Tutorial on Files and Packages

> Based on the tutorial of "2020S-Java-A" designed by teaching group in SUSTech
>
> Modified (only change to markdown file) by ZHU Yueming in 2021. April. 12th
>
> Modified (Part of the content change from word to markdown) by JIA Yanhong in 2022. Oct. 27th
>
> Updated the order of demo by Yida Tao, Nov. 4 2022

Objectives

- Learn how to read and write text files in Java
- Learn to create packages to organize classes

Part 1. Read File and Write File

1 Write a Java program to get a list of all file/directory names in a directory.

```
import java.io.File;

public class ListFolder {
    public static void main(String a[]) {
        File file = new File("C:\\Users\\");
        String[] fileList = file.list();
        for (String name : fileList) {
            System.out.println(name);
        }

    }
}
```

Think

- What if we change `C:\\Users\\` to `C:\Users\`?
- What if we change `C:\\Users\\` to some non-existed folders?
- What should we do if we want to deep-traverse the folder, i.e., list all files and sub-folders?

2 Create a txt file named `bytes.txt` in your Intellij IDEA project, then add `1234_sdf~3` to this txt. Then run the following code:

```
public class ReadBytes {
    public static void main(String[] args) {
        try {
            FileInputStream inputStream = new
FileInputStream("src/lab9/bytes.txt");
            int item;
            while ((item = inputStream.read()) != -1) {
```

```
                System.out.printf("ascii = %d, char = '%c'\n", item, (char) item);
            }

        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}
```

Think

- What is the output and why?

3 Write a Java program to read all the content from an input text file, convert the content into uppercase letters, and write the result into an output file. Assume that the input file contains only English letters, numbers and punctuation marks.

```
import java.io.*;
public class ReadWriteTextFile
{
    public static void main(String[] args)
    {
        try
        {
            BufferedReader in=new BufferedReader(new
FileReader("src/TestIn.txt"));
            BufferedWriter writer=new BufferedWriter(new
FileWriter("src/TestOut.txt"));
            int temp;
            while ((temp=in.read())!=-1)
            {
                if (temp>='a' &&temp<='z')
                {
                    writer.write(temp-'a'+'A');
                }else
                {
                    writer.write(temp);
                }
            }
            in.close();
            writer.close();
        } catch (IOException e)
        {
            System.out.println("There is no this file!");
        }
    }
}
```

Think

- Where should we put `TestIn.txt` and `TextOut.txt`? How should we change the path accordingly?
- There are many ways to read and write text files; what we've shown here is only an example. You may get more details from chapter 17 in the text book **Java How to Program**.

Exercise 1: Grammar correction tool

Kids always make mistake in English writing. A common mistake is that they always forget to use capital letter at the beginning of a sentence. Please write a Java program to help them correct the mistakes.

Usually, the word after a full stop (.) is regarded as the beginning of a new sentence. However, a full stop (.) also means abbreviation. When a word is abbreviated after the first few letters, the traditional rule is to put a full stop after the abbreviation, for example, Dr., Mr. and FYI.. We can assume every abbreviation begins with a capital letter. This indicates whether the current stop means the end of sentence or an abbreviation.

The Java program should first read the text scanner, and then write the result to an text file.

Sample Input 1
today I borrow a book from my neighbor. he come and get it back tomorrow.

Sample Output 1
Today I borrow a book from my neighbor. He comes and gets it back tomorrow.

Sample Input 2
please get the report from BBC. news ASAP. as the boss want to read them now. he will not stay here until the evening.

Sample Output 2
Please get the report from BBC. news ASAP. as the boss want to read them now. He will not stay here until the evening.

Exercise 2: Reading and writing checkerboard files

- Step1. Drag `GobangChess.java`, `GobangTest.java` into `src` folder in your Intellij IDEA project.
- Step2. Create a txt file named `chessboard.txt` in the root path of your Intellij IDEA project, then add following content.

