<https://developer.android.com/courses/android-basics-compose/course>

### [Основы Android с Compose](https://developer.android.com/courses/android-basics-compose/unit-1)

### https://developer.android.com/courses/android-basics-compose/course

<https://developer.android.com/courses/pathways/android-basics-compose-unit-1-pathway-1>

В этом курсе вы будете создавать приложения, написав код на языке программирования Kotlin, который Google рекомендует при создании новых приложений для Android.

Kotlin — современный язык программирования, который помогает разработчикам работать более продуктивно. Например, Kotlin позволяет вам быть более кратким и писать меньше строк кода для той же функциональности по сравнению с другими языками программирования. Приложения, созданные с использованием Kotlin, также менее склонны к сбоям, что делает приложение более стабильным и надежным для пользователей. По сути, с помощью Kotlin вы можете писать более качественные приложения для Android за более короткое время. В результате Kotlin набирает обороты в отрасли и является языком, который использует большинство профессиональных разработчиков Android.

Чтобы начать создавать приложения для Android на Kotlin, важно сначала заложить прочную основу концепций программирования на Kotlin. С помощью лабораторных работ по этому пути вы изучите основы программирования на Kotlin, прежде чем погрузиться в создание приложений.

## [**2. Начните работу**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#1)

Для написания и редактирования кода вы используете инструмент, называемый редактором кода . Он похож на текстовый редактор, в котором вы можете писать и редактировать текст, но редактор кода также предоставляет функциональные возможности, которые помогут вам писать код более точно. Например, редактор кода отображает предложения автозаполнения во время ввода и отображает сообщения об ошибках, если код неверен.

Чтобы попрактиковаться в основах языка Kotlin, вы будете использовать интерактивный редактор кода под названием Kotlin Playground. Вы можете получить к нему доступ через веб-браузер, поэтому вам не нужно устанавливать какое-либо программное обеспечение на свой компьютер.

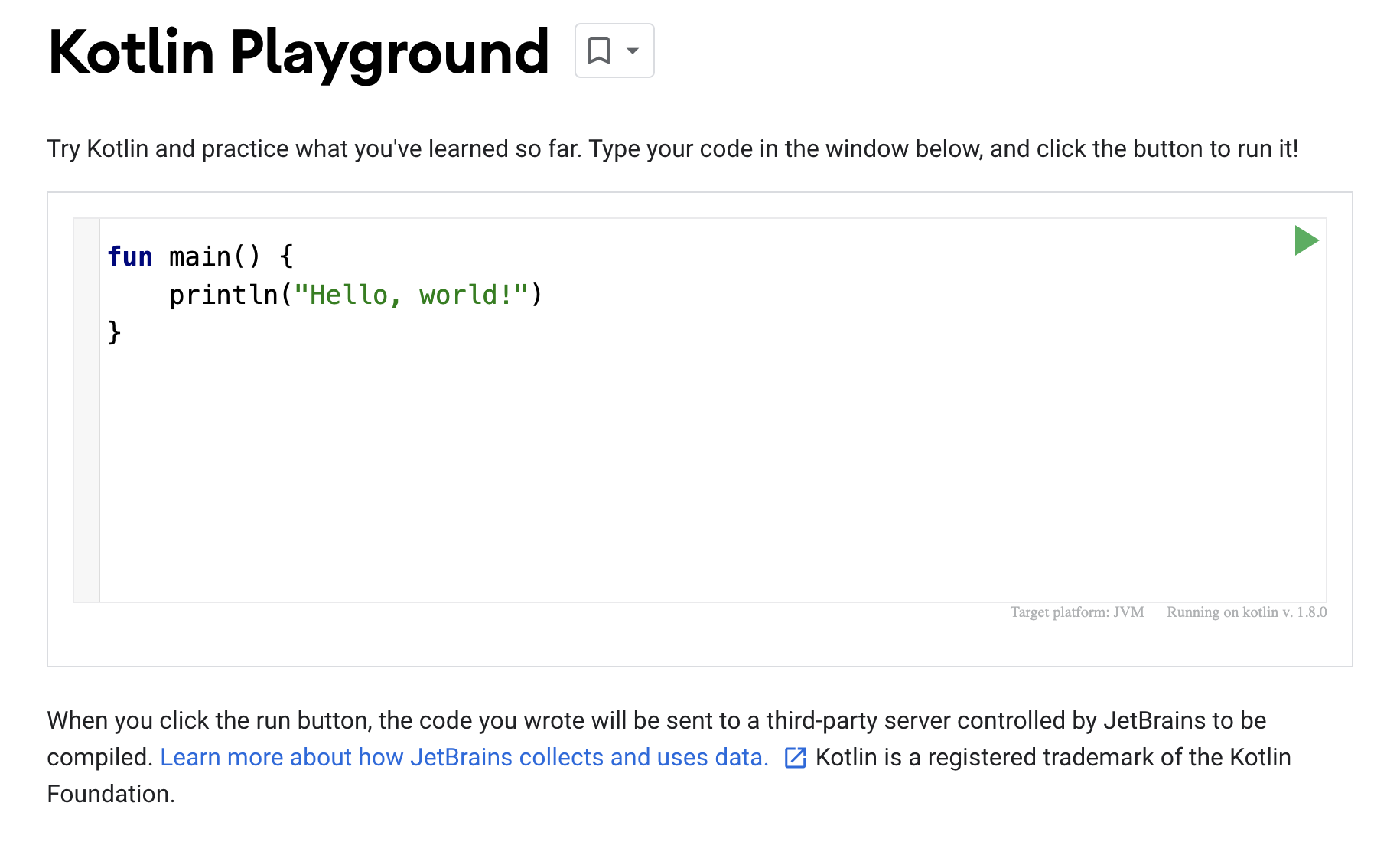
Обратите внимание, что вы не можете создавать приложения для Android на Kotlin Playground. На более поздних этапах вы установите и будете использовать инструмент под названием Android Studio для написания и редактирования кода вашего приложения Android.

Теперь, когда у вас есть некоторое представление о Kotlin, взгляните на свою первую программу!

## [**3. Открытая площадка Kotlin**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#2)

В веб-браузере на вашем компьютере откройте [Kotlin Playground](https://developer.android.com/training/kotlinplayground" \t "_blank) .

Вы должны увидеть веб-страницу, похожую на это изображение:



В редакторе кода уже есть код по умолчанию. Эти три строки кода составляют простую программу:

fun main() {  
    println("Hello, world!")  
}

Даже если вы никогда раньше не программировали, сможете ли вы догадаться, что делает программа?

Проверьте, верна ли ваша догадка.

## [**4. Запустите свою первую программу**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#3)

Нажмите Кнопка запуска, чтобы запустить программу.

Когда вы нажимаете кнопку «Выполнить», происходит многое. Код на языке программирования Kotlin должен быть понятен людям, чтобы им было легче читать, писать и совместно работать над программами Kotlin. Однако ваш компьютер не сразу понимает этот язык.

Вам нужно что-то, называемое компилятором Kotlin , который берет написанный вами код Kotlin, просматривает его построчно и переводит во что-то, что может понять компьютер. Этот процесс называется компиляцией кода.

Если ваш код успешно скомпилируется, ваша программа запустится (или выполнится). Когда компьютер выполняет вашу программу, он выполняет каждую из ваших инструкций. Если вы когда-либо следовали рецепту приготовления, выполнение каждого шага рецепта считается выполнением каждой инструкции.

Вот скриншот того, что вы должны увидеть при запуске программы.

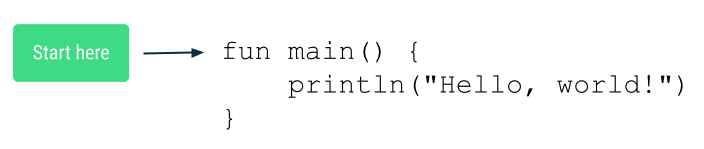


В нижней части редактора кода вы должны увидеть панель, показывающую выходные данные или результат вашей программы:

Hello, world!

Цель этой программы — распечатать или отобразить сообщение с надписью Hello, world!.

Как это работает? Программа Kotlin должна иметь функцию main — определенное место в вашем коде, где программа начинает работать. Основная функция — это точка входа или отправная точка программы.



Теперь вам может быть интересно, что такое функция?

## [**5. Части функции**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#4)

Функция — это сегмент программы, выполняющий определенную задачу. Ваша программа может иметь одну или несколько функций.

## **Определить или вызвать функцию**

В вашем коде вы сначала определяете функцию. Это означает, что вы указываете все инструкции, необходимые для выполнения этой задачи.

Как только функция определена, вы можете вызвать ее, чтобы можно было выполнить инструкции внутри этой функции.

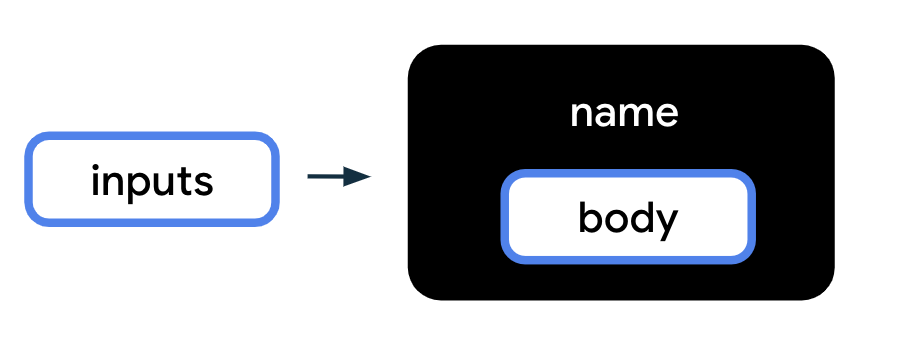
Вот аналогия. Вы записываете пошаговую инструкцию, как испечь шоколадный торт. Этот набор инструкций может иметь имя: bakeChocolateCake. Каждый раз, когда вы захотите испечь торт, вы можете выполнить bakeChocolateCakeинструкции. Если вы хотите 3 торта, вам нужно выполнить bakeChocolateCake инструкцию 3 раза. Первым шагом является определение шагов и присвоение им имени, которое считается определением функции. Затем вы можете обращаться к шагам в любое время, когда захотите, чтобы они были выполнены, что считается вызовом функции.

**Примечание.** Вы можете услышать альтернативную фразу «объявить функцию». Слова объявлять и определять могут использоваться взаимозаменяемо и иметь одно и то же значение. Вы также можете услышать термин «определение функции» или «объявление функции», который относится к точному коду, определяющему функцию. В некоторых других языках программирования объявления и определения имеют разные значения.

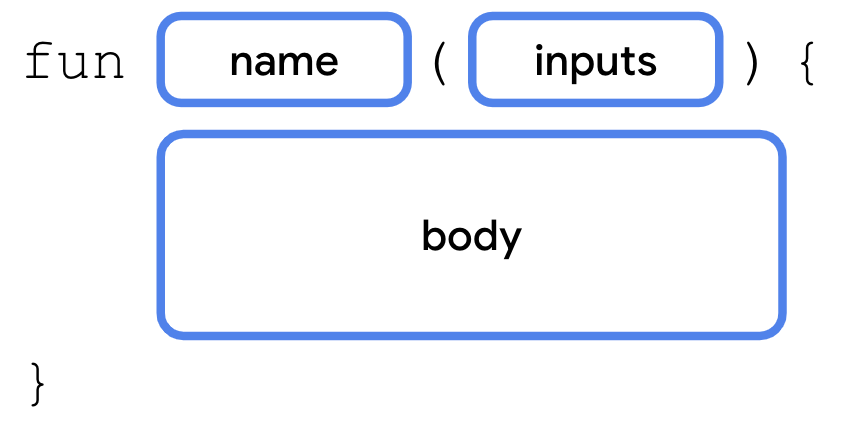
## **Определить функцию**

Вот ключевые части, необходимые для определения функции:

* Функции нужно **имя** , чтобы вы могли вызвать ее позже.
* Функция также может требовать некоторых **входных данных** или информации, которую необходимо предоставить при вызове функции. Функция использует эти входные данные для достижения своей цели. Требование ввода не является обязательным, а некоторые функции не требуют ввода.
* Функция также имеет **тело** , содержащее инструкции по выполнению задачи.

