CS 1050 – How to use the methods in the Toolkit class

and, among other methods, its leftPad and padString methods

Toolkit.java contains over a dozen handy methods. In this document we’ll see how to reference the Toolkit class and access the leftPad method.

Accessing the Toolkit class

Download the file Toolkit.java (it’s in Blackboard > Session Supplements > Flip\_Files.zip) and place it in the folder in which you keep your assignments. If you do not keep all your assignments in the same folder, you will have to place Toolkit.java in every folder in which you use the Toolkit.

Using the leftPad Method

The leftPad method with three parameters pads a number (not a string) so that it pads a certain number of spaces on the left, effectively right-justifying the result. One use is to line up columns with the same decimal point. The method’s three parameters are as follows:

numberToConvert A double or integer number to pad on the left

fieldWidth The total number of positions to allow for the resultant string

EditMask The edit mask used to convert the number to a string. This edit mask is the same as ones used with the DecimalFormat class

The four-parameter leftPad method lets you pad the string with the character of your choice as the fourth parameter. Note: The rightPad method works similarly, but pads the resultant string on the right effectively left-justifying the result. Let ^ denote a space and assume the variable “number” has the value 123.456. Then we get these results:

Toolkit.leftPad(number, 8, “0.0”) ^^^123.5 (Again, ^ denotes a space)

Toolkit.leftPad(number, 8, “0.00”) ^^123.46

Toolkit.leftPad(number, 8, “0.00”, “\*”) \*\*123.46 (Note the 4th parameter)

Toolkit.leftPad(number, 8, “0.00”, “$”) $$123.46

Toolkit.rightPad(number, 8, “0.0”) 123.5^^^

Example of Using the leftPad method in a Triangle Perimeter Calculation Program

Say we want to pad the three sides and the perimeter with leftPad.

Step 1: In the declaration area at the start of the main program, define the variables used to convert numbers to strings. Example:

double side1, side2, side3; // The three sides of a triangle

double perimeter; // Perimeter of the triangle = sum of the three sides

Step 2 resumes in the output display area where you display results. There are two approaches to do this. The first is useful if you display the information only once, while the second is useful if you display the information more than once.

Approach #1

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void outputResults(double side1, double side2, double side3,

double perimeter) {

final String MASK = “0.00”;

// Output the results to the console

System.out.println(“Side 1 = “ + Toolkit.leftPad(side1, 8, MASK)

+ “, side 2 = “ + Toolkit.leftPad(side2, 8, MASK) +

+ “, and side 3 = “ + Toolkit.leftPad(side3, 8, MASK)

+ “\nThe perimeter = “ + Toolkit.leftPad(perimeter, 10, “0.0”));

} // End outputResults

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Approach #2

public static void outputResults(double side1, double side2, double side3,

double perimeter) {

String side1Str, side2Str, side3Str; // Receive three left-padded sides

String perimeterStr; // Receive left-padded perimeter value

final String MASK = “0.00”;

// Prepare strings to display for the numeric values

side1Str = Toolkit.leftPad(side1, 8, MASK);

side2Str = Toolkit.leftPad(side2, 8, MASK);

side3Str = Toolkit.leftPad(side3, 8, MASK);

perimeterStr = Toolkit.leftPad(perimeter, 10, “0.0”);

// Output the results to the console

System.out.println(“Side 1 = “ + side1Str

+ “, side 2 = “ + side2Str

+ “, and side 3 = “ + side3Str

+ “\nThe perimeter = “ +strPerimeter);

} // End outputResults

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3. Use the padString method to pad strings on the right so that subsequent columns can begin at the same column number. rightPad’s two parameters (there is no edit mask with strings) are:

stringToPad The string to pad on the right

fieldWidth The total number of positions to allow for the resultant string

With “^” denoting a space, and variable “name” containing the name “John Smith” (10 characters), here are some examples of using padString, some with four parameters:

Toolkit.padString(name, 16) John Smith^^^^^^

Toolkit.padString(name, 16, “”, “ ”) John Smith^^^^^^ (same as previous line)

Toolkit.padString(name, 16, “ ”, “”) ^^^^^^John Smith

Toolkit.padString(name, 16, “ ”, “ ”) ^^^John Smith^^^

Toolkit.padString(name, 16, “\*”, “+”) \*\*\*John Smith+++

Toolkit.padString(name, 5, “ ”, “”) John Smith

The third and fourth optional parameters are the characters to use to pad on the left and right side, respectively, of the string. The 2-parameter format will handle all situations when you are padding on the right with a blank.