

Документация

Учебники Java™

Trail: Getting Started

Lesson: The "Hello World!" Application

Учебники Java были написаны для JDK 8. Примеры и практики, описанные на этой странице, не используют преимущества улучшений, представленных в более поздних выпусках, и могут использовать технологии, которые больше не доступны.

См. [Dev.java](#) для обновленных учебных пособий, использующих преимущества последних выпусков.

См. [Java Language Changes](#) для обзора обновленных языковых функций в Java SE 9 и последующих выпусках.

См. [JDK Release Notes](#) для информации о новых функциях, улучшениях и удаленных или устаревших опциях для всех выпусков JDK.

«Hello World!» для ОС Solaris, Linux и Mac OS X

Пришло время написать свое первое приложение! Эти подробные инструкции предназначены для пользователей Solaris OS, Linux и Mac OS X. Инструкции для других платформ находятся в ["Hello World!" для Microsoft Windows](#) и ["Hello World!" для NetBeans IDE](#).

Если у вас возникли проблемы с инструкциями на этой странице, обратитесь к разделу [Распространенные проблемы \(и их решения\)](#).

- [Контрольный список](#)
- [Создание вашего первого приложения](#)
 - [Создать исходный файл](#)
 - [Скомпилировать исходный файл в .class-файл](#)
 - [Запустить программу](#)



Контрольный список

Чтобы написать свою первую программу, вам понадобится:

1. Комплект разработчика Java SE 8 (JDK 8)

Вы можете [загрузить версию для Solaris OS, Linux или Mac OS X](#). (Убедитесь, что вы загружаете **JDK**, а не JRE.) Ознакомьтесь с [инструкциями по установке](#).

2. Текстовый редактор

В этом примере мы будем использовать Pico, редактор, доступный для многих платформ на базе UNIX. Вы можете легко адаптировать эти инструкции, если используете другой текстовый редактор, например vi или emacs.

Эти два пункта — все, что вам понадобится для написания вашего первого заявления.

Создание вашего первого приложения

Ваше первое приложение, HelloWorldApp, просто отобразит приветствие "Hello world!". Чтобы создать эту программу, вам нужно:

• Создать исходный файл

Исходный файл содержит код, написанный на языке программирования Java, который вы и другие программисты можете понять. Вы можете использовать любой текстовый редактор для создания и редактирования исходных файлов.

• Скомпилируйте исходный файл в файл .class.

Компилятор языка программирования Java (javac) берет ваш исходный файл и переводит его текст в инструкции, которые может понять виртуальная машина Java. Инструкции, содержащиеся в этом .class-файле, называются *байт-кодами*.

• Запустить программу

Инструмент запуска приложений Java (java) использует виртуальную машину Java для запуска вашего приложения.

Создать исходный файл

Чтобы создать исходный файл, у вас есть два варианта:

[Настройки файлов cookie](#) | [Выбор рекламы](#)

- Вы можете сохранить файл на своем компьютере и избежать многого ввода. Затем вы можете сразу перейти к [Компиляции исходного файла . HelloWorldApp.java](#)
- Или вы можете воспользоваться следующими (более подробными) инструкциями.

Сначала откройте окно оболочки или «терминала».



Новое окно терминала.

Когда вы впервые вызываете приглашение, вашим *текущим каталогом* обычно будет ваш *домашний каталог* . Вы можете изменить свой текущий каталог на домашний каталог в любое время, введя `cd` в приглашении и нажав **Return** .

Исходные файлы, которые вы создаете, должны храниться в отдельном каталоге. Вы можете создать каталог, используя команду `mkdir` . Например, чтобы создать каталог `examples/javav` в каталоге `/tmp` , используйте следующие команды:

```
компакт-диск /tmp
примеры mkdir
примеры компакт-дисков
mkdir java
```

Чтобы изменить текущий каталог на этот новый, введите:

```
cd /tmp/примеры/java
```

Теперь вы можете приступить к созданию исходного файла.