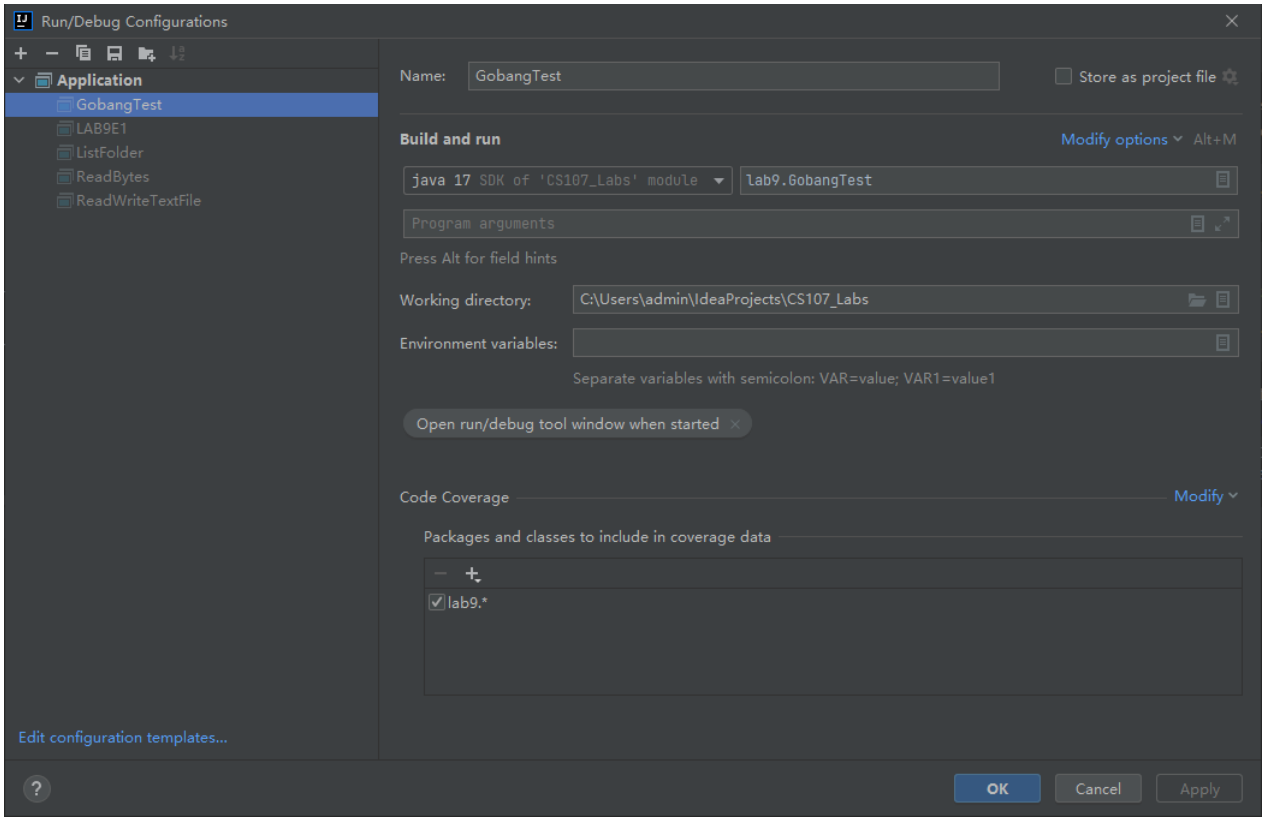
```
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,1,2,1,0,0,0,0
0,0,0,0,1,2,0,0,0,0
0,0,0,2,1,2,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
```

In the demo, this txt file represents a stored chessboard you need to load. The file can be read using any of the following methods:

```
public List<String> readFileByFileReader(String path)
```

```
/**
  * need JDK 11 or higher version
  */
public List<String> readFileByLib(String path)
```

- Step3. Check your project structure. Make sure that the input path and output path are correct w.r.t to the **working directory** (click Run -> Edit Configurations).



- Step4. Run `GobangTest.java`, then check the output and the new file named `new_chessboard.txt` created in this project.

Part 2. Package and classpath

In Command Line

Given the `Circle.java` file you wrote previously, add the following package declaration statement at the beginning of the `.java` file.

```
package lab8;
```

Now go to the directory where the `Circle.java` file resides and run the following command to compile the `Circle.java` file. Observe what would happen.

```
javac Circle.java
```

You will find a `Circle.class` file appearing in your working directory. (If you run the directory listing command or check in the directory, you can see both the `.java` and `.class` files).

Now suppose you want to run the `Circle` class. You might wish to run the `java Circle` command: Unfortunately, you will get the following error message:

```
PS C:\Users\admin\IdeaProjects\CS107_Labs\src\lab8> java Circle
错误: 找不到或无法加载主类 Circle
原因: java.lang.NoClassDefFoundError: lab8/Circle (wrong name: Circle)
```

This is because by adding a package declaration, the fully qualified name of the Circle class becomes `lab8.Circle`. We cannot simply use the simple name `Circle` to run the class.

