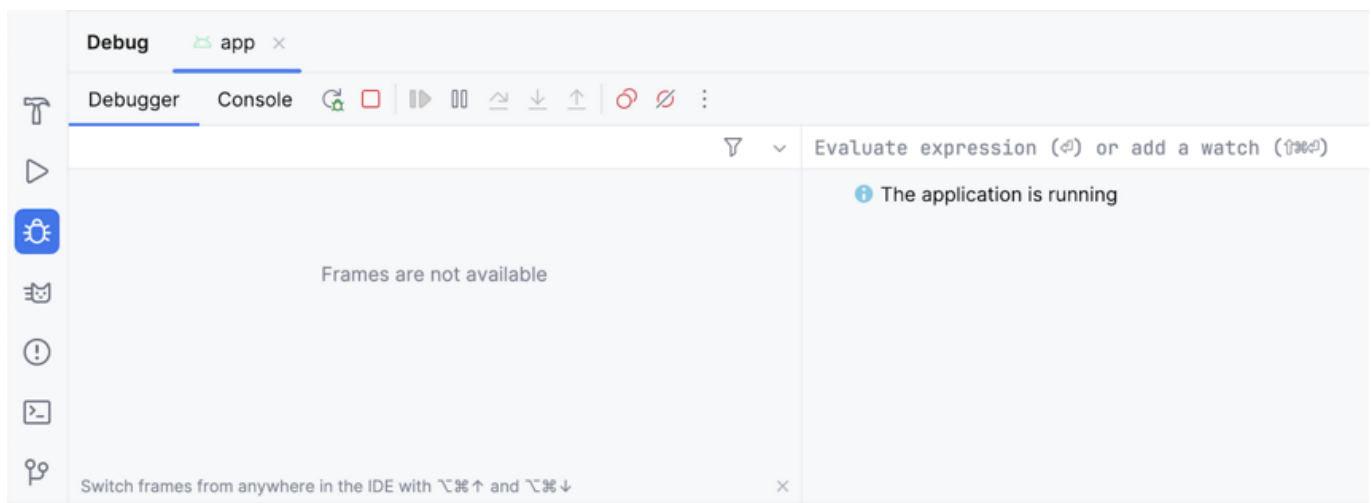
Имейте в виду, что приложение не всегда останавливается немедленно только потому, что была установлена точка останова. Это зависит от того, где вы разместили точку останова в коде. В данном случае вы поместили ее в строку, которая выполняется при запуске приложения.

Главное помнить, что приложение приостанавливается только в точке останова, когда происходит попытка выполнить строку, на которой была установлена точка останова. Существует множество способов переместить отладчик вперед, но сейчас вы используете кнопку Resume Program.

Click !()(https://developer.android.com/static/codelabs/basic-android-kotlin-compose-intro-debugger/img/937f070d95764107_856.png) Resume Program.



Теперь вы должны увидеть нечто похожее на это изображение:



Большая часть информации исчезает, а кнопки снова становятся некликабельными. На устройстве или в эмуляторе приложение также отображается как обычно. Причина в том, что код больше не приостановлен в точке останова и приложение находится в нормальном запущенном состоянии. Отладчик подключен, но он ничего не делает, пока не попытается выполнить строку кода, в которой установлена точка останова. Оставьте эту точку останова на месте, потому что она пригодится в следующих примерах.

Использование кнопки Step Into Кнопка Step Into в отладчике - это удобный способ углубиться в код во время выполнения. Если инструкция вызывает метод или другой фрагмент кода, кнопка Step Into позволяет войти в код без необходимости переходить туда вручную, прежде чем запускать отладчик для установки точки останова.