Запустите редактор Pico, введя текст `pico` в строке и нажав **Return** . Если система ответит сообщением `pico: command not found`, то Pico, скорее всего, недоступен. Обратитесь к системному администратору за дополнительной информацией или используйте другой редактор.

При запуске Pico отобразится новый пустой *буфер* . Это область, в которой вы будете вводить свой код.

Введите следующий код в новый буфер:

```
/**
 * Класс HelloWorldApp реализует приложение, которое
 * просто выводит «Hello World!» на стандартный вывод.
 */
класс HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Отображаем строку.
    }
}
```

Будьте осторожны при наборе текста



Примечание: Введите весь код, команды и имена файлов точно так, как показано. Как компилятор (`javac`), так и средство запуска (`java`) чувствительны к регистру , поэтому вы должны использовать заглавные буквы последовательно.

[Настройки файлов cookie](#) | [Выбор рекламы](#)

HelloWorldAppне *то* же самое, что helloworldapp.

Сохраните код в файле с именем HelloWorldApp.java. В редакторе Pico это можно сделать, набрав **Ctrl-O**, а затем, внизу, где вы видите приглашение `File Name to write:`, введите каталог, в котором вы хотите создать файл, а затем HelloWorldApp.java. Например, если вы хотите сохранить HelloWorldApp.java в каталоге /tmp/examples/java, то вы вводите /tmp/examples/java/HelloWorldApp.java и нажимаете **Return**.

Для выхода из Pico можно нажать **Ctrl-X**.

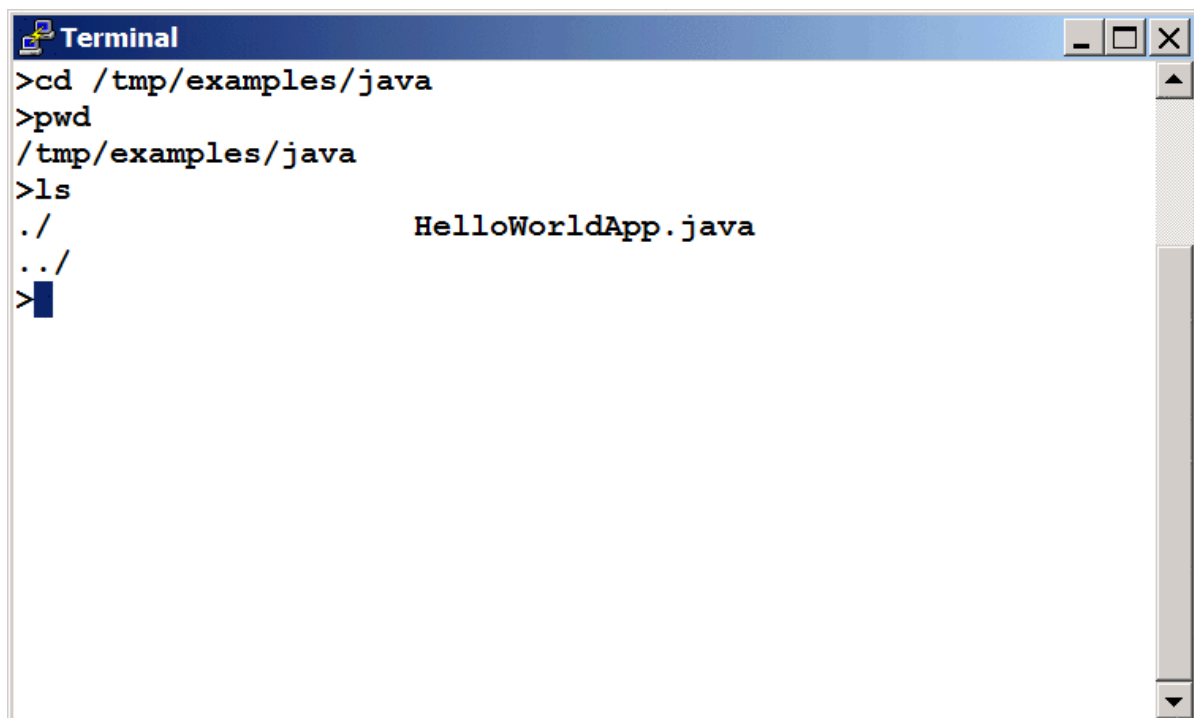
Скомпилировать исходный файл в .class-файл

