**Java Regular Expressions (java regex) Tutorial with examples**

compact1, compact2, compact3

java.util.regex

## Class Pattern

* [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)
  + java.util.regex.Pattern
* **All Implemented Interfaces:**

[Serializable](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html)

public final class **Pattern** extends [Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) implements [Serializable](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html)

**Regular expressions** are used for defining String patterns that can be used for searching, manipulating and editing a text. These expressions are also known as **Regex** (short form of Regular expressions).

The java.util.regex API (the package which we need to import while dealing with Regex) has two main classes:

1. java.util.regex.Pattern – Used for defining patterns

2) java.util.regex.Matcher – Used for performing match operations on text

using patterns.

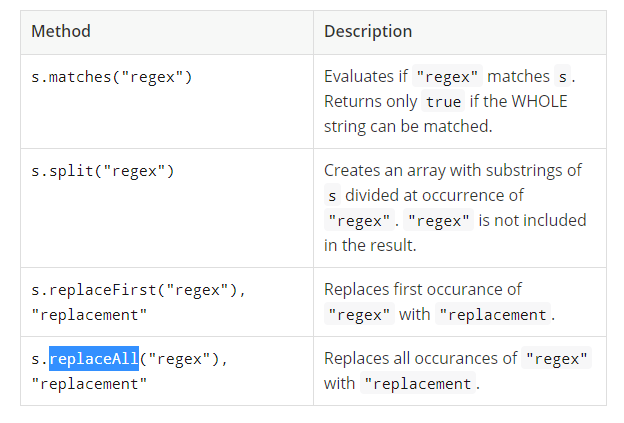
## **Java Regex Package**

The java.util.regex package consists of three classes: Pattern, Matcher and PatternSyntaxException:

* Pattern object is a compiled regex.
* The Pattern class provides no public constructors.
* To create a pattern, we must first invoke one of its public static compile methods, which will then return a Pattern object. These methods accept a regular expression as the first argument.

Thw **compile(String)** method of the **Pattern** class in Java is used to create a pattern from the regular expression passed as parameter to method. Whenever you need to match a text against a regular expression pattern more than one time, create a Pattern instance using the Pattern.compile() method.

* Matcher object interprets the pattern and performs match operations against an input String. It also defines no public constructors. We obtain a Matcher object by invoking the matcher method on a Pattern object.
* PatternSyntaxException object is an unchecked exception that indicates a syntax error in a regular expression pattern.



A compiled representation of a regular expression.

A regular expression, specified as a string, must first be compiled into an instance of this class. The resulting pattern can then be used to create a [Matcher](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Matcher.html) object that can match arbitrary [character sequences](https://docs.oracle.com/javase/8/docs/api/java/lang/CharSequence.html) against the regular expression. All of the state involved in performing a match resides in the matcher, so many matchers can share the same pattern.

A typical invocation sequence is thus

Pattern p = Pattern.[compile](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html#compile-java.lang.String-)("a\*b");

Matcher m = p.[matcher](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html#matcher-java.lang.CharSequence-)("aaaaab");

boolean b = m.[matches](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Matcher.html#matches--)();

A [matches](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html#matches-java.lang.String-java.lang.CharSequence-) method is defined by this class as a convenience for when a regular expression is used just once. This method compiles an expression and matches an input sequence against it in a single invocation. The statement

boolean b = Pattern.matches("a\*b", "aaaaab");

is equivalent to the three statements above, though for repeated matches it is less efficient since it does not allow the compiled pattern to be reused.

Instances of this class are immutable and are safe for use by multiple concurrent threads. Instances of the [Matcher](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Matcher.html) class are not safe for such use.

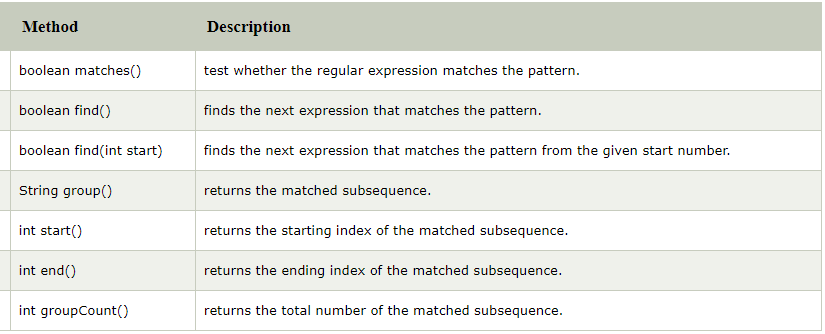
#### **java.util.regex package**

The Matcher and Pattern classes provide the facility of Java regular expression. The java.util.regex package provides following classes and interfaces for regular expressions.

1. MatchResult interface
2. Matcher class
3. Pattern class
4. PatternSyntaxException class

## **Matcher class**

It implements the **MatchResult** interface. It is a regex engine which is used to perform match operations on a character sequence.



## **Pattern class**

It is the compiled version of a regular expression. It is used to define a pattern for the regex engine.