Чтобы использовать кнопку Step Into, выполните следующие действия:

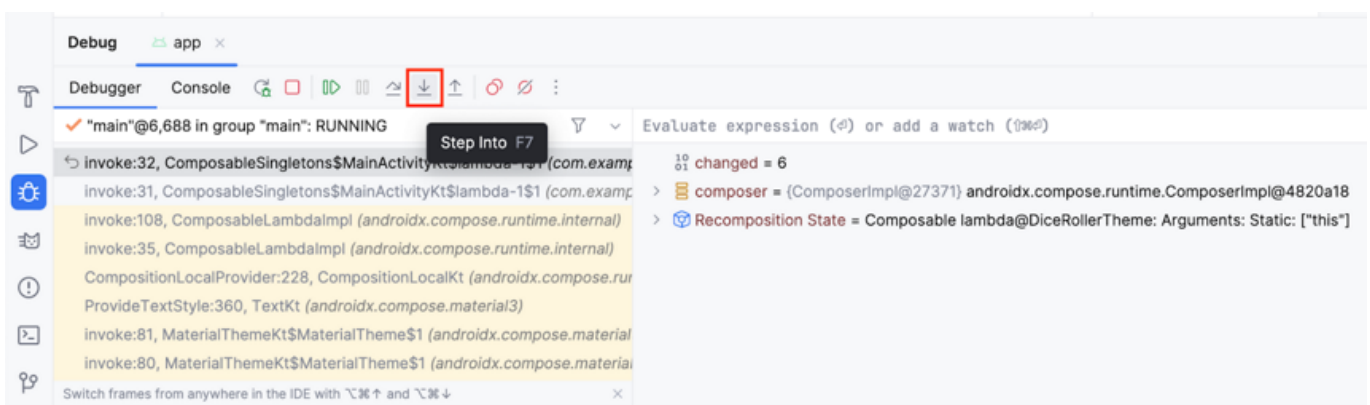
- Создайте точку останова в теле лямбда-функции `setContent` в функции `onCreate()` класса `MainActivity`, где вызывается функция `DiceRollerApp()`.

```

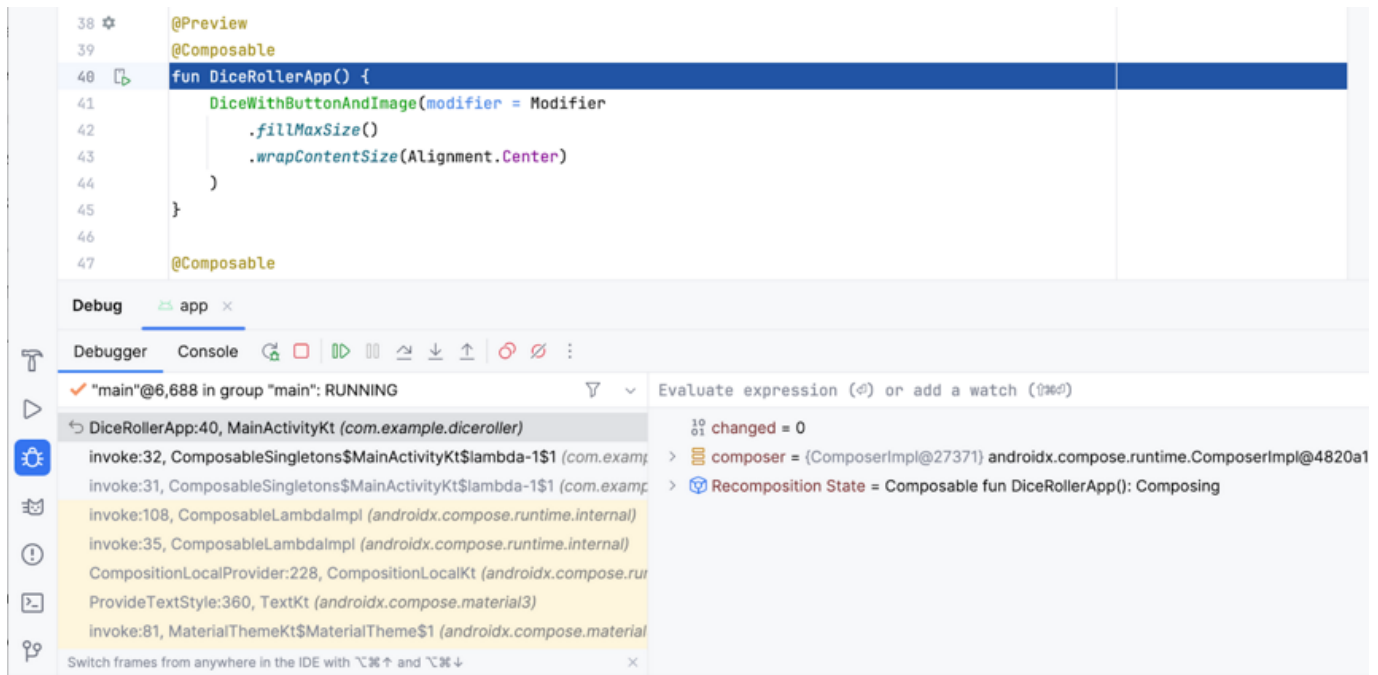
27  <> class MainActivity : ComponentActivity() {
28      @Override fun onCreate(savedInstanceState: Bundle?) {
29          super.onCreate(savedInstanceState)
30          setContent {
31              DiceRollerTheme {
32                  DiceRollerApp()
33              }
34          }
35      }
36  }

