JavaDecaf User Manual

1 Introduction

JavaDecaf is a simplified version of the Java programming language. It uses Java syntax, but without some of the more complicated features that can be confusing for beginners.

JavaDecaf features simplified I/O and supports the following methods:

- `print` - print a `String` to the command line
- `println` - print a `String` to the command line on a new line
- `readLine` - read a `String` from the command line
- `readInt` - read an `int` from the command line

2 Setting up JavaDecaf

Before you can use JavaDecaf, you should have the Java JDK (Java Development Kit) installed. The latest version is Java SE 8, available from the Oracle website. To check whether the JDK is installed, open a command prompt[1] and type `javac`. If you get an error or a popup, it is either not installed or not set up properly. If it is installed, it will give you a list of options.

Note to Windows users: Once you have installed Java, you need to tell your system where it is installed.

- Click Start and type "System Properties". Open it.
- Select the tab *Advanced system properties*, then click: *Environment variables*, *System variables*, *PATH.*
- Make a note of the version number of your Java installation (the last two digits in the folder name). Go to the beginning of the line and type the following:
  `C:\Program Files\Java\jdk1.8.0_XX\bin;`
  (XX being the version number, and the folder path being wherever you installed Java. Make sure to include the semicolon at the end.)

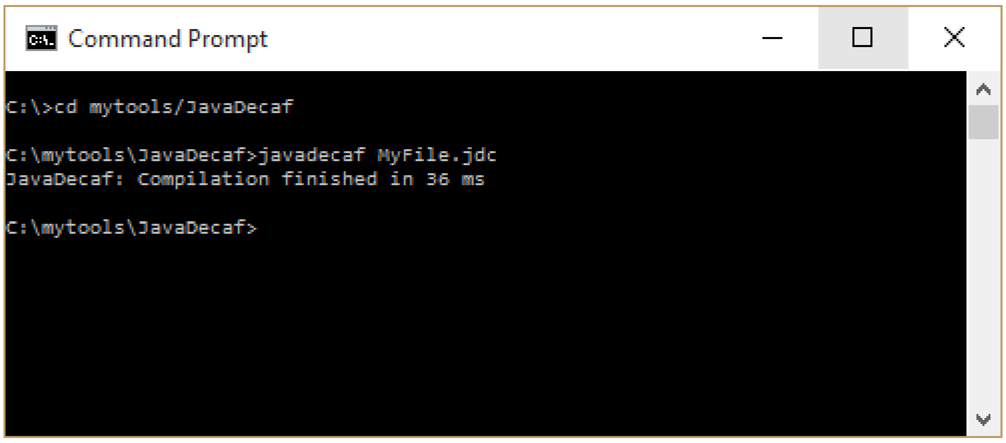
[1]On Windows: press Start and type cmd; on Mac/Linux: search for Terminal

- Click OK on all the windows.

3 Compiling & running your program

3.1 Windows

- Open up the command prompt - click the Start menu and type `cmd`.
- Navigate to the folder where you have downloaded JavaDecaf by typing `cd path\to\folder`, e.g. `cd H:\JavaDecaf`.
- Finally, type `javadecaf path\to\your\file`, e.g. `javadecaf H:\Programming in Java\MyFile.jdc`.
- NB. If your file is in the same folder as JavaDecaf, you only need to type the filename, not the full file path.



```
C:\>cd mytools/JavaDecaf

C:\mytools\JavaDecaf>javadecaf MyFile.jdc
JavaDecaf: Compilation finished in 36 ms

C:\mytools\JavaDecaf>
```