Откройте еще одно окно оболочки. Чтобы скомпилировать исходный файл, измените текущий каталог на каталог, в котором находится ваш файл. Например, если исходный каталог — /tmp/examples/java, введите следующую команду в командной строке и нажмите **Return**:

```
cd /tmp/примеры/java
```

Если вы введете команду pwd в командной строке, вы должны увидеть текущий каталог, который в этом примере был изменен на /tmp/examples/java.

Если вы введете команду ls в командной строке, вы должны увидеть свой файл.



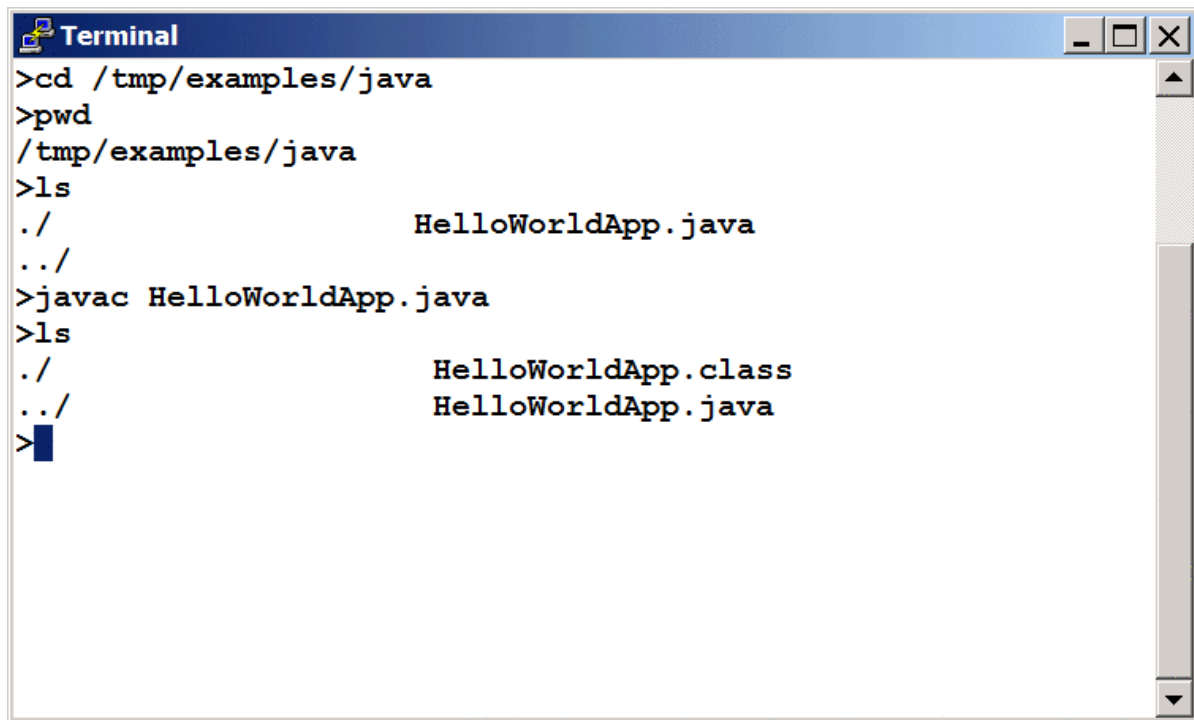
```
Terminal
>cd /tmp/examples/java
>pwd
/tmp/examples/java
>ls
./                               HelloWorldApp.java
../
>
```

Результаты команды ls, показывающие .java-исходный файл.

Теперь все готово для компиляции исходного файла. В командной строке введите следующую команду и нажмите **Return**.

```
javacПриветМирПриложение.java
```

Компилятор сгенерировал файл байт-кода, HelloWorldApp.class. В командной строке введите , ls чтобы увидеть новый сгенерированный файл: следующий рисунок .



```
Terminal
>cd /tmp/examples/java
>pwd
/tmp/examples/java
>ls
./                               HelloWorldApp.java
../
>javac HelloWorldApp.java
>ls
./                               HelloWorldApp.class
../                               HelloWorldApp.java
>
```

Результаты выполнения `ls`команды, показывающие сгенерированный `.class`файл.