```

- Нажмите Debug 'app', чтобы повторно запустить приложение с помощью отладчика. Выполнение приостанавливается на строке, где вызывается функция `DiceRollerApp()`.
- Нажмите Step Into.



Теперь строка 40 выделена, а панель Frames в панели Debug показывает, что код приостановлен на строке 40.

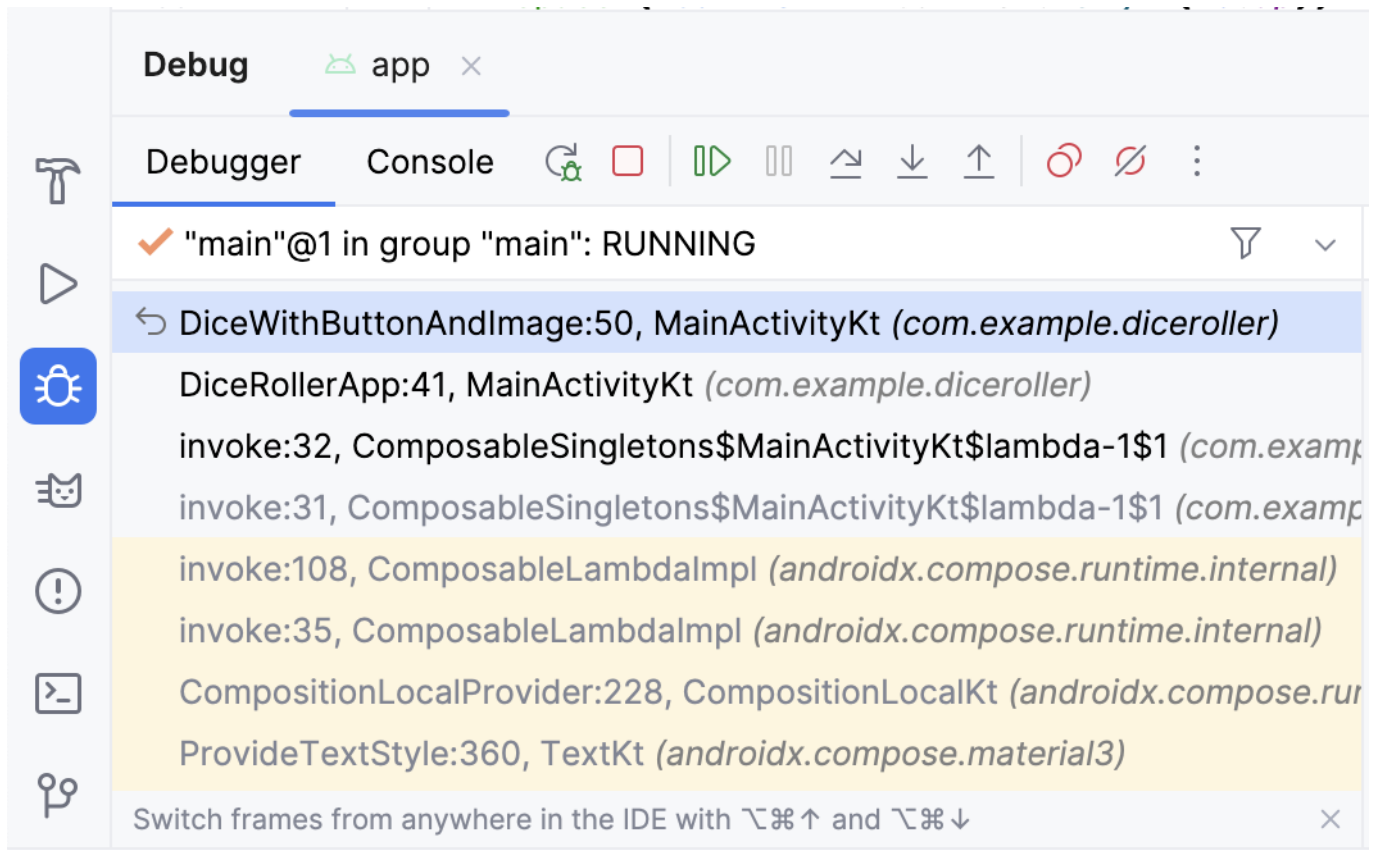


Если вы развернете панель Frames, то увидите, что строка после выделенной строки начинается с `invoke:`, за которой следует номер строки, что на предыдущем изображении равно 32. Это так называемый стек вызовов. По сути, он показывает цепочку вызовов, которые приводят выполнение кода к текущей строке. В данном случае в строке 32 находится инструкция, вызывающая функцию `DiceRollerApp()`.

Когда вы нажали кнопку Step Into, когда отладчик остановился на точке останова, установленной на вызове этой функции, отладчик перешел в эту функцию, что привело выполнение к строке 40, где функция объявлена. Выделенная строка указывает на место приостановки выполнения. Если строки после выделенной строки имеют связанные с ними номера строк, это указывает на путь выполнения. В данном конкретном случае отладчик указывает, что инструкция в строке 32 привела вас к строке 40.