We must put the `Circle.java` in the `src/lab8` directory, go to its parent `src` directory, and run the following command in the `src` directory :

```
PS C:\Users\admin\IdeaProjects\CS107_Labs\src\lab8> cd ../
PS C:\Users\admin\IdeaProjects\CS107_Labs\src> java lab8.Circle
```

You will get the following message:

```
PS C:\Users\admin\IdeaProjects\CS107_Labs\src> java lab8.Circle
错误: 在类 lab8.Circle 中找不到 main 方法, 请将 main 方法定义为:
   public static void main(String[] args)
否则 JavaFX 应用程序类必须扩展javafx.application.Application
```

This is normal information because we did not write the main method in `Circles.java.`

Tips: the `javac` command may run in 2 ways:

1. In directory `lab8`: `javac Cirle.java`
2. In the parent directory of `lab8`:

```
C:\Users\admin\IdeaProjects\CS107_Labs\src> javac lab8/Circle.java
```

In both ways, the `Circle.class` will be in the directory `src/lab8`.

Now, continue to create a `CircleTest.java` file with the following code in `lab9`:

```
package lab9;

import lab8.Circle;

public class CircleTest {
    public static void main(String[] args) {
        Circle c = new Circle(1, 1.0, 0.0, 0.0);
        c.position();
    }
}
```

In the code, we need to `import lab8.Circle` class because it is declared in another package.

Run the following command in the `src` directory:

```
PS C:\Users\admin\IdeaProjects\CS107_Labs\src> javac lab9/CircleTest.java
PS C:\Users\admin\IdeaProjects\CS107_Labs\src> java lab9.CircleTest
Position of the circle is (0.0, 1.0)
```

Note that for all the above steps, we assume that we don't change our working directory during the whole process. If we change to another directory and wish to run the `CircleTest` class from there, we need to specify the `classpath` as follows:

```
java -cp working-dir lab9.CircleTest
```

For example:

```
PS C:\Users\admin\IdeaProjects\CS107_Labs\src> cd ../
PS C:\Users\admin\IdeaProjects\CS107_Labs> java -cp src lab9.CircleTest
Position of the circle is (0.0, 1.0)
```

If the classpath contains several directories, we should use directory separators to separate them. Semicolon is the directory separator on Windows. On Unix/Linux/Mac, you should use colon. Below is an example:

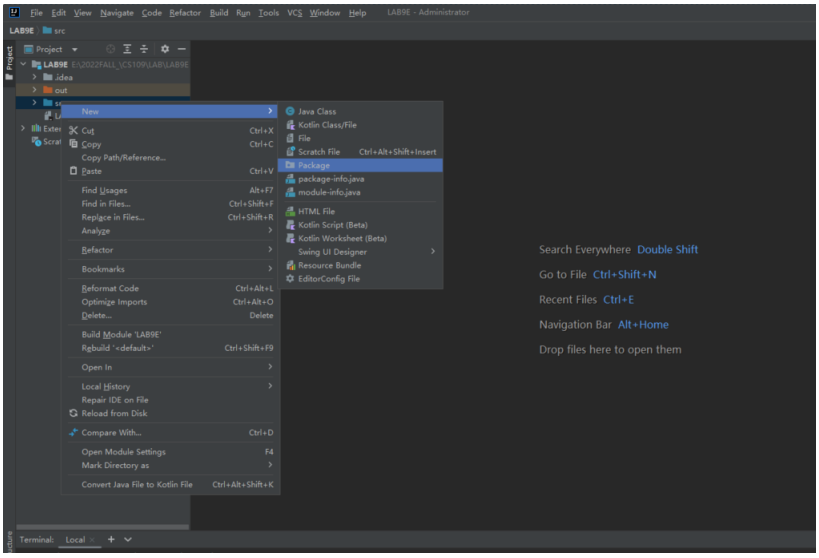
```
java -cp dir1;dir2;dir3 ClassToRun
```

The above tutorial explains package and classpath at a low level.

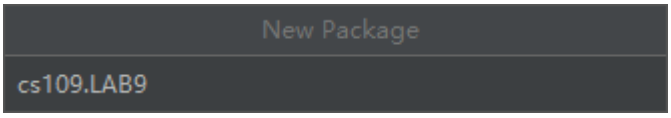
In IDE

In an IDE, creating packages and declaring classes in them is as easy as pie. We provide the steps below for IntelliJ IDEA.