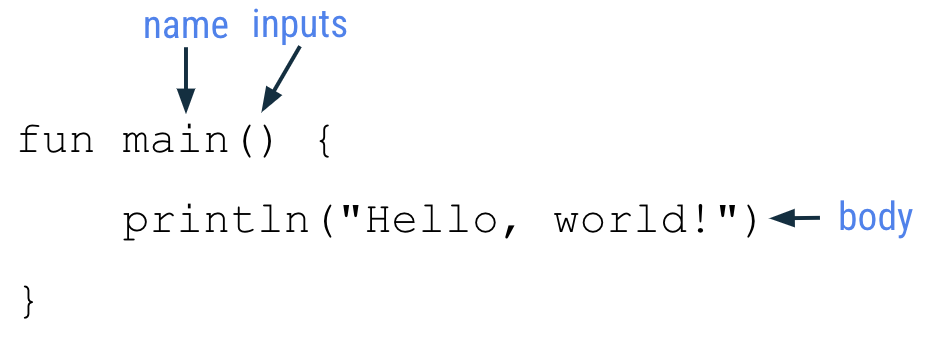


Чтобы перевести приведенную выше диаграмму в код Kotlin, используйте следующий синтаксис или формат для определения функции. Порядок этих элементов имеет значение. Сначала должно идти слово fun, затем имя функции, затем вводимые данные в круглых скобках, а затем фигурные скобки вокруг тела функции.



Обратите внимание на ключевые части функции main в примере, который вы видели на Kotlin Playground:

* Определение функции начинается со слова fun.
* Тогда имя функции main.
* У функции нет входных данных, поэтому круглые скобки пусты.
* В теле функции есть одна строка кода println("Hello, world!"), которая расположена между открывающей и закрывающей фигурными скобками функции.



Каждая часть функции описана более подробно ниже.

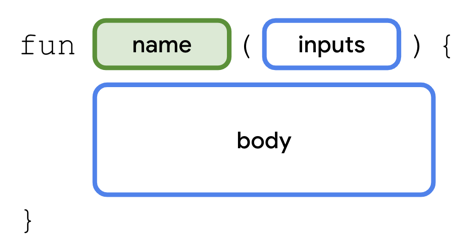
### ****Ключевое слово функции****

Чтобы указать, что вы собираетесь определить функцию в Kotlin, используйте специальное слово fun(сокращение от function) в новой строке. Вы должны ввести funточно так, как показано строчными буквами. Вы не можете использовать func, function или какое-либо альтернативное написание, потому что компилятор Kotlin не распознает, что вы имеете в виду.

Эти специальные слова называются ключевыми словами в Котлине и зарезервированы для определенной цели, например, для создания новой функции в Котлине.

### ****Имя функции****

Функции имеют имена, поэтому их можно отличить друг от друга, подобно тому, как люди имеют имена, чтобы идентифицировать себя. Имя функции находится после ключевого слова fun.



Выберите подходящее имя для вашей функции в зависимости от ее назначения. Имя обычно представляет собой глагол или глагольную фразу. Рекомендуется избегать использования [ключевого слова Kotlin](https://kotlinlang.org/docs/keyword-reference.html) в качестве имени функции.

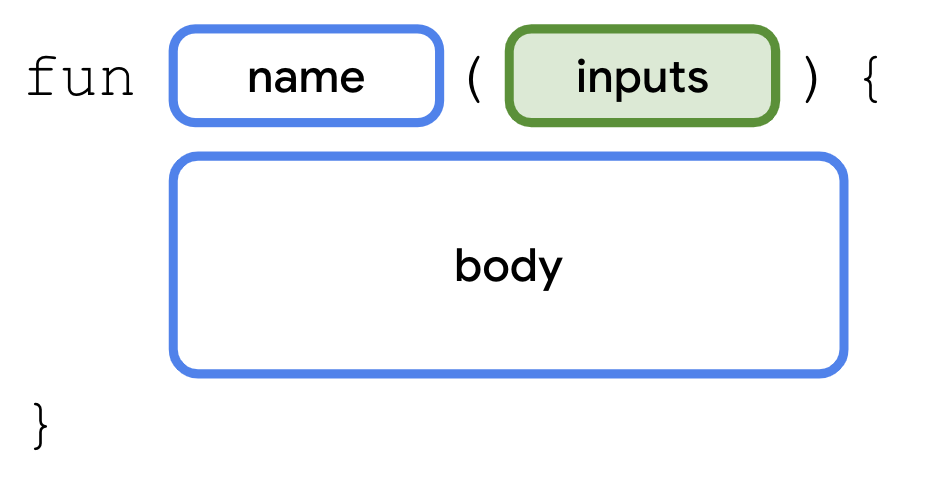
Имена функций должны соответствовать соглашению camelCase, при котором первое слово имени функции должно быть в нижнем регистре. Если в имени несколько слов, пробелов между ними нет, а все остальные слова должны начинаться с заглавной буквы.

Примеры имен функций:

* calculateTip
* displayErrorMessage
* takePhoto

### ****Функциональные входы****

Обратите внимание, что за именем функции всегда следуют круглые скобки. В этих круглых скобках вы указываете входные данные для функции.



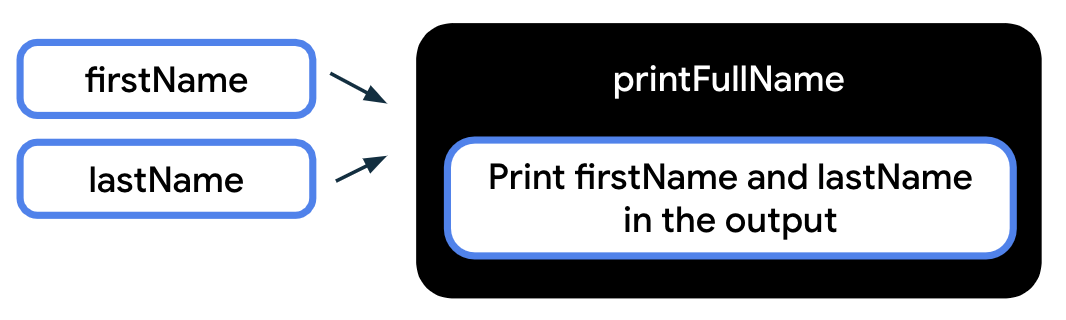
Входные данные — это фрагмент данных, который необходим функции для выполнения своей цели. Когда вы определяете функцию, вы можете потребовать, чтобы определенные входные данные передавались при вызове функции. Если для функции не требуются входные данные, круглые скобки пусты ().

Вот несколько примеров функций с разным количеством входов:

На следующей диаграмме показана функция с именем addOne. Цель функции — прибавить 1 к заданному числу. Существует один вход, который представляет собой заданное число. Внутри тела функции есть код, который добавляет 1 к числу, переданному в функцию.



В следующем примере есть функция под названием printFullName. Для функции необходимы два входа: один для имени и один для фамилии. Тело функции выводит на выходе имя и фамилию, чтобы отобразить полное имя человека.

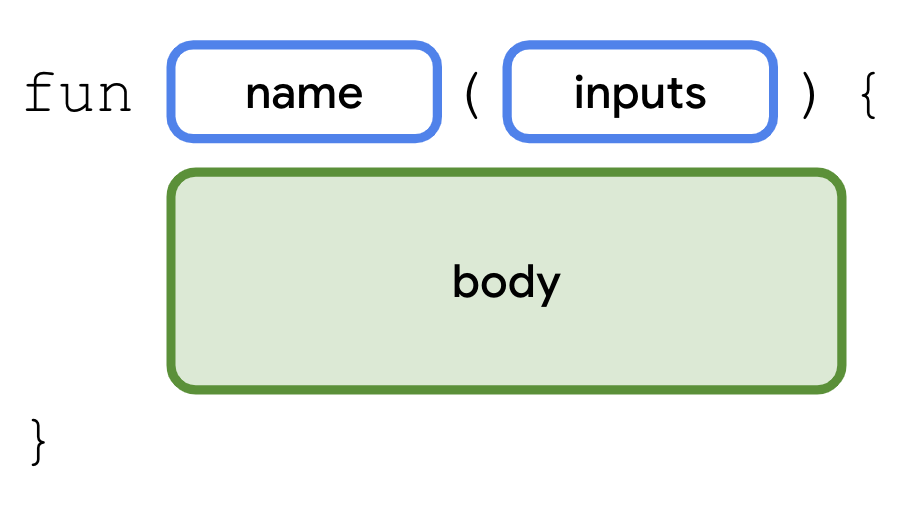


В этом последнем примере показана функция, которая не требует передачи каких-либо входных данных при вызове функции. Когда вы вызываете displayHello() функцию, на выходе выводится сообщение Hello.



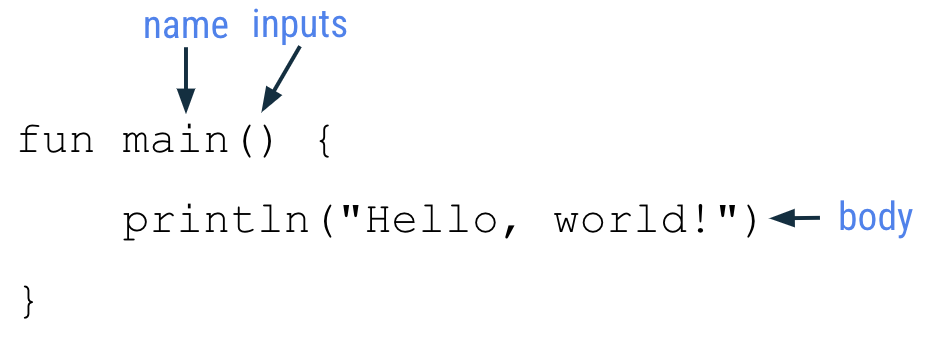
### ****Тело функции****

Тело функции содержит инструкции, необходимые для достижения цели функции. Вы можете найти тело функции, найдя строки кода, заключенные в открывающие и закрывающие фигурные скобки (body).



## **Объяснение простой программы**

Вернитесь к простой программе, которую вы видели ранее.



Программа содержит одну функцию: mainfunction. main— это специальное имя функции в Котлине. Когда вы пишете свой код на Kotlin Playground, ваш код должен быть написан внутри функции main()или вызываться из main()функции.

В теле этой функции всего одна строка кода main():

println("Hello, world!")

Эта строка кода является оператором, поскольку она выполняет определенное действие, а именно распечатывает текст Hello, world! на панели вывода. Точнее, функция println() вызывается в этой строке кода.

[println()](https://kotlinlang.org/api/latest/jvm/stdlib/kotlin.io/println.html)— это функция, которая уже определена в языке Kotlin. Это означает, что команда инженеров, создавшая язык Kotlin, уже написала объявление функции println(). Для функции требуется один ввод — сообщение, которое должно быть напечатано.