- Нажмите кнопку Возобновить работу программы.

Это должно привести вас к исходной точке останова, которую вы установили. Вы можете понять немного больше о том, что вы видели, когда остановили выполнение в первом примере. Это та же самая картинка с шестого шага раздела «Возобновление программы»:



В стеке вызовов видно, что функция `DiceWithButtonAndImage()` приостановилась на строке 50, а сама она была вызвана со строки 41 в функции `DiceRollerApp()`, которая была вызвана на строке 32. Функция call-stack поможет вам понять путь выполнения. Это очень полезно, когда функция вызывается из разных мест в приложении.

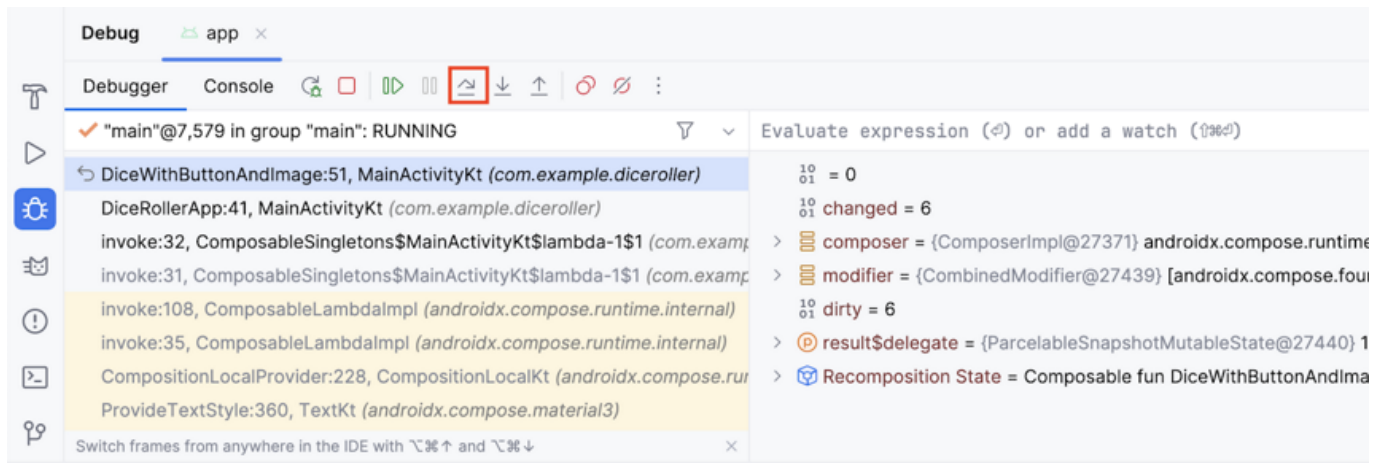
Кнопка **Step Into** позволяет войти в функцию и приостановить ее выполнение, не устанавливая точку останова в самой функции. В данном случае вы установили точку останова на вызове функции `DiceRollerApp()`. Когда вы нажмете кнопку **Step Into**, выполнение приостановится в функции `DiceRollerApp()`.