Figure 1: Using JavaDecaf on Windows

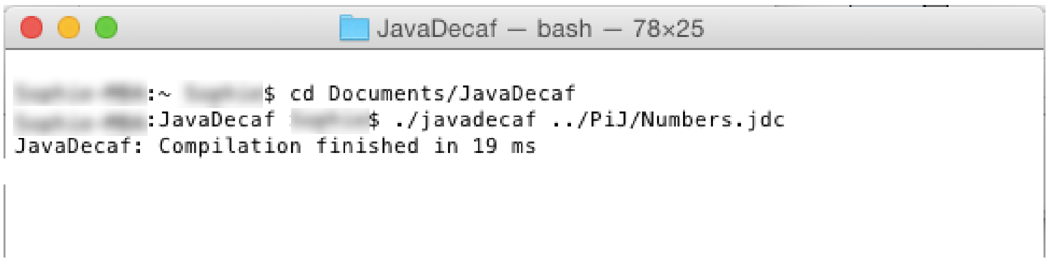
3.2 Mac OS X & Linux

- *OS X*: Open Terminal by clicking the magnifying glass on the menu bar (or pressing ⌘+ Space) and typing `terminal`.
- *Linux*: Press CTRL + Alt + T.
- Navigate to the folder where you have downloaded JavaDecaf by typing `cd path/to/your/folder`, e.g. `cd Documents/JavaDecaf`.
- Finally, type `./javadecaf path/to/your/file` to execute your file, e.g. `./javadecaf PiJ/MyFile.jdc`.
- NB. If your file is in the same folder as JavaDecaf, you only need to type the filename, not the full file path.

3.3 Running your program

If the compiler completes successfully, it will print a success message and output a file with the extension `.class`.

Once you've compiled your program you need to run it using the `java` command: `java MyFile` (where MyFile is the name of your file, with **no extension**).

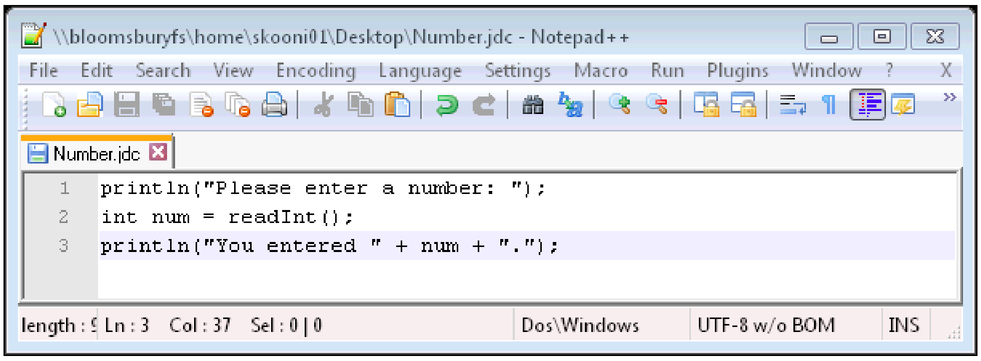


```
[redacted]:~ [redacted]$ cd Documents/JavaDecaf
[redacted]:JavaDecaf [redacted]$ ./javadecaf ../PiJ/Numbers.jdc
JavaDecaf: Compilation finished in 19 ms
```

Figure 2: Using JavaDecaf on OS X

4 Writing code

There is no need to define a `main` method in JavaDecaf: the compiler will do it for you. Simply write your code in a text editor such as Notepad and save it with the extension `jdc`. Your filename should begin with a capital letter and contain no spaces or special characters.


\\bloomsburyfs\home\skooni01\Desktop\Number.jdc - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
Number.jdc

```
println("Please enter a number: ");
int num = readInt();
println("You entered " + num + ".");
```

length : 9 Ln : 3 Col : 37 Sel : 0 | 0 Dos\Windows UTF-8 w/o BOM INS

Figure 3: JavaDecaf code in a text editor

5 Methods

You can define methods in JavaDecaf: an example is given in Figure 4. They must come **after** any main code you write.

```
int myNum = 5;
int squared = square(myNum);
println(squared);

int square(int number) {
    return number * number;
}
```

Figure 4: Example method declaration in JavaDecaf

6 Nested classes

JavaDecaf supports nested classes: an example is given in Figure 5. These must be declared **after** the main code and any methods.

```
Person person1 = new Person();
person1.talk();

class Person {
      void talk() {
          println("Hello!");
      }
}
```

Figure 5: Example nested class in JavaDecaf

7 Error reporting

If there are any errors in your code, JavaDecaf will terminate. It will report the error and tell you the location and reason for the problem.

JavaDecaf will also print warning messages if you have not indented your code correctly. It is advisable to fix the indentation before running your program - good indentation makes for good programming habits.