При вызове функции println() помещайте текст сообщения в круглые скобки после имени функции. Обязательно используйте кавычки вокруг отображаемого текста, например "Hello, world!".

Когда программа выполняется, сообщение, переданное в функцию println(), выводится на выход:

Hello, world!

## **Попробуй**

Теперь вернемся к исходному коду программы. Можете ли вы изменить код Kotlin Playground, чтобы вместо этого в выводе отображалось это сообщение?

Hello, Android!

## [**6. Измените свою программу**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#5)

Изменяем в онлайн редакторе ->

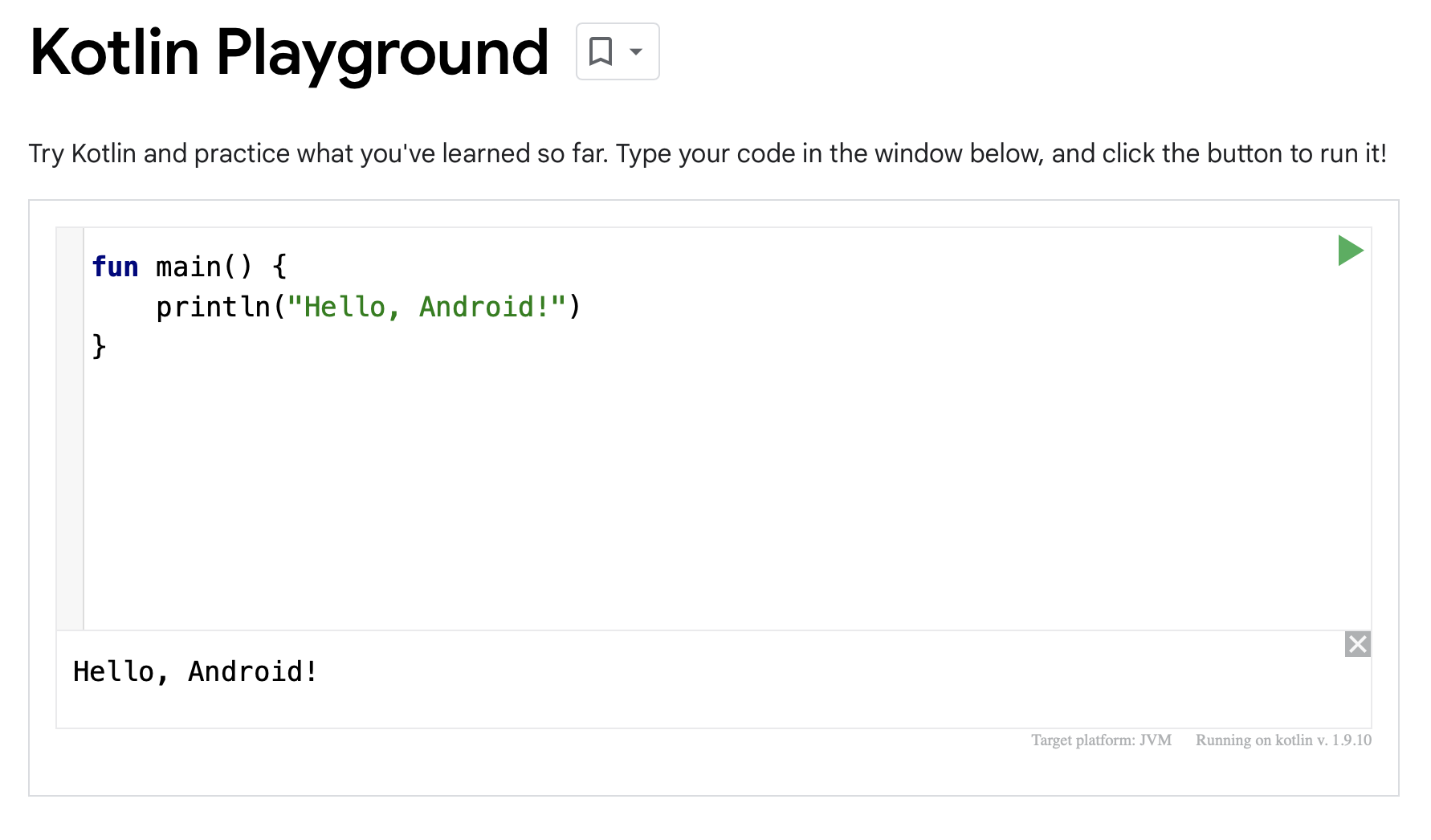
<https://developer.android.com/training/kotlinplayground>

1. Чтобы изменить сообщение, отображаемое на выходе, измените  вызов функции println()во второй строке программы. Замените world на Android в функции println(). Убедитесь, что "Hello, Android!" оно все еще находится в кавычках и скобках.

fun main() {  
    println("Hello, Android!")  
}

1. Запустите программу.
2. Вывод должен показать это сообщение:

Hello, Android!



Отличная работа, вы изменили свою первую программу!

Теперь можете ли вы изменить свой код, чтобы сообщение печаталось дважды? Посмотрите желаемый результат:

Hello, Android!

Hello, Android!

## **Распечатать более одного сообщения**

Вы можете поместить в функцию столько строк инструкций, сколько необходимо для выполнения задачи. Однако обратите внимание, что в Котлине в каждой строке должно быть только одно выражение. Если вы хотите написать еще один оператор, поместите его в новую строку функции.

Чтобы изменить вашу программу для печати нескольких строк текста:

1. Скопируйте исходный println()оператор и вставьте второй оператор под ним в тело функции. Оба println()оператора должны содержаться в фигурных скобках основной функции. Теперь у вас есть два оператора в теле функции.

fun main() {  
    println("Hello, Android!")  
    println("Hello, Android!")  
}

1. Запустите программу.
2. Когда вы запустите программу, результат должен быть таким:

Hello, Android!

Hello, Android!

Вы можете увидеть, как изменения в коде влияют на результат.

1. Измените код так, чтобы там было написано .Hello, *YOUR\_NAME*!

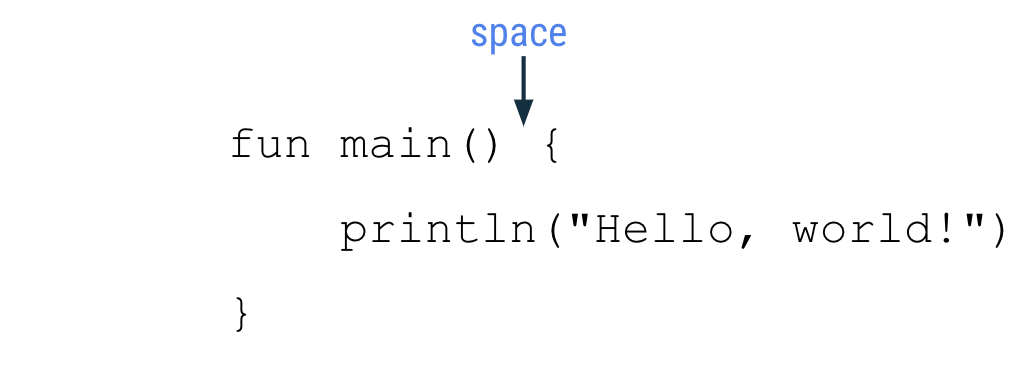
[7. Руководство по стилю Kotlin](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#6)

Хороший метод кодирования — следовать стандартам кодирования Google для Android для кода, написанного на Kotlin. Полное руководство называется руководством по стилю и объясняет, как код следует форматировать с точки зрения внешнего вида, а также соглашения, которым следует следовать при написании кода. Например, руководство по стилю включает рекомендации по использованию пробелов, отступов, именования и т. д.

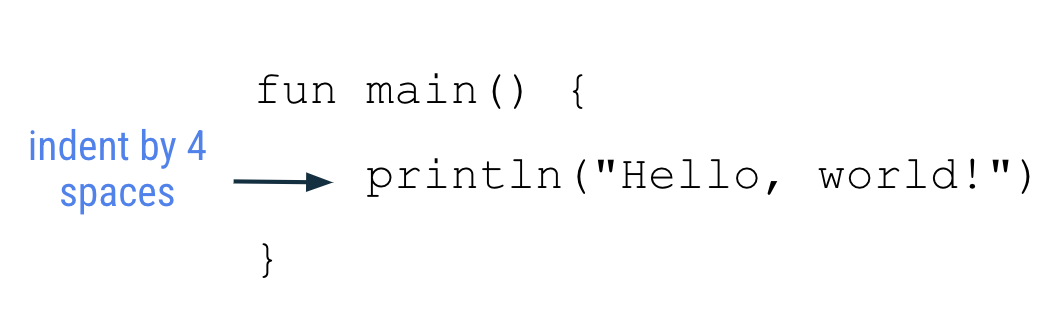
Цель следования руководству по стилю — сделать ваш код более удобным для чтения и более совместимым с тем, как пишут свой код другие разработчики Android. Такая согласованность важна при совместной работе над большими проектами, чтобы стиль кода был одинаковым для всех файлов в проекте.

Вот некоторые из соответствующих рекомендаций по стилю для того, что вы уже изучили в Kotlin:

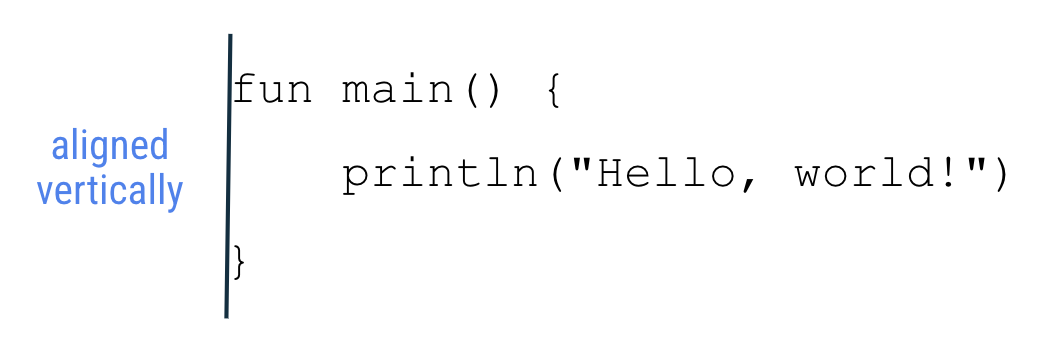
* Имена функций должны быть написаны в верблюжьем регистре и представлять собой глаголы или глагольные фразы.
* Каждый оператор должен находиться на отдельной строке.
* Открывающая фигурная скобка должна появиться в конце строки, где начинается функция.
* Перед открывающей фигурной скобкой должен быть пробел.



* Тело функции должно иметь отступ в 4 пробела. Не используйте символ табуляции для отступа в коде, введите 4 пробела.



* Закрывающая фигурная скобка находится на отдельной строке после последней строки кода в теле функции. Закрывающая скобка должна соответствовать ключевому слову fun в начале функции.



По мере приобретения знаний в Kotlin вы узнаете больше о правилах кодирования Android. Полное руководство по стилю находится [здесь](https://developer.android.com/kotlin/style-guide) , но не волнуйтесь, оно охватывает и другие темы Kotlin, которые вы еще не изучили.

## [**8. Исправьте ошибки в вашем коде**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#7)

Когда вы изучаете человеческий язык, существуют правила синтаксиса и грамматики, позволяющие правильно использовать слова и формировать предложения. Аналогично, для языков программирования существуют определенные правила, определяющие, что делает код действительным, то есть кодом, который успешно компилируется.

