String Processing

Oracle® Certified Professional, Java® SE 7 Programmer

Module 5

Module 5 - Objectives

- Search, parse and build strings
- Search, parse, and replace strings by using regular expressions, using expression patterns for matching limited to: . (dot), * (star), + (plus), ?, \d, \D, \s, \S, \w, \W, \b. \B, [], ().
- Format strings using the formatting parameters: %b, %c, %d, %f, and %s in format strings.

String

- Strings in Java are classes. They are **not** primitives.
- There are two types of Strings: literals, and objects.
- Strings literals are created using quotes (“ ”).
- A String object is created using a constructor, e.g. String s = new String(“stringy”);
- Strings are immutable.

String Concatenation

- Strings can be concatenated using the + symbol.
- Strings can be concatenated using +=
- Strings can be concatenated using the concat() method.

StringBuffer

- A StringBuffer is considered heavy weight since the methods are synchronized (thread safe).
- A StringBuffer uses a builder pattern to allow chaining of append() methods.
- The synchronized methods made this class overkill for most implementations. Something newer was needed.
- StringBuffer is for use with mutable String operations.

StringBuilder

- StringBuilder is a drop-in method equivalent replacement for StringBuffer.
- There is no guarantee of thread safety with a StringBuilder.
- The StringBuilder should be considered for all mutable String operations which do not need thread safe operations.

Regular Expressions

- . (dot) – matches any character
- * (star) – matches zero, or more times.
- + (plus) – matches one, or more times.
- ? – matches once, or not at all.
- \d and \D – matches a digit (0-9) and a non digit respectively.
- \s and \S – matches a whitespace character: [\t\n\xoB\f\r] and a non-whitespace character: [^\s] respectively.

Regular Expressions (cont.)

- \w and \W – matches a word character: [a-zA-Z_0-9], or a non-word character [^\w] respectively.
- \b and \B – matches a word boundary, and non-word boundary respectively.
- [] – Character class. This is used to group matching characters.
- {} – indicates a repetition A{3} == AAA.
- () – indicates a matching group. ATCGATCGATCG== (ATCG){3}

Regular Expressions (cont.)

- The only way to learn regex (regular expressions) is to do them.
- Code a number of examples to determine what the characters, character classes, and boundary matchers do.

Formatting

- String formatting will save you a lot of time. If you are familiar with printf()and sprintf() methods from c\c++, these behave in a very similar fashion.
- The String.format() method returns a String object rather than a PrintStream like printf().
- Syntax: %[argument_index$][flags][width]conversion
- See Formatter in SE7 API for full explanation of syntax. We are only interested in the conversions.
- %b this is a conversion for a boolean value.

Formatting (cont.)

- %c – conversion for a Unicode character.
- %d – conversion for an integral type
- %f – conversion for a floating point type.
- %s – an Object, or String conversion.

creative commons

Attribution-NonCommercial-ShareAlike 3.0 Unported