Dice Roller - довольно маленькое приложение, потому что в нем не так много файлов, классов и функций. Когда вы работаете с более крупными приложениями, функция **Step Into** отладчика становится более полезной, поскольку она дает вам возможность углубиться в код без необходимости самостоятельно перемещаться по нему.

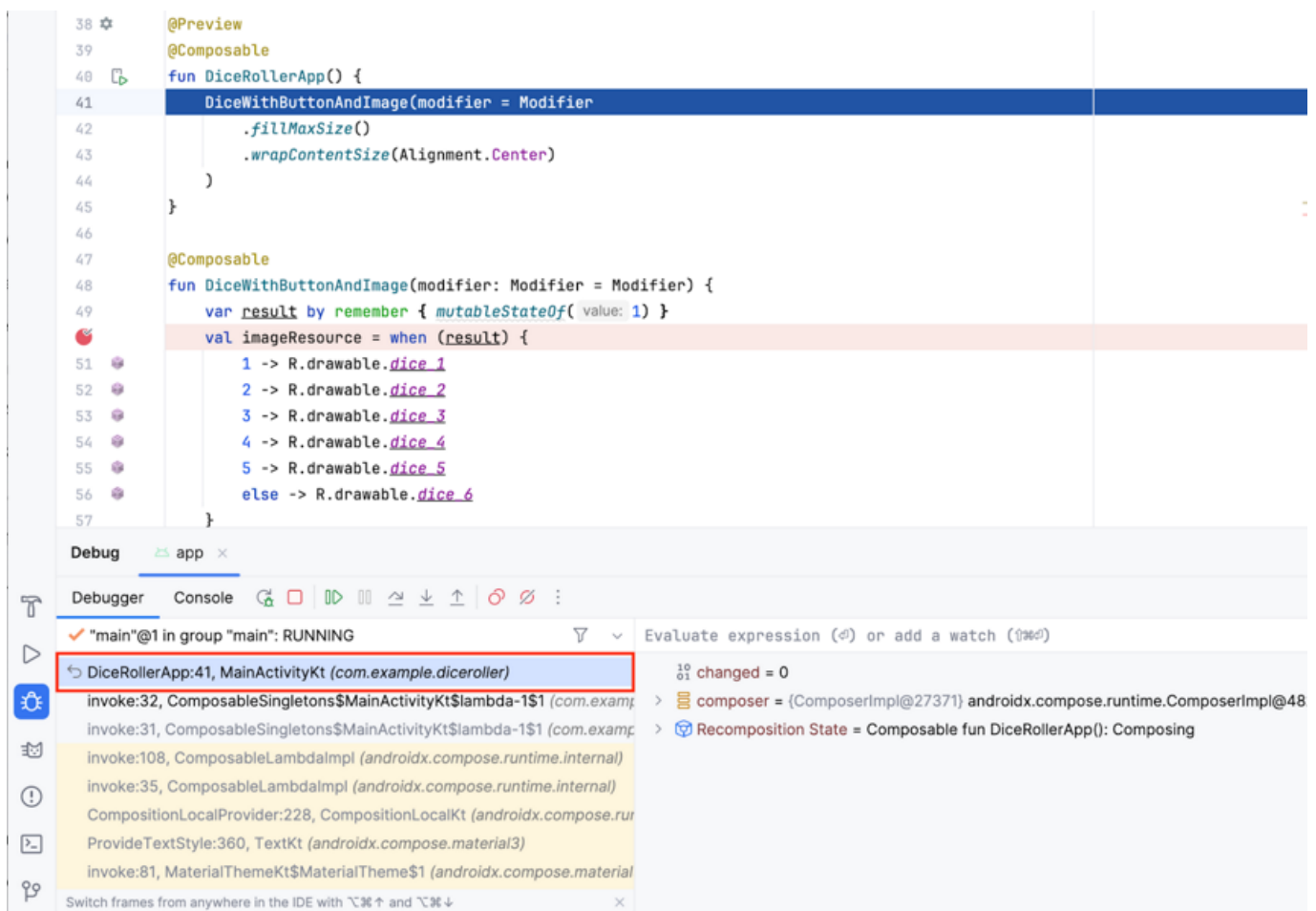
Использование кнопки **Step Over**

Кнопка **Step Over** предоставляет еще одно средство, с помощью которого вы можете перемещаться по коду приложения во время выполнения. Она перемещает выполнение к следующей строке кода и продвигает отладчик.

- Чтобы воспользоваться кнопкой **Step Over**, выполните следующие действия:
- Нажмите **Step Over**.



Теперь вы видите, что отладчик приостановил выполнение кода на следующей строке, которая является строкой 51. Теперь вы можете последовательно пройти по каждой строке.