Обычная часть процесса кодирования включает в себя ошибки и случайное написание неверного кода. Новичка эти ошибки могут сбить с толку или утомить. Но не волнуйтесь, это ожидаемо. Код редко работает идеально с первого раза. Точно так же, как написание документа требует множества черновиков, написание кода может занять много итераций, пока он не заработает должным образом.

Если код не удалось успешно скомпилировать, возникла ошибка. Например, если у вас есть опечатка, например, отсутствует кавычка или скобка, компилятор не поймет ваш код и не сможет преобразовать его в шаги, которые сможет выполнить компьютер. Если ваш код не работает должным образом или вы видите сообщение об ошибке в редакторе кода, вам придется вернуться к своему коду и исправить его. Процесс устранения этих ошибок называется устранением неполадок.

1. Скопируйте и вставьте следующий фрагмент кода в Kotlin Playground и запустите программу. Что ты видишь?

fun main() {  
    println("Today is sunny!)  
}

В идеале вы хотите, чтобы сообщение Today is sunny! отображалось. Вместо этого на панели вывода вы видите значки восклицательных знаков с сообщениями об ошибках.



Сообщение об ошибке от Kotlin Playground

Сообщения об ошибках начинаются со слова «Expecting (Ожидание)», потому что компилятор Kotlin что-то «ожидает», но не находит этого в коде. В этом случае компилятор ожидает закрывающую кавычку и закрывающую скобку для кода во второй строке вашей программы.

Обратите внимание, что в операторе println() отображаемое сообщение имеет открывающую кавычку, но не имеет закрывающей кавычки. Несмотря на то, что в коде есть закрывающая скобка, компилятор считает, что скобка является частью печатаемого текста, поскольку перед ней нет закрывающей кавычки.

1. Добавьте закрывающую кавычку после восклицательного знака перед закрывающей скобкой.

fun main() {  
    println("Today is sunny!")  
}

Основная функция содержит одну строку кода, представляющую собой оператор println(), текст которого заключен в кавычки и помещен в круглые скобки: "Today is sunny!".

1. Запустите программу еще раз.

Никаких ошибок быть не должно, и на панели вывода должен отображаться этот текст:

Today is sunny!

Удачи в исправлении ошибки! Как только вы приобретете больше опыта в написании кода и устранении ошибок, вы поймете, насколько важно обращать внимание на использование заглавных букв, орфографию, интервалы, символы и имена при вводе кода.

[9. Упражнения](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#8)

1. Можете ли вы прочитать код этой программы и угадать, каков будет результат (не запуская ее в Kotlin Playground)?

fun main() {  
    println("1")  
    println("2")  
    println("3")  
}

Если у вас есть предположение, скопируйте и вставьте этот код в Kotlin Playground, чтобы проверить свой ответ.

1. Используйте Kotlin Playground для создания программы, которая выводит следующие сообщения:

I'm

learning

Kotlin!

1. Скопируйте и вставьте эту программу в Kotlin Playground.

fun main() {  
    println("Tuesday")  
    println("Thursday")  
    println("Wednesday")  
    println("Friday")  
    println("Monday")  
}

Исправьте программу, чтобы она печатала этот вывод:

Monday

Tuesday

Wednesday

Thursday

Friday

Для начала практики устранения неполадок исправьте ошибки в следующих упражнениях. Для каждого упражнения скопируйте код в Kotlin Playground в своем браузере. Попробуйте запустить программу, и вы увидите сообщение об ошибке.

1. Исправьте ошибку в этой программе, чтобы она выдавала желаемый результат.

fun main() {  
    println("Tomorrow is rainy")

Желаемый результат:

Tomorrow is rainy

1. Исправьте ошибку в этой программе, чтобы она выдавала желаемый результат.

fun main() {  
    printLine("There is a chance of snow")  
}

Желаемый результат:

There is a chance of snow

1. Исправьте ошибку в этой программе, чтобы она выдавала желаемый результат.

fun main() {  
    println("Cloudy") println("Partly Cloudy") println("Windy")  
}

Желаемый результат:

Cloudy

Partly Cloudy

Windy

1. Исправьте ошибку в этой программе, чтобы она выдавала желаемый результат.

fun main() (  
    println("How's the weather today?")  
)

Желаемый результат:

How's the weather today?

После выполнения этих упражнений сверьте свои ответы с решениями в следующем разделе.

Правильные ответы <https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#9>

## [**11. Заключение**](https://developer.android.com/codelabs/basic-android-kotlin-compose-first-program?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-first-program#10)

Вы работали с простыми программами на Kotlin и запускали их, чтобы увидеть текст, выводимый на выходе. Вы модифицировали программы по-разному и наблюдали, как эти изменения повлияли на результат. Ошибки при программировании — это нормально, поэтому вы также начали учиться находить и исправлять ошибки в коде, а это важный навык, который поможет вам в будущем.

## **Краткое содержание**

* Программа Kotlin требует  функции main в качестве точки входа в программу.
* Чтобы определить функцию в Kotlin, используйте  ключевое слово fun, за которым следует имя функции, все входные данные, заключенные в круглые скобки, а затем тело функции, заключенное в фигурные скобки.
* Имя функции должно соответствовать стилю camal Case и начинаться со строчной буквы.
* Используйте вызов функции println(), чтобы напечатать текст на выходе.
* Устранение неполадок — это процесс устранения ошибок в вашем коде.

<https://developer.android.com/codelabs/basic-android-kotlin-compose-variables?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-1%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-variables#0>

<https://developer.android.com/courses/pathways/android-basics-compose-unit-1-pathway-2>

# **Установка Android Studio**

Android Studio — это официальная интегрированная среда разработки (IDE) для разработки приложений Android, созданная и распространяемая Google. IDE содержит инструменты, которые позволяют разработчикам программного обеспечения проектировать, создавать, запускать и тестировать программное обеспечение, в данном случае приложения для платформы Android. Android Studio использует IntelliJ IDEA в качестве основы и включает в себя предустановленный плагин Android, а также некоторые модификации, специально предназначенные для платформы Android.

## [2. Windows: проверьте системные требования.](https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-2%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-install-android-studio#1)

## Системные требования Android Studio

Ниже приведены системные требования для Android Studio в Windows.

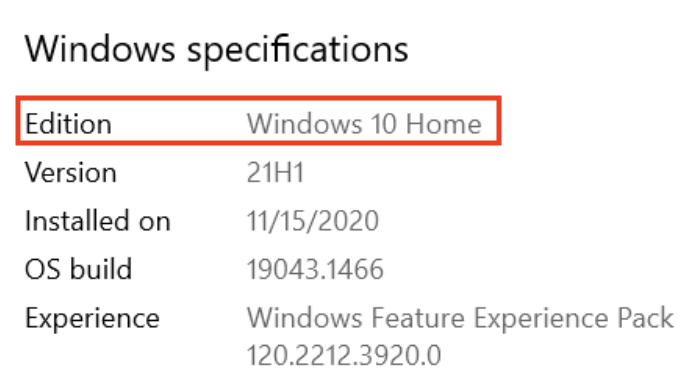
* 64-разрядная версия Microsoft® Windows® 10.08.11
* архитектура процессора x86\_64; Процессор Intel Core 2-го поколения или новее или процессор AMD с поддержкой гипервизора [Windows](https://developer.android.com/studio/run/emulator-acceleration#vm-windows)
* 8 ГБ ОЗУ или больше
* Минимум 8 ГБ доступного дискового пространства (IDE + Android SDK + эмулятор Android)
* Минимальное разрешение экрана 1280 x 800

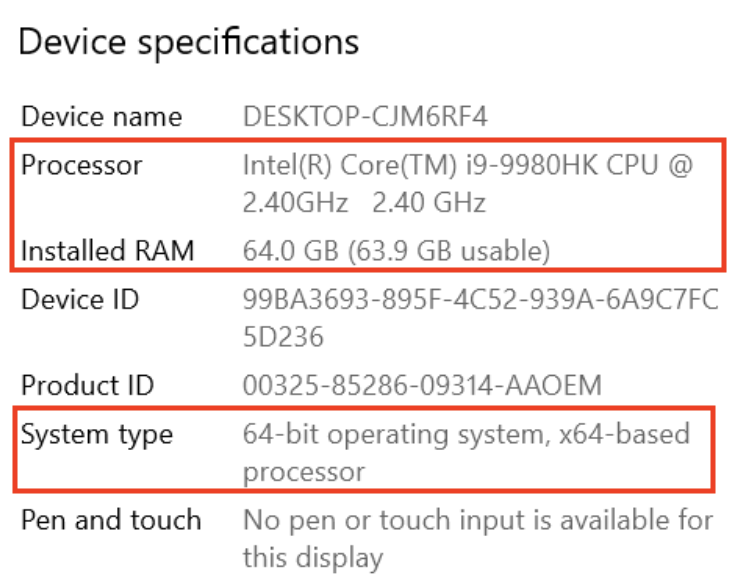
## Проверьте системные требования (Windows 10)

На компьютере с Windows вы можете найти всю информацию, необходимую для проверки системных требований, в приложении «Настройки».

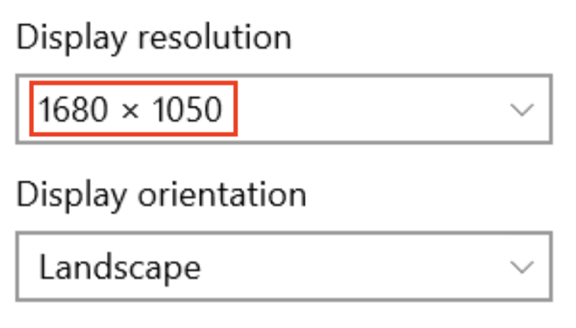
1. Открыть **настройки** .

**Совет:** Чтобы найти его, вы можете использовать инструмент поиска рядом с кнопкой «Пуск» в нижней части экрана.

1. Нажмите **Система** .
2. В нижней части левой панели навигации нажмите **«О программе»** .
3. Убедитесь, что **спецификации Windows** соответствуют требованиям или превосходят их.
4. Выберите **Характеристики устройства** . Убедитесь, что установленной оперативной памяти по крайней мере столько, сколько требуется, и что типом системы является 64-разрядная версия операционной системы.

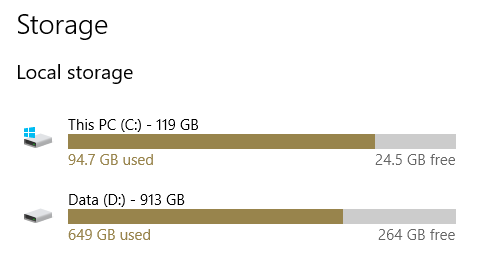


1. В области навигации нажмите **«Показать»** . Убедитесь, что **разрешение** такое же или лучше требуемого.



## Подтвердить хранилище

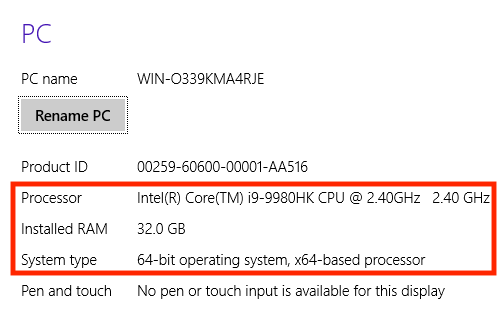
1. Откройте файловый менеджер.
2. На левой панели навигации нажмите **«Этот компьютер»** .
3. Убедитесь, что в локальном хранилище достаточно свободного места для установки Android Studio.

