Tutorial of Constructor and String

> 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. 6th
>
> Update by ZHU Yueming in 2021. Nov. 1st. Add File read and write demo

Objective

- Learn to declare constructors and use them to construct objects
- Learn to declare and use of static data field and `toString( )` method.
- Learn to use various String methods
- Learn to load file and write file.

Part 1: Constructors and instance methods

The Circle class defined in the previous lab does not contain any explicitly declared constructor. The Java compiler will provide a default constructor that would initialize all three fields (radius, x, y) to 0.0 when called. If we want to create a circle object, of which the three fields have the following values: radius = 2.0, x = 1.0, y = 1.0, we can write a main method like the one below.

```
public static void main(String[] args) {
    Circle c = new Circle();
    c.setRadius(2.0);
    c.setX(1.0);
    c.setY(1.0);
}
```

However, this is quite troublesome. A better solution is to declare constructors so that they can be called to construct objects with certain radius values and center positions. The following code declares two such constructors. The first constructor takes one argument to initialize the radius field (the fields x and y will be initialized to 0.0). The second constructor takes three arguments to initialize all three fields.

```
public Circle(double radius) {
    this.radius = radius;
}

public Circle(double radius, double x, double y) {
    this.radius = radius;
    this.x = x;
    this.y = y;
}
```

Note that in the constructors, "this" keyword is needed to differentiate the field access from method argument access. Now we can simply create the circle (radius = 2.0, x = 1.0, y = 1.0) with a constructor call. Much easier, right?

```
public static void main(String[] args) {
   Circle c = new Circle(2.0, 1.0, 1.0);
}
```

Continue to type the following code in the main method and see what happens.

```
Circle c = new Circle();
```

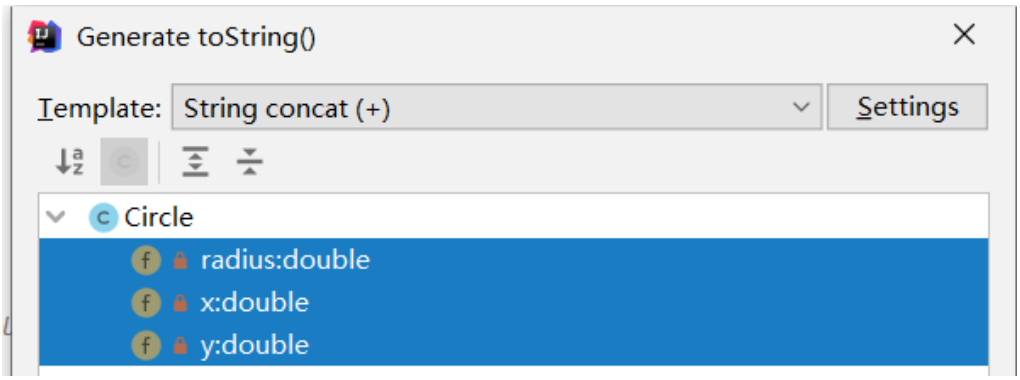
The code would not compile. Do you know why?

Part 2: toString( ) method

A special instance method **`toString( )`** which return a string related to the current object.

While the object is used as a string, the `toString( )` is invoked by default. For example:

- Create a Circle object c : `Circle c = new Circle(1.0, 1.0, 1.0);`
- Use `System.out.print(c)` to print the c, what's the output?
- Use `System.out.print(c.toString())` instead of `System.out.print(c)`, what's the output?
- Use IDEA to generated the `toString()` of the current class.



The following instance method `toString()` is generated, all the instance data field is added into the String:

```
public String toString() {
    return "Circle{" +
        "radius=" + radius +
        ", x=" + x +
        ", y=" + y +
        '}';
}
```

- Invoke `System.out.print(c)` and `System.out.print(c.toString())` again, what's the output?
- While change the `toString()` as follow code, invoke `System.out.print(c)`. what's the output, why?

```
public String toString(char z) {
    return "Circle{" +
          "radius=" + radius +
          ", x=" + x +
          ", y=" + y +
          ", z=" + z +
          '}';
}
```

Part 3: String manipulations

Please use String methods

(https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) to finish the tasks below. Methods in the Character class are also helpful:

https://docs.oracle.com/javase/8/docs/api/java/lang/Character.html.

Exercise 1

Write a program to check if a string provided by a user is a palindrome or not. A string is a palindrome is the reverse of the string is the same as the string (we do not differentiate upper-case and lower-case letters in this task). For example, "abba" is a palindrome. "#Aa#" and "0" are also palindromes.

Your program should continuously take user inputs for checking and stops when the user types “quit”.