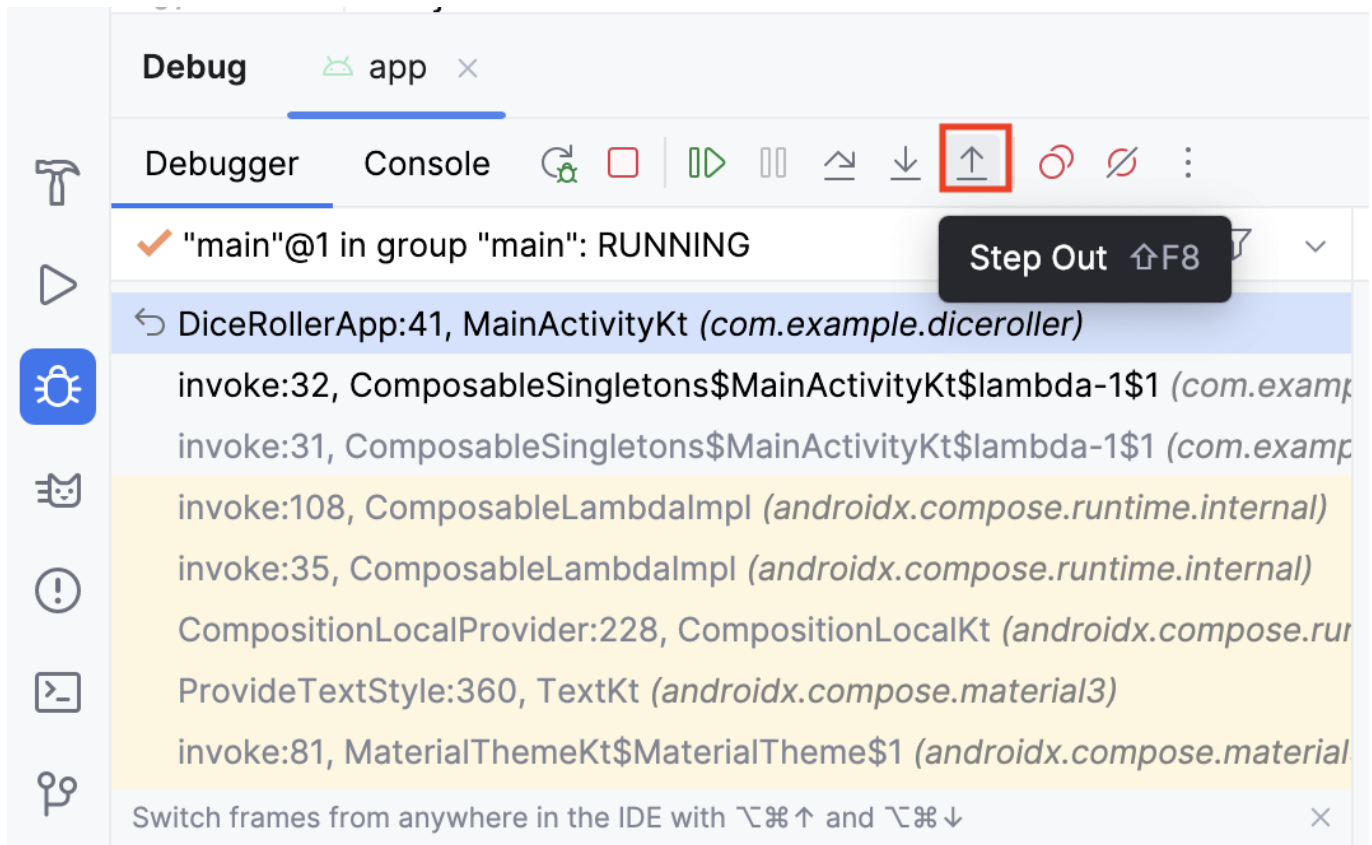
Используйте кнопку Step Out

Кнопка Step Out выполняет действие, противоположное кнопке Step Into. Вместо того чтобы спускаться в стек