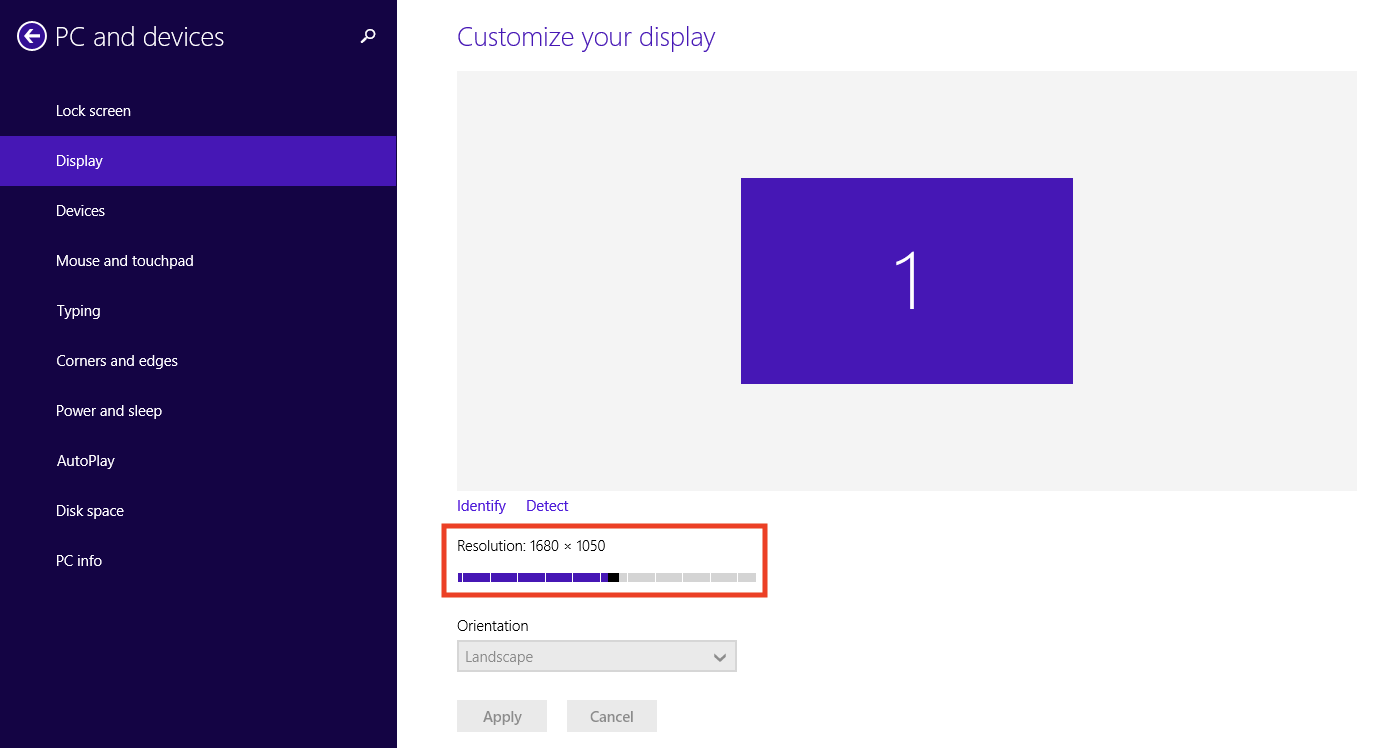
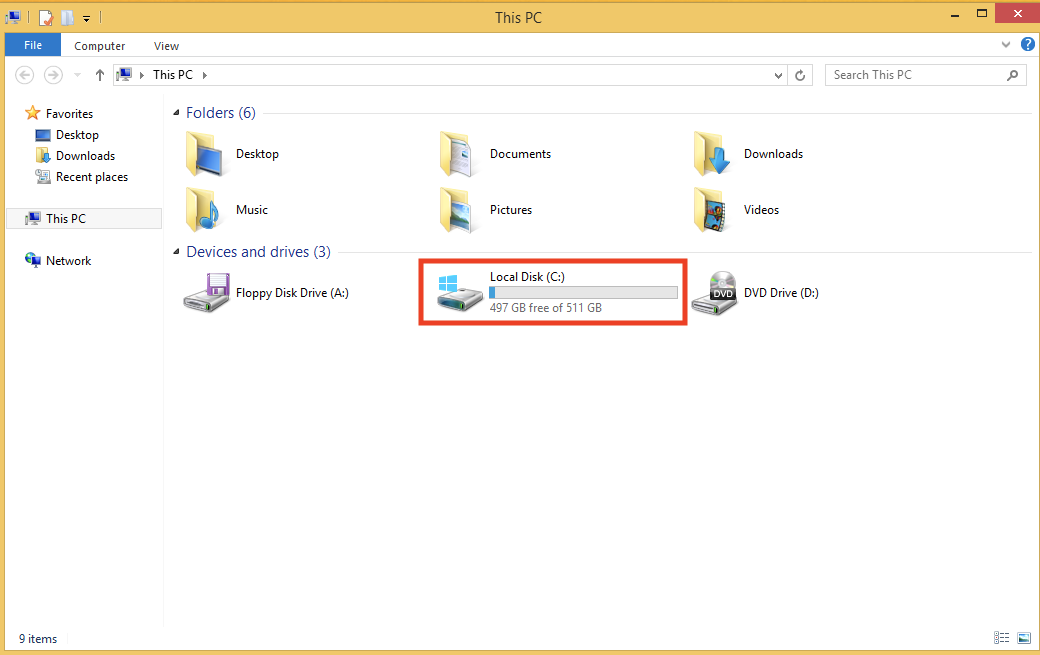


## Проверьте системные требования (Windows 8.1 )

Если вы используете Windows 8.1, шаги по поиску характеристик устройства следующие:

1. Откройте приложение «Настройки».
2. В области навигации щелкните **ПК и устройства** .
3. На панели навигации нажмите **«Информация о компьютере»** . Убедитесь, что ваш процессор и оперативная память соответствуют минимальным требованиям, и убедитесь, что вы используете 64-разрядную операционную систему.



1. В области навигации нажмите **«Показать»** . Убедитесь, что ваш дисплей соответствует требованиям.
2. Откройте проводник, нажмите **«Этот компьютер** » и убедитесь, что у вас достаточно места на диске.

## [3. Windows: загрузите и установите Android Studio.](https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-2%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-install-android-studio#2)

## Загрузите Android-студию

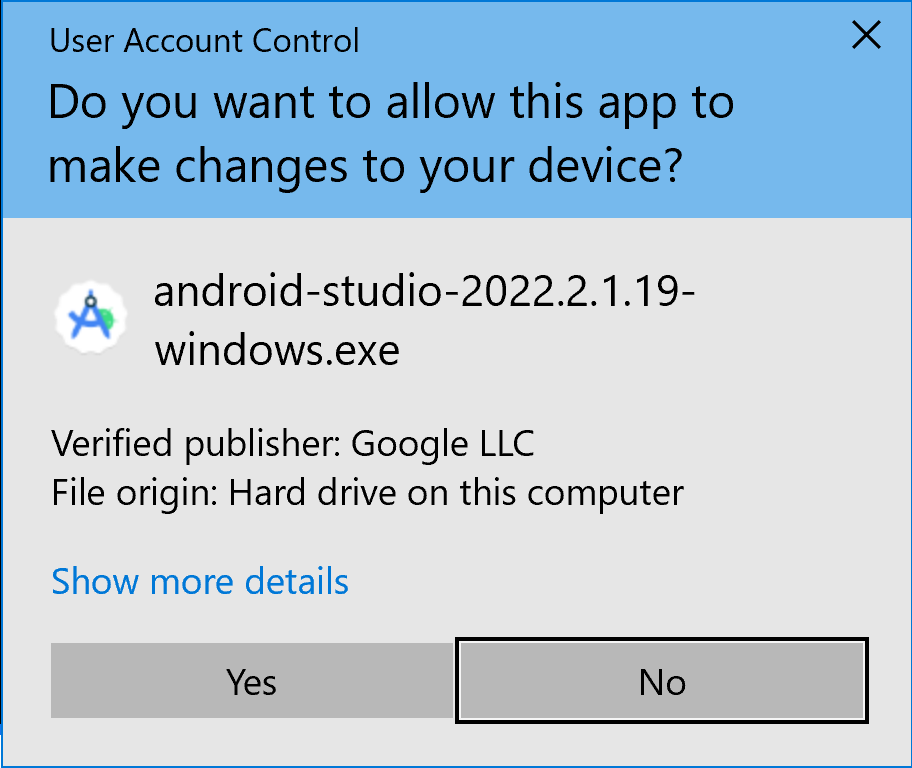
1. Откройте любой веб-браузер и перейдите на [страницу загрузки Android Studio](https://developer.android.com/studio#get-android-studio) .

Это веб-сайт разработчиков Android, где вы можете скачать Android Studio. Эта страница автоматически определяет вашу операционную систему.

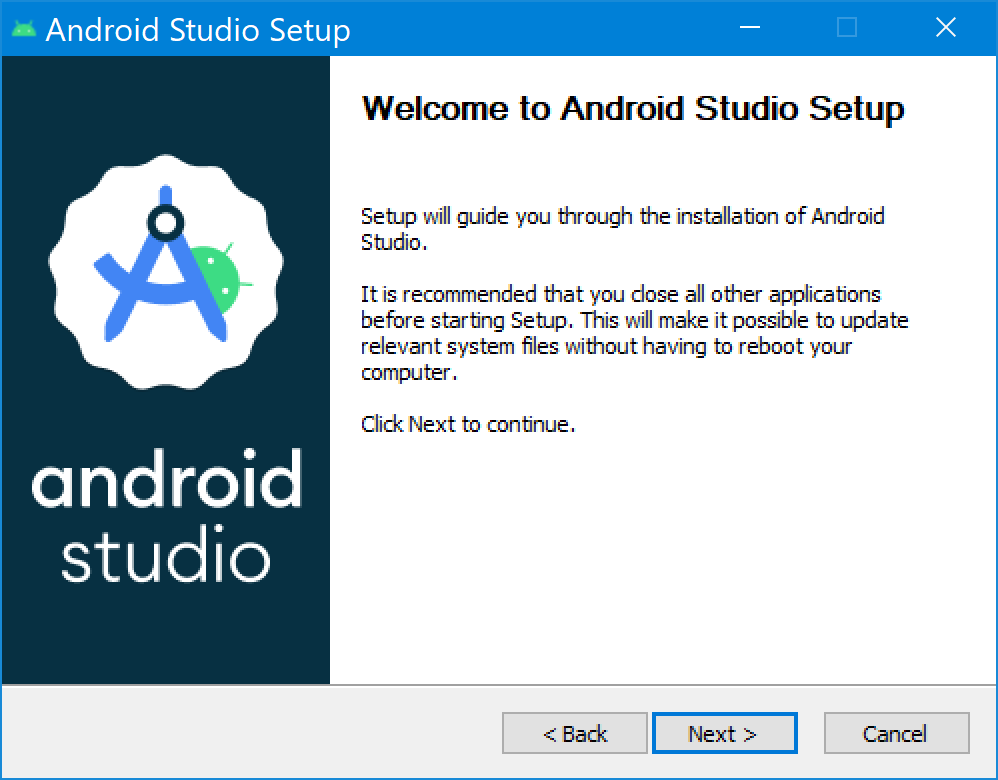
1. Нажмите **«Загрузить Android Studio** ». Откроется страница « Правила **и условия» с лицензионным соглашением** Android Studio .
2. Прочтите **Лицензионное соглашение** .
3. В нижней части страницы, если вы согласны с условиями, установите флажок **Я прочитал и согласен с вышеуказанными условиями** .
4. Нажмите **«Загрузить Android Studio»** , чтобы начать загрузку.
5. При появлении запроса сохраните файл в месте, где его можно будет легко найти, например в Downloadsпапке.
6. Подождите завершения загрузки. Это может занять некоторое время, но, возможно, сейчас самое время выпить чашечку чая!

