Regular Expressions in Java

Tutorial 1

CSC 340

- Java's regexe classes are kept in package java.util.regex.Pattern.

- There are only two classes in this package: Pattern and Matcher.
- You should browse the Javadoc for
  - Pattern class,
  - followed by Matcher class.

Three steps are required to perform regexe matching:

1. Allocate a Pattern object.
   - There is no constructor for the Pattern class.
   - Instead, you invoke the static method Pattern.compile(*regexeString*) to compile the *regexeString*, which returns a Pattern instance.

2. Allocate a Matcher object.
   - Again, there is no constructor for the Matcher class.
   - Instead, you invoke the matcher(*inputString*) method from the Pattern instance (created in Step 1).
   - You also bind the input sequence to this Matcher.

3. Use the Matcher instance (created in Step 2) to perform the matching and process the matching result.
    - The Matcher class provides a few boolean methods for performing the matches:

Important Methods in Matcher class

- **boolean find()**: scans the input sequence to look for the *next* substring that matches the pattern.
- If match is found, you can use the group(), start() and end() to retrieve the matched subsequence and its starting and ending indices.

- **boolean matches():** try to match the entire input sequence against the regexe pattern. It returns true if the entire input sequence matches the pattern.
- **boolean lookingAt()**: try to match the input sequence, starting from the beginning, against the regexe pattern. It returns true if a *prefix* of the input sequence matches the pattern.

- To perform case-insensitive matching, use Pattern.compile(*regexeString*, Pattern.CASE_INSENSITIVE) to create the Pattern instance (as commented out in the above example).

Methods in Matcher Useful for replacing the matching string with another

- String replaceAll(String replacement);
  Performs the matching and process the matching result

- String replaceFirst(String replacement);
  Replaces the first match only

The String.split() Method

- The String class contains a method split(), which takes a regular expression and splits this String object into an array of Strings.
- // In String class public

  String[] split(String regex)

Example on String.Split

```
public class StringSplitTest {
    public static void main(String[] args) {
        String source = "There are thirty-three big-apple";
        String[] tokens = source.split("\\s+|-"); // whitespace(s) or –
        for (String token : tokens)
            System.out.println(token);
    } }
```

Output:

```
There
are
thirty
three
big
apple
```

Regular Expression Syntax as Supported by the java.util.regex API

- The metacharacters supported by this API are: <([{\^-=$!|]})?*+.>

More on Regular Expressions

- http://docs.oracle.com/javase/tutorial/essential/regex/intro.html
- A good video tutorial on youtube (highly recommended)
  - http://www.youtube.com/watch?v=EkluES9Rvak