A sample run:

```
Type a string ("quit" to finish): hello
hello is not a palindrome
Type a string ("quit" to finish): many
many is not a palindrome
Type a string ("quit" to finish): 0
0 is a palindrome
Type a string ("quit" to finish): 900
900 is not a palindrome
Type a string ("quit" to finish): #Aa#
#Aa# is a palindrome
Type a string ("quit" to finish): quit

Process finished with exit code 0
```

Exercise 2

Write a program to remove all repeated characters in a string provided by the user and return a new string without any repeating characters or white spaces. Please use `StringBuilder` to build the new string.

Sample runs:

```
Please type a string: hello
After removing repeating chars and spaces: helo
```

```
Please type a string:
Empty string, exit…
```

```
Please type a string: abcd  bcde cdef
After removing repeating chars and spaces: abcdef
```

Exercise 3

Write a program to count the occurrence of a substring in a string. The program should ask the user to input two strings s1 and s2, and output the number of occurrences of s2 in s1.

Sample runs:

```
s1: JavaExamplesJavaCodeJavaProgram
s2: Java
Found at index: 0
Found at index: 12
Found at index: 20
Total occurrences: 3

s1: abcd  bcde cdef
s2: bc
Found at index: 1
Found at index: 6
Total occurrences: 2

s1: abcdefg
s2: xyz
Total occurrences: 0
```

API References:

String methods:

https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

```
public int length()

public char charAt(int index)

public boolean startsWith(String prefix)

public boolean equals(Object anObject)
```

```
public boolean equalsIgnoreCase(String anotherString)

public String trim()

public int indexOf(String str)

public int indexOf(String str, int fromIndex)

public String substring(int beginIndex)

public String substring(int beginIndex, int endIndex)

public String[] split(String regex)

public char[] toCharArray()
```

Character methods:

https://docs.oracle.com/javase/8/docs/api/java/lang/Character.html

```
public static char toLowerCase(char ch)

public static boolean isWhitespace(char ch)
```

StringBuilder methods:

https://docs.oracle.com/javase/8/docs/api/java/lang/StringBuilder.html

```
public StringBuilder append(char c)

public String toString()
```

Part 4. Read File and Write FIle

Build the demo

- Step1. Drag `GobangChess.java`, `GobangTest.java` into one **generate sources root** 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. In the demo, this txt file represents a stored chessboard you need to load.

  The file is to test the method:

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

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

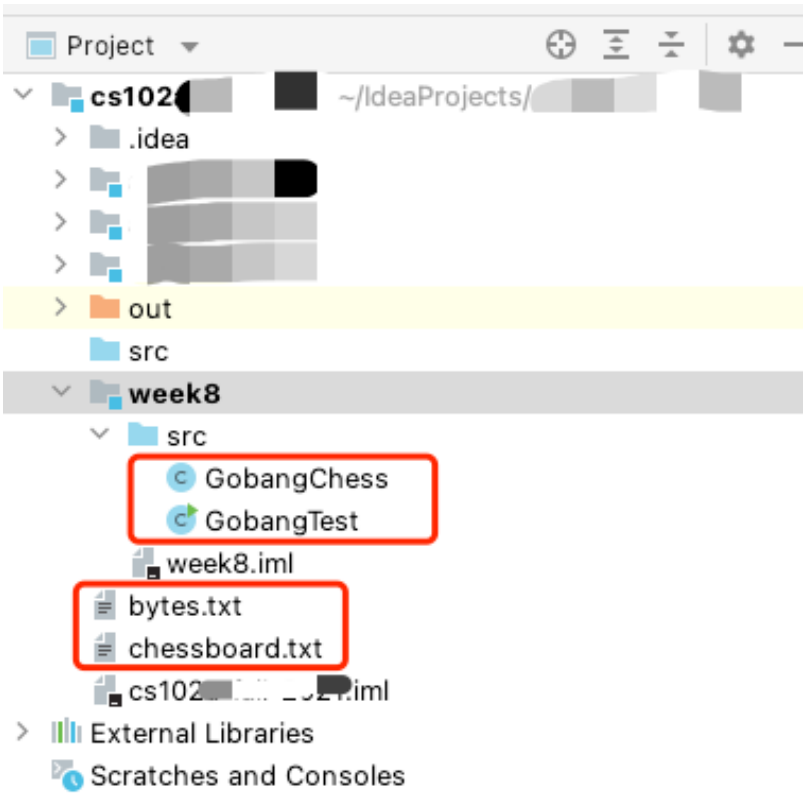
```
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,1,2,1,0,0,0,0
0,0,0,0,1,2,0,0,0,0
0,0,0,2,1,2,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0
```

- Step3. Create a txt file named `bytes.txt` in the root path of your Intellij IDEA project, then add following content. In the demo, this txt file is to test the method

```
public void readFileByByte(String path)
```

```
1234_sdf~3
```

- Step4. Check your project structure. (Following graph is an example of my structure)



- Step5. Run GobangTest.java, then check the output and the new file named `new_chessboard.txt` created in this project catalog.