вызовов, кнопка Step Out перемещает вверх по стеку вызовов.

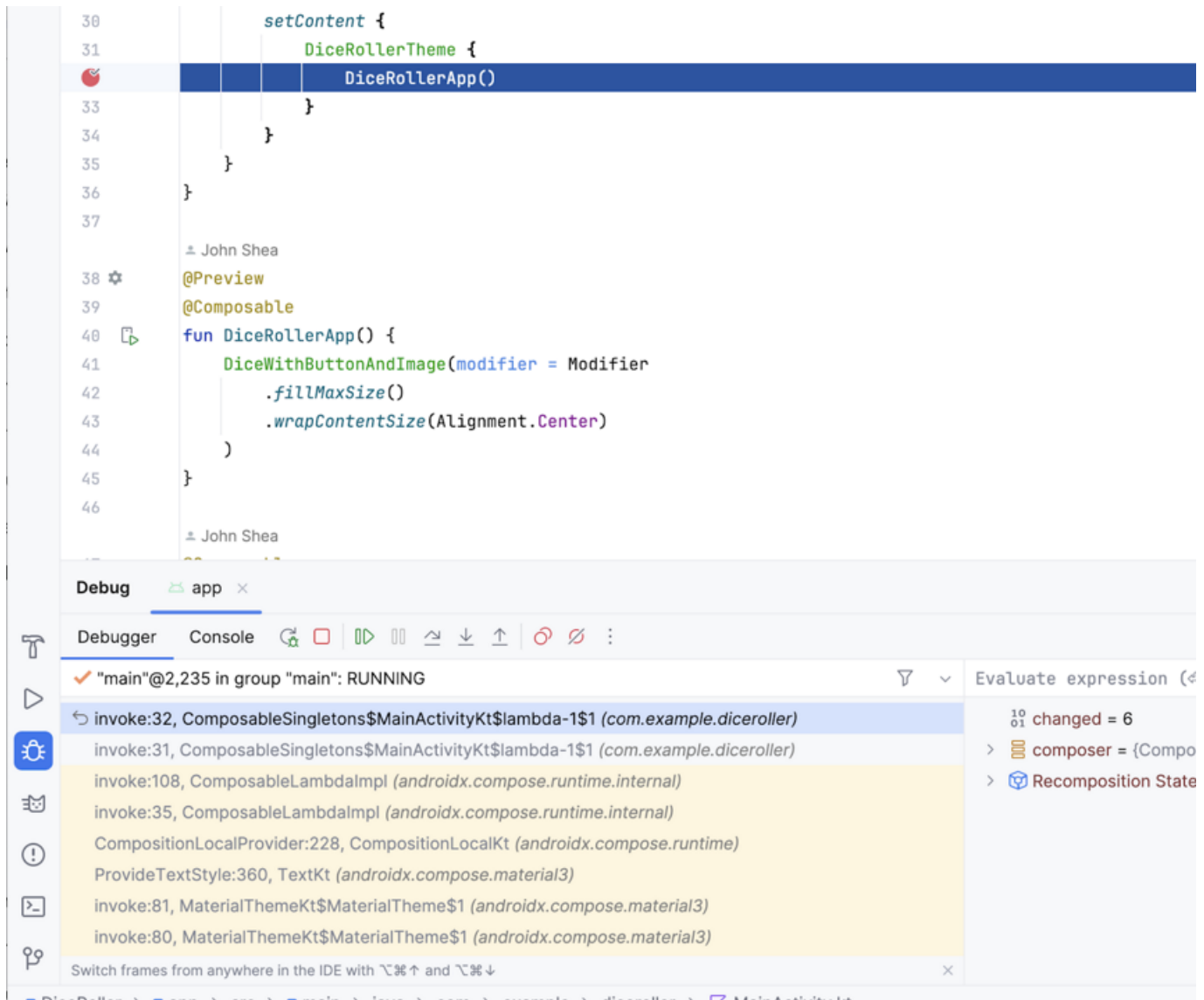
Чтобы воспользоваться кнопкой Step Out, выполните следующие действия:

