**Split String**

public String[] split(String regex)

Splits this string around matches of the given [regular expression](https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html#sum).

This method works as if by invoking the two-argument split method with the given expression and a limit argument of zero. Trailing empty strings are therefore not included in the resulting array.

The string "boo:and:foo", for example, yields the following results with these expressions:

|  |  |
| --- | --- |
| **Regex** | **Result** |
| : | { "boo", "and", "foo" } |
| o | { "b", "", ":and:f" } |

Parameters:

regex - the delimiting regular expression

Returns:

The array of strings computed by splitting this string around matches of the given regular expression

Throws:

PatternSyntaxException - if the regular expression's syntax is invalid

The **string split()** method breaks a given string around matches of the given regular expression. After splitting against the given regular expression, this method returns a string array.

Input String: 016-78967

Regular Expression: -

Output : {"016", "78967"}

Following are the two variants of the split() method in Java:

### **1. Public String [] split ( String regex, int limit)**

**Parameters:**

* regex – a delimiting regular expression
* Limit – the resulting threshold

**Returns:** An array of strings is computed by splitting the given string.

**Exception Thrown:** *PatternSyntaxException* – if the provided regular expression’s syntax is invalid.

**The limit parameter can have 3 values:**

* **limit > 0 –** If this is the case, then the pattern will be applied at most limit-1 times, the resulting array’s length will not be more than n, and the resulting array’s last entry will contain all input beyond the last matched pattern.
* **limit < 0 –** In this case, the pattern will be applied as many times as possible, and the resulting array can be of any size.
* **limit = 0 –** In this case, the pattern will be applied as many times as possible, the resulting array can be of any size, and trailing empty strings will be discarded.

**Here’s how it works:**  
   
Let the string that is to be split is – **geekss@for@geekss**

| **Regex** | **Limit** | **Result** |
| --- | --- | --- |
| @ | 2 | {“geekss”, ”for@geekss”} |
| @ | 5 | {“geekss”, ”for”, ”geekss”} |
| @ | -2 | {“geekss”, ”for”, ”geekss”} |
| s | 5 | {“geek”, ”“, “@for@geek”, “”, “”} |
| s | -2 | {“geek”, ” “, ” “, “@for@geek”, “”, “”} |
| s | 0 | {“geek”, ””, ”@for@geek”} |