## Установите Android Studio в Windows

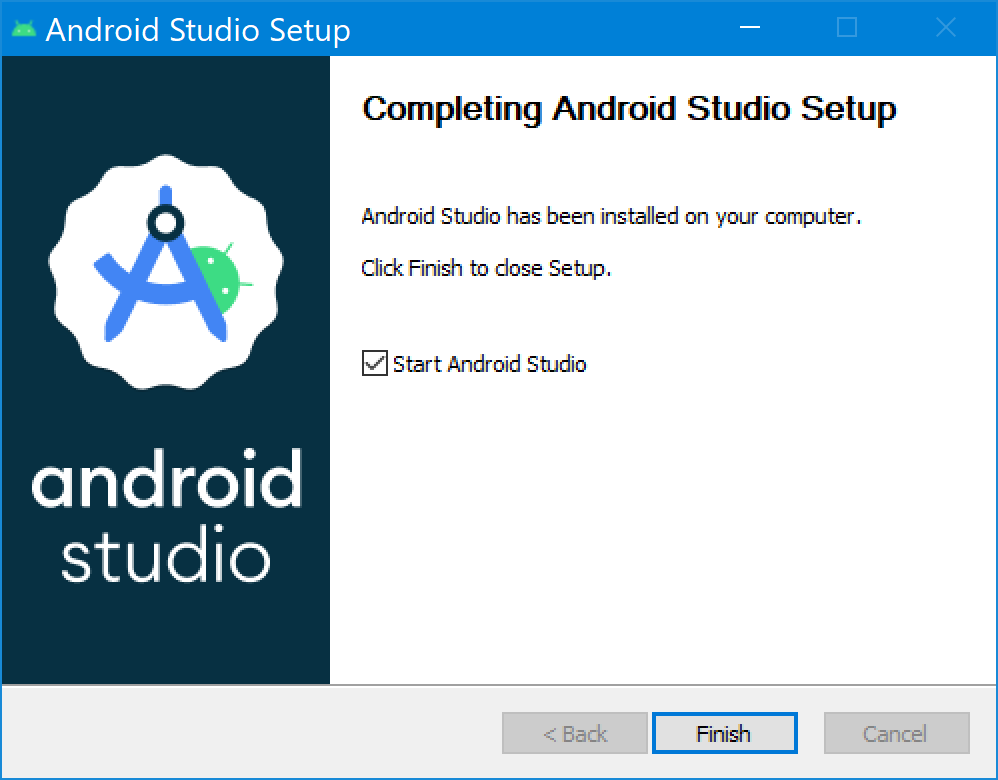
1. Откройте папку, в которую вы скачали и сохранили установочный файл Android Studio.
2. Дважды щелкните загруженный файл.
3. Если вы увидите диалоговое окно **«Контроль учетных записей»** о разрешении установки внести изменения в ваш компьютер, нажмите « **Да»** , чтобы подтвердить установку.



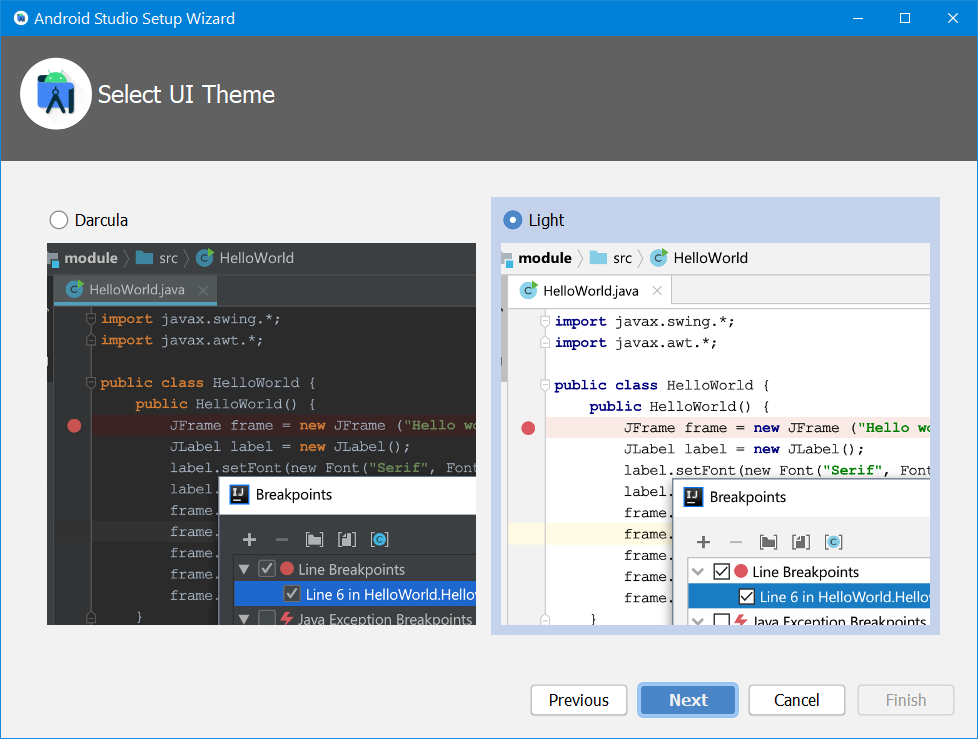
Откроется диалоговое окно **«Добро пожаловать в установку Android Studio»** .



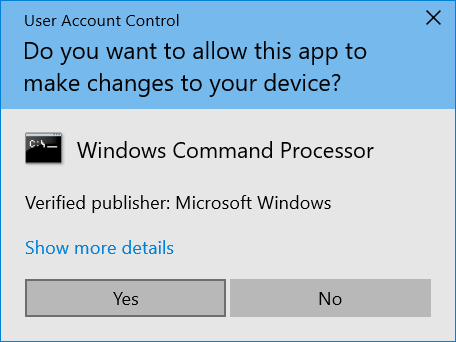
1. Нажмите **Далее** , чтобы начать установку.
2. Примите настройки установки по умолчанию для всех шагов.
3. Нажмите **«Готово»** , когда установка завершится, чтобы запустить Android Studio.



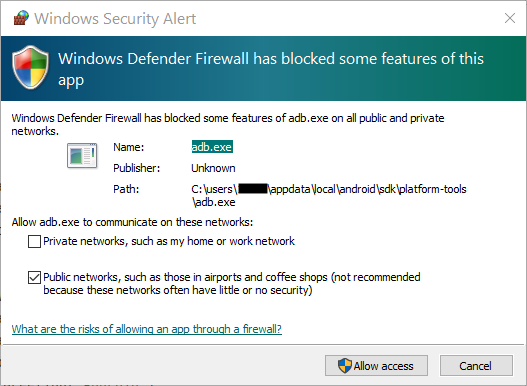
1. Выберите светлую или темную тему при первом запуске Android Studio. Скриншоты в этом курсе используют светлую тему, но выберите ту, которая вам больше нравится.



1. Во время установки мастер установки загружает и устанавливает дополнительные компоненты и инструменты, необходимые для разработки приложений для Android. Это может занять некоторое время в зависимости от скорости вашего интернета. В это время вы можете увидеть диалоговое окно **«Контроль учетных записей пользователей»** для **командного процессора Windows** . Нажмите **Да** , чтобы принять диалоговое окно.

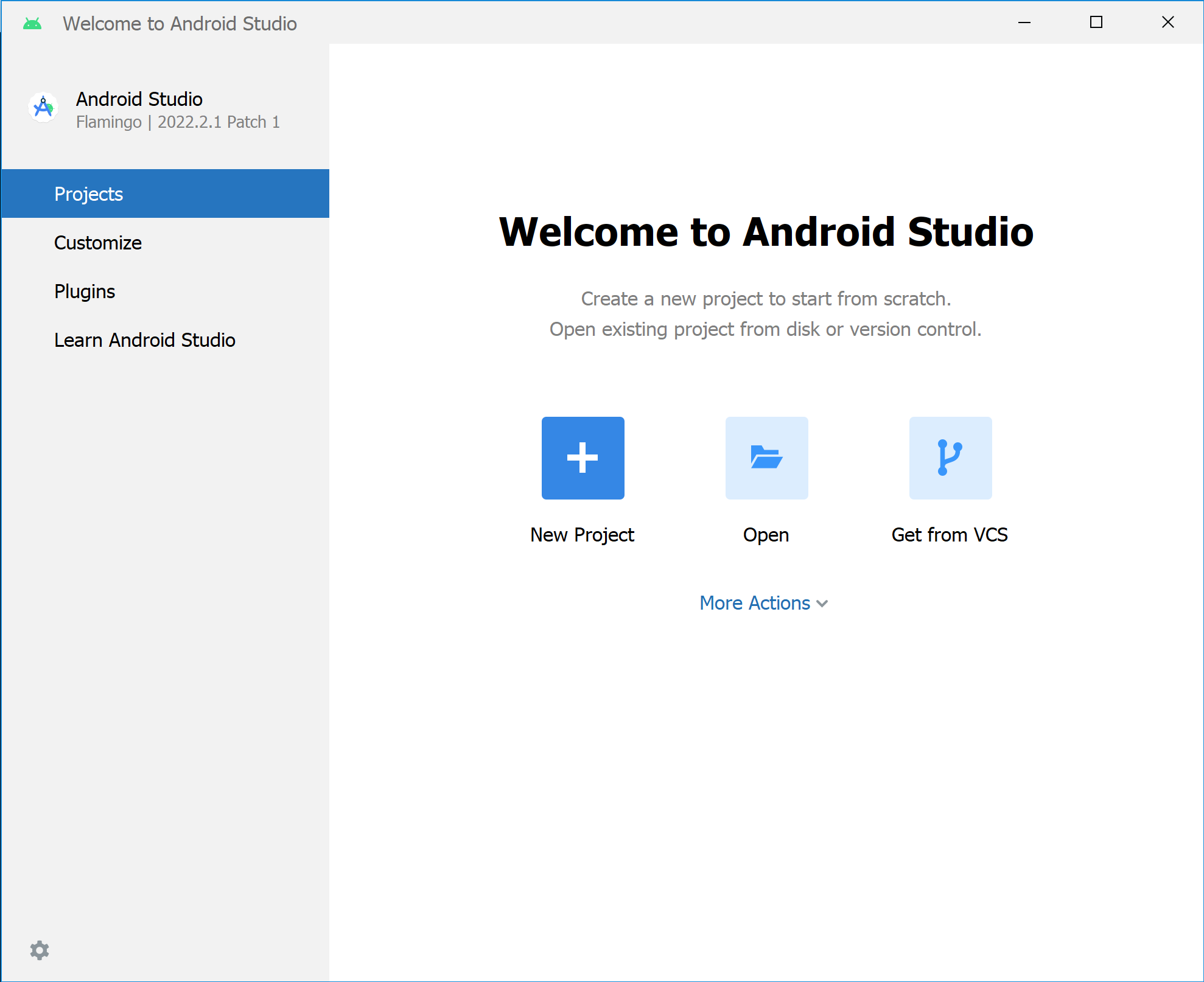


1. Вы также можете получить **предупреждение системы безопасности Windows** об adb.exe. При необходимости нажмите **«Разрешить доступ»** , чтобы продолжить установку.