Теперь, когда у вас есть `.class`файл, вы можете запустить свою программу.

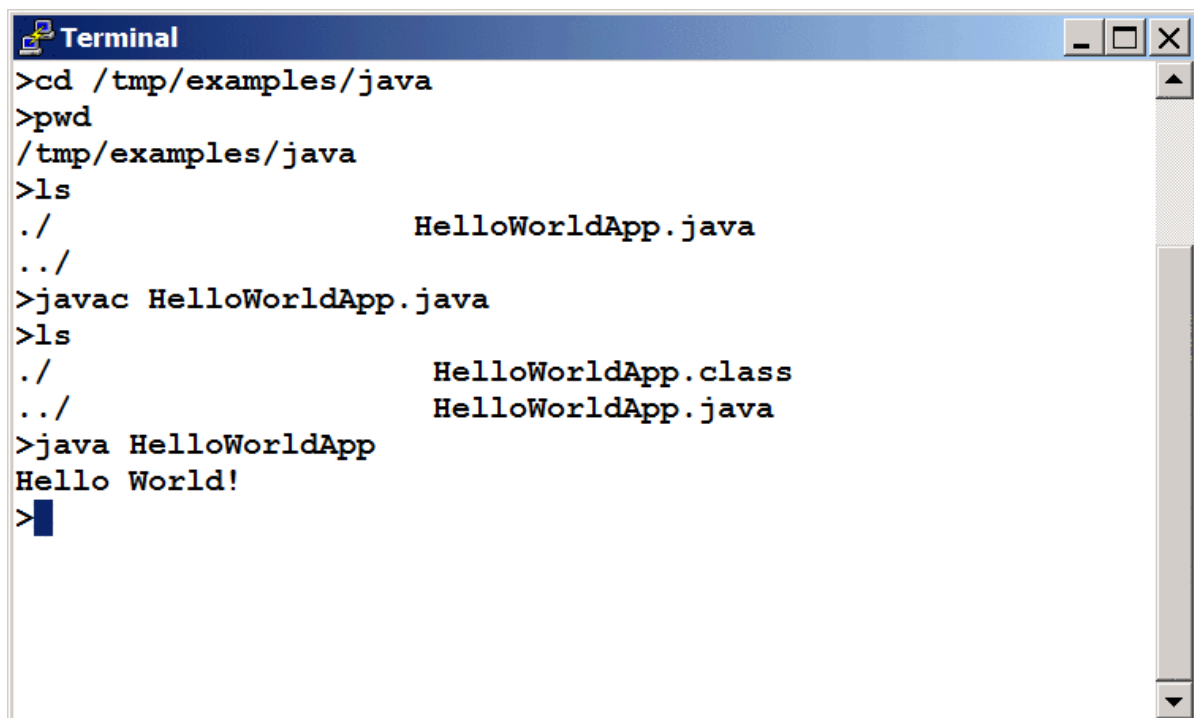
Если у вас возникли проблемы с инструкциями на этом этапе, обратитесь к разделу [Распространенные проблемы \(и их решения\)](#).

Запустить программу

В том же каталоге введите в командной строке:

```
javaПриложение HelloWorld
```

На следующем рисунке показано то, что вы сейчас увидите.



```
Terminal
>cd /tmp/examples/java
>pwd
/tmp/examples/java
>ls
./                               HelloWorldApp.java
../
>javac HelloWorldApp.java
>ls
./                               HelloWorldApp.class
../                               HelloWorldApp.java
>java HelloWorldApp
Hello World!
>
```

В результате на экране появится надпись «Hello World!».

Поздравляем! Ваша программа работает!

Если у вас возникли проблемы с инструкциями на этом этапе, обратитесь к разделу [Распространенные проблемы \(и их решения\)](#).