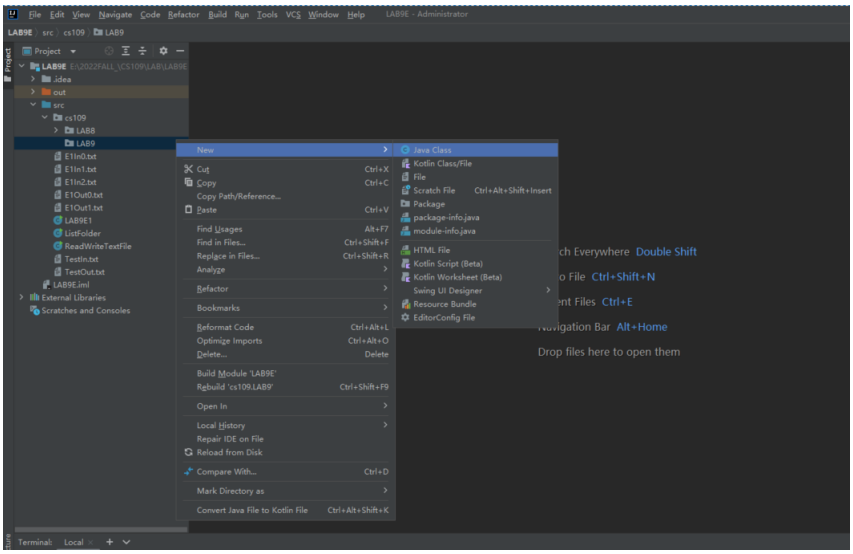
Step 1: In a project, right click on the src, then click New->Package



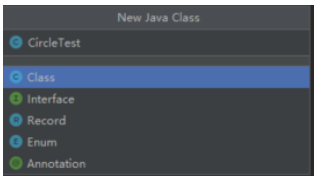
Step 2: Enter the package nam in the dialog box, click ok and you will see the package created.



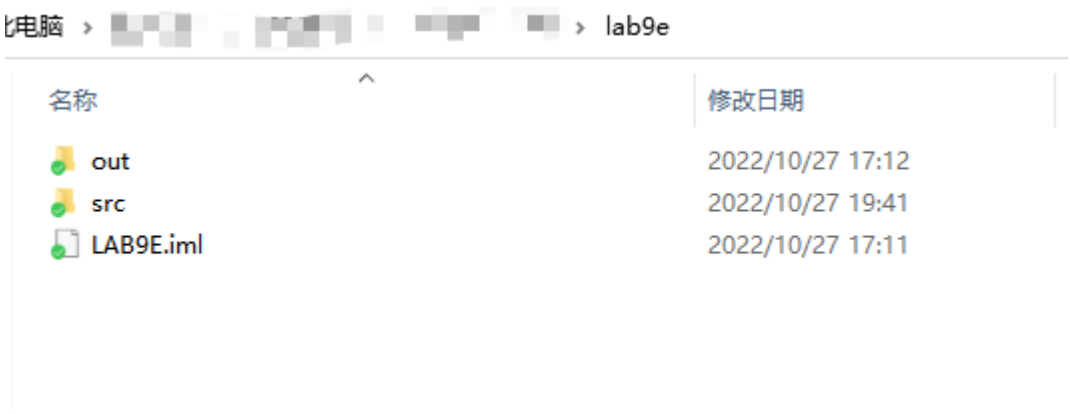
Step 3: Right click on the package, then click New->Java Class

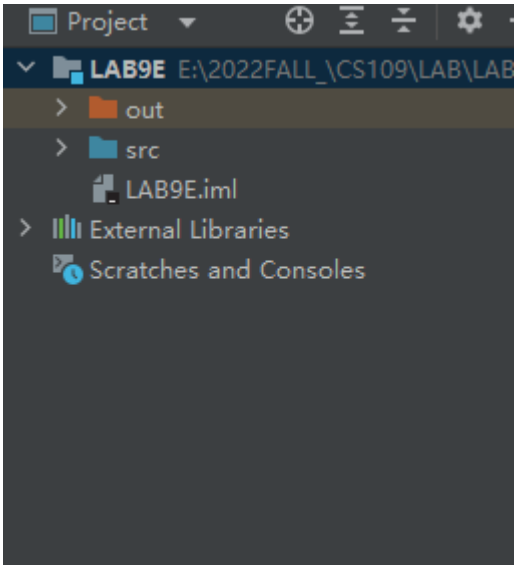


Step 4: Enter the class name in the dialog box, click ok and you will see the skeleton code of the newly created class. You will find that the package declaration statement is added automatically.



Now compile the class and go to the directory where the project is stored. You will see that project directory contains two sub-directories: “src” stores the .java source files and “out” stores the .class files after compilation.





If you check the `src` and out directories, you will find that the `CircleTest.java` and `CircleTest.class` files reside in the following directories, respectively:

```
src/cs109/LAB9/CircleTest.java
out/production/LAB9E/cs109/LAB9/CircleTest.class
```

IDEA helps manage all the stuff automatically. The way how source files and class files are organized may be different for other IDEs (for example, Eclipse).

Exercise 3:

For exercise 1, suppose that you've created a file `Lab9E1.java`.

- Put this `Lab9E1.java` file into a package `lab9`,
- Change the path of input file and output file accordingly.
- Executing the program in both IDEA and in command line.