1. Когда загрузка и установка завершится, нажмите **«Готово»** .

Откроется окно « Добро **пожаловать в Android Studio»,** и вы готовы приступить к созданию приложений!



## [4. macOS: проверьте системные требования.](https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-2%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-install-android-studio" \l "3)

## **Если установили на Windows, то этот раздел пропустить**

## Системные требования Android Studio (macOS)

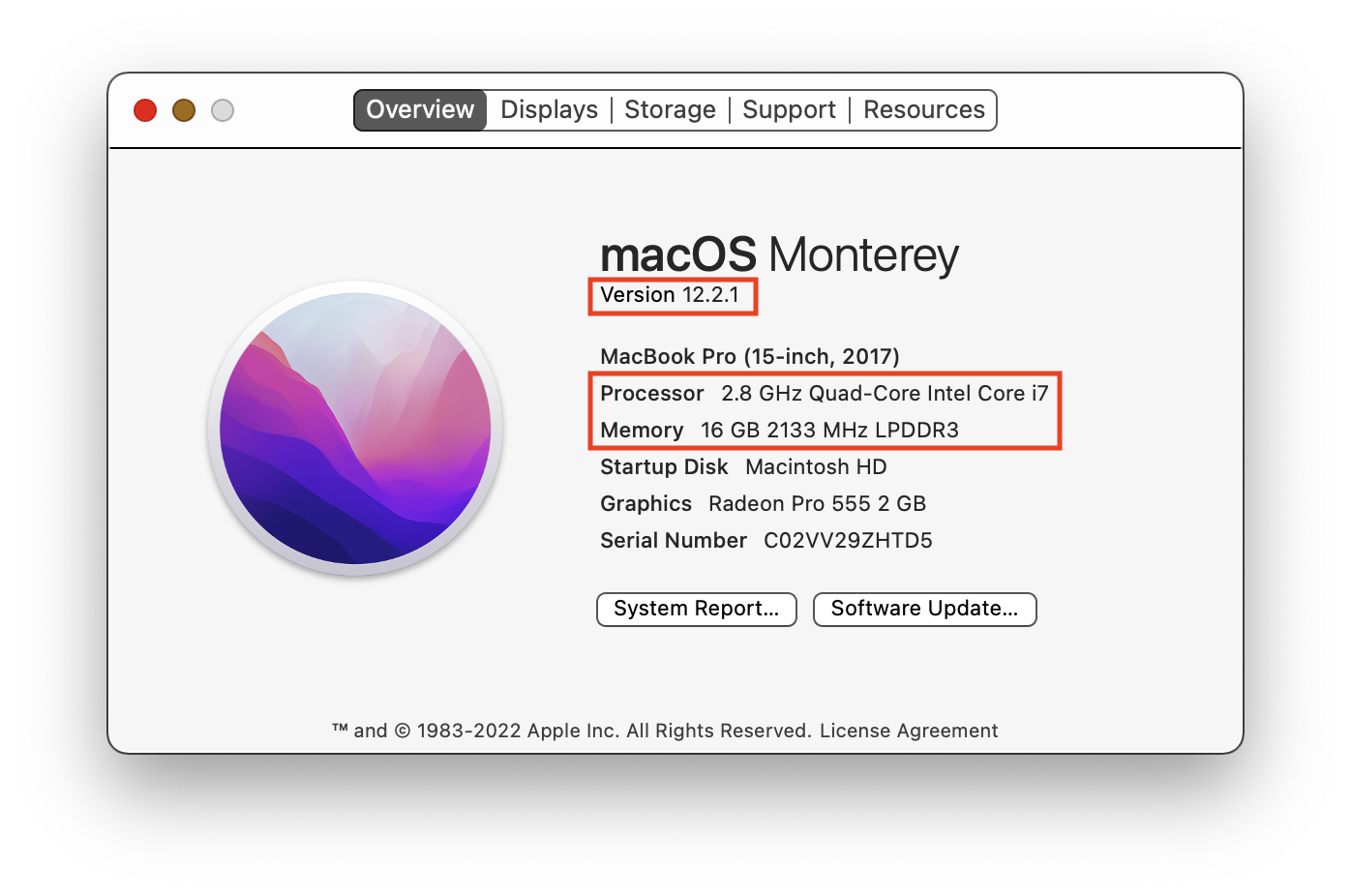
Ниже приведены системные требования для Android Studio на macOS.

* MacOS® 10.14 (Мохаве) или выше
* Чипы на базе ARM или Intel Core 2-го поколения или новее с поддержкой [Hypervisor Framework](https://developer.android.com/studio/run/emulator-acceleration" \l "vm-mac" \t "_blank)
* 8 ГБ ОЗУ или больше
* Минимум 8 ГБ доступного дискового пространства (IDE + Android SDK + эмулятор Android)
* Минимальное разрешение экрана 1280 x 800

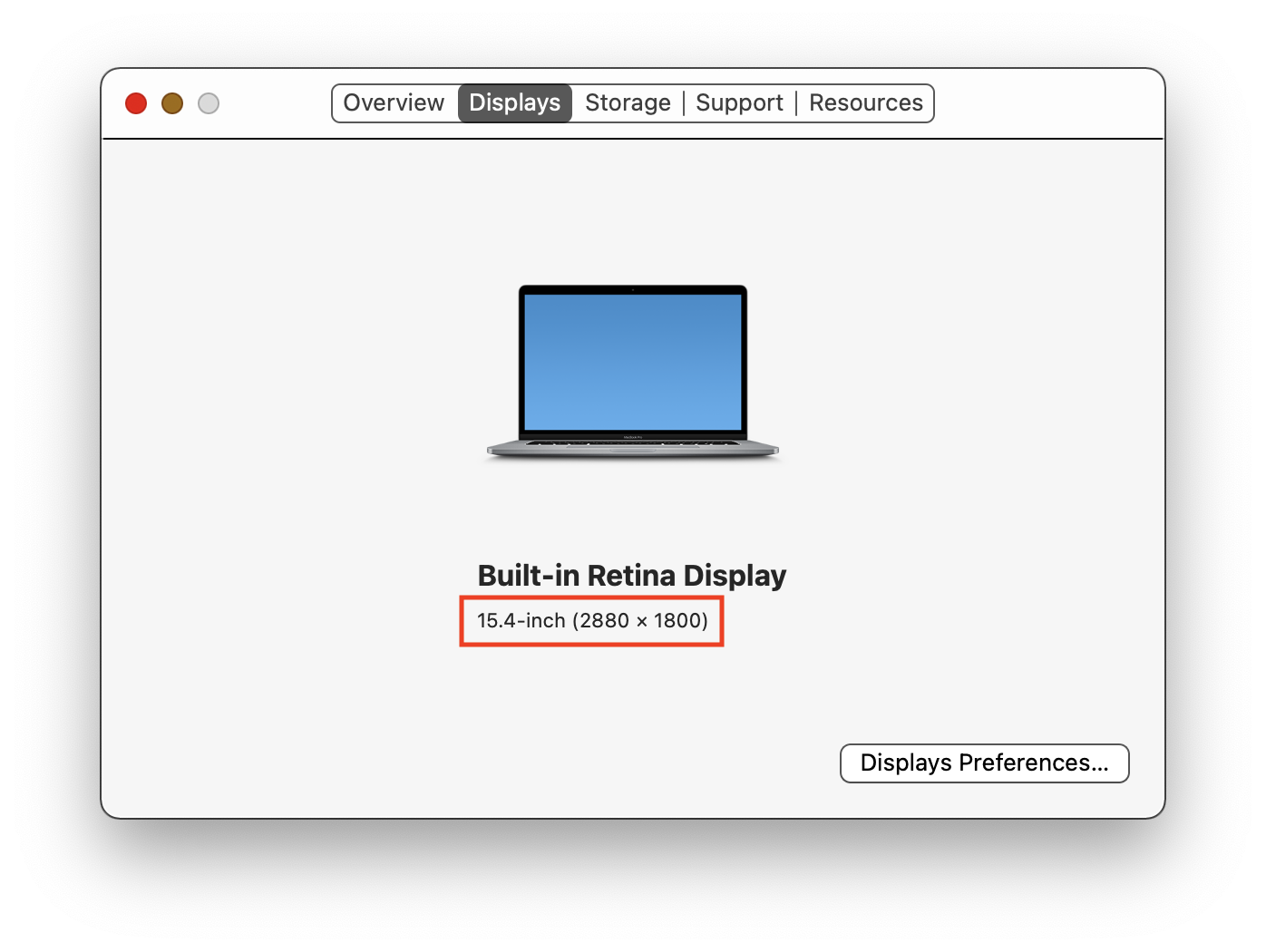
## Проверьте системные требования

1. Выберите **> Об этом Mac** .
2. В диалоговом окне на вкладке **«Обзор»** найдите номер версии ОС и убедитесь, что он находится в требуемом диапазоне.
3. На вкладке **«Память»** убедитесь, что указанный общий объем памяти соответствует или превышает требуемый минимум.

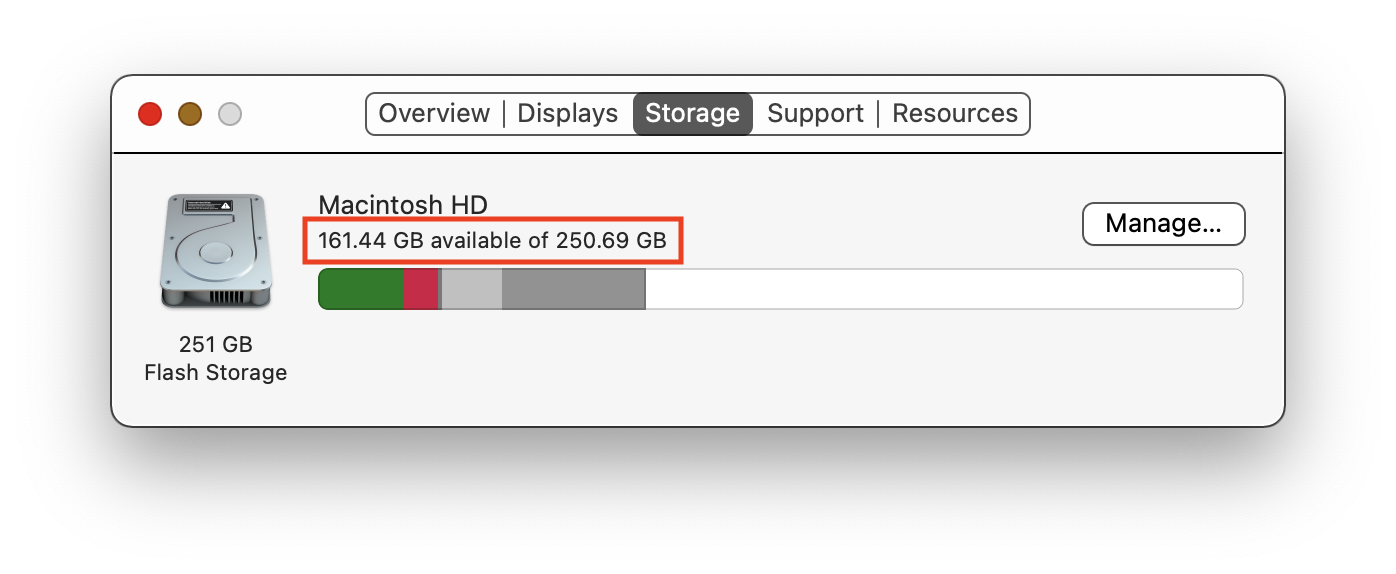
Например, на скриншоте ниже версия ОС — 12.2.1, а памяти — 16 ГБ.