- Нажмите Step Out.

Можете ли вы угадать, на какой строке программа приостанавливается?



Обратите внимание, что отладчик вышел из функции `DiceRollerApp()` и вернулся к строке, которая ее вызвала.



Кнопка Step Out - полезный инструмент, когда вы оказываетесь слишком глубоко в стеке вызовов методов. Она позволяет проделать путь вверх по стеку вызовов без необходимости просматривать весь код для каждого метода, в который вы зашли.

Проверьте переменные

Ранее было дано краткое описание панели «Переменные». В этой статье вы найдете более подробное объяснение того, как проверять переменные, отображаемые в панели, чтобы помочь вам отладить проблемы в вашем приложении.

Чтобы просмотреть переменные, выполните следующие действия:

- Щелкните на точке останова, чтобы удалить ее с места вызова функции DiceRollerApp(), но оставьте точку останова там, где установлена переменная imageResource.
- Нажмите кнопку Отладка 'app'. Вы должны увидеть, что переменная result\$delegate является MutableState со значением 1. Это потому, что когда переменная определяется, она инстанцируется с mutableStateOf 1. MutableState означает, что переменная result имеет состояние, которое может быть изменено.

Примечание: Переменная result\$delegate в панели переменных относится к переменной result в коде. Обозначение \$delegate используется потому, что переменная result является

запоминающим делегатом.

```

10 01 = 0
10 01 changed = 6
> composer = {ComposerImpl@27429} androidx.compose.runtime.ComposerImpl@101cc71
> modifier = {CombinedModifier@27430} [androidx.compose.foundation.layout.FillElement@cb3f38e8, androidx.compose.foun ... View
10 01 dirty = 6
> result$delegate = {ParcelableSnapshotMutableState@27431} 1
> Recomposition State = Composable fun DiceWithButtonAndImage(): Arguments: Static: ["modifier"]

```

- Нажмите кнопку Программа возобновления.

В приложении нажмите Roll. Ваш код снова приостановится на точке останова, и вы можете увидеть другое значение переменной `result$delegate`. На этом изображении изменяемое состояние переменной `result$delegate` имеет значение 2. Это демонстрирует, как можно проверять переменные во время выполнения с помощью отладчика. В более полнофункциональном приложении значение переменной может привести к аварийному завершению работы. Используя отладчик для проверки переменных, вы можете получить больше информации о деталях сбоя, чтобы исправить ошибку.

```

10 01 = 0
10 01 changed = 7
> composer = {ComposerImpl@27429} androidx.compose.runtime.ComposerImpl@101cc71
> modifier = {CombinedModifier@27430} [androidx.compose.foundation.layout.FillElement@cb3f38e8, androidx.compose.foun ... View
10 01 dirty = 7
> result$delegate = {ParcelableSnapshotMutableState@27431} 2
> Recomposition State = Composable fun DiceWithButtonAndImage(): Recomposition Root

```

Заключение

Поздравляем! Вы воспользовались отладчиком в Android Studio.

Резюме

- Прикрепите отладчик к приложению.
- Запустите приложение с уже подключенным отладчиком.
- Знакомство с панелью отладчика.
- Установите точку останова.
- Возобновление работы программы из отладчика.
- Используйте кнопку Step Into.
- Используйте кнопку Step Over.
- Используйте кнопку Step Out.
- Проверка переменных с помощью отладчика.