1. В том же диалоговом окне перейдите на вкладку **«Дисплеи»** .
2. В описании дисплея убедитесь, что разрешение экрана вашего компьютера соответствует рекомендуемому разрешению или превышает его.



1. Откройте вкладку **«Хранилище»** .
2. Проверьте доступное дисковое пространство и убедитесь, что оно соответствует или превышает необходимое дисковое пространство для установки и запуска Android Studio.



## [5. macOS: загрузите и установите Android Studio.](https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio?continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-1-pathway-2%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-install-android-studio#4)

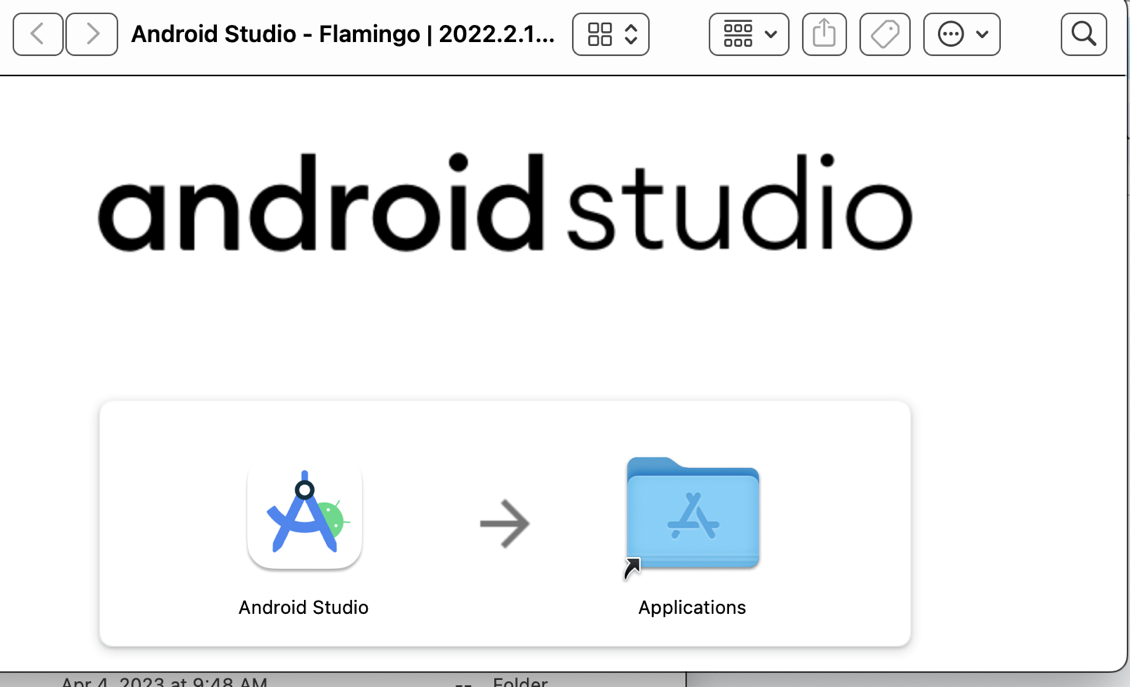
## Загрузите Android-студию

1. Откройте любой веб-браузер и перейдите на [страницу загрузки Android Studio](https://developer.android.com/studio/?gclid=Cj0KCQiAjJOQBhCkARIsAEKMtO3zEhdK4_I0CEZic3UH4dl-9gVXuHFR9dCl3TOHKjmv3xWLU3UxfhYaApfAEALw_wcB&gclsrc=aw.ds) . Это веб-сайт разработчиков Android, где вы можете скачать Android Studio. Эта страница автоматически определяет вашу операционную систему.
2. Нажмите **«Загрузить Android Studio** ». Откроется страница « Правила **и условия» с лицензионным соглашением** Android Studio .
3. Прочтите **Лицензионное соглашение** .
4. В нижней части страницы, если вы согласны с условиями, установите флажок **Я прочитал и согласен с вышеуказанными условиями** .
5. Нажмите **Mac с чипом Intel** или **Mac с чипом Apple** , чтобы начать загрузку.
6. При появлении запроса сохраните файл в месте, где его можно будет легко найти, например в Downloadsпапке.
7. Подождите завершения загрузки. Это может занять некоторое время, но, возможно, сейчас самое время выпить чашечку чая!

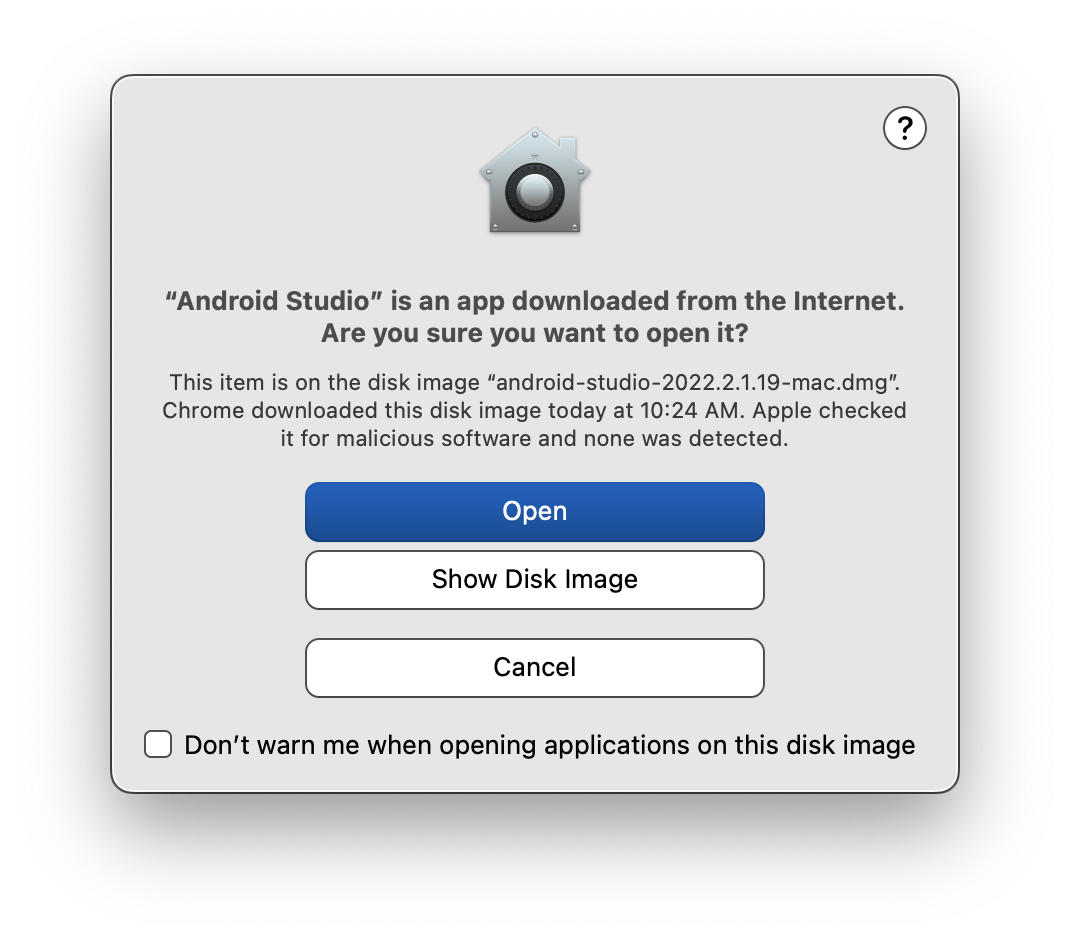
**Примечание.** Если вам нужна дополнительная помощь или вы хотите настроить установку, обратитесь к инструкциям по [установке Android Studio](https://developer.android.com/studio/install.html) , которые включают в себя скринкасты.

## Установите Android Studio на macOS

1. Откройте папку, в которую вы скачали и сохранили установочный файл Android Studio.
2. Дважды щелкните загруженный файл. Появится следующее диалоговое окно:

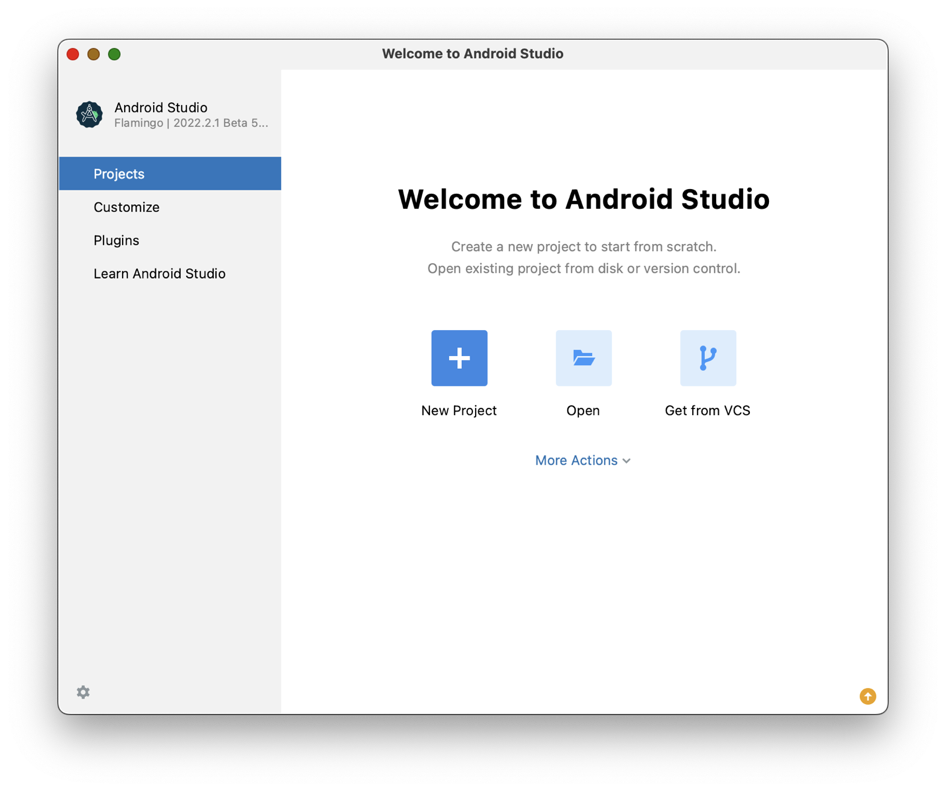


1. Перетащите значок **Android Studio** в Applicationsпапку.
2. В Applicationsпапке дважды щелкните значок **Android Studio** , чтобы запустить **мастер установки Android Studio** .
3. Если вы видите предупреждение об установке или запуске файла, скачанного из Интернета, согласитесь на установку.



1. Следуйте **инструкциям мастера установки Android Studio** и примите настройки по умолчанию для всех шагов. Во время установки мастер установки загружает и устанавливает дополнительные компоненты и инструменты, необходимые для разработки приложений для Android. Это может занять некоторое время, в зависимости от скорости вашего интернета. Итак, вы можете пойти наполнить чашку чая!
2. Когда установка завершится, Android Studio запустится автоматически.

Откроется окно « Добро **пожаловать в Android Studio»,** и вы готовы приступить к созданию приложений!



**Устранение неполадок Android Studio**

https://developer.android.com/studio/troubleshoot