Real's HowTo RSS FEED 8

Search
Google™ Custom Search

Split a string using String.split()

TAG(S): STRING/NUMBER

The String class has a split() (since 1.4) method that will **return a String array**.

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real-How-To";
    System.out.println
      (java.util.Arrays.toString(testString.split("-")));
    // output : [Real, How, To]
  }
}
```

split() is based on regex expression, a special attention is needed with some characters which have a special meaning in a regex expression.

For example :

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real.How.To";
    // bad
    System.out.println
       (java.util.Arrays.toString(testString.split(".")));
    // output : []

    // good
    System.out.println
       (java.util.Arrays.toString(testString.split("\\.")));
    // output : [Real, How, To]
  }
}
```

And

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real|How|To";
    // bad
    System.out.println
       (java.util.Arrays.toString(testString.split("|")));
    // output : [, R, e, a, l, |, H, o, w, |, T, o]

    // good
    System.out.println
       (java.util.Arrays.toString(testString.split("\\|")));
    // output : [Real, How, To]
  }
}
```

The special character needs to be escaped with a "\" but since "\" is also a special character in Java, you need to escape it again with another "\" !

Consider this example

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real|How|To|||";
    System.out.println
       (java.util.Arrays.toString(testString.split("\\|")));
    // output : [Real, How, To]
  }
}
```

The result does not include the empty strings between the "|" separator. To keep the empty strings :

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real|How|To|||";
    System.out.println
       (java.util.Arrays.toString(testString.split("\\|", -1)));
    // output : [Real, How, To, , , ]
  }
}
```

See split(String,int).

String.split() is only available since JDK 1.4.

With previous version, java.util.StringTokeniser can be used.

See this HowTo

Some notes from A. Gonzales about String.split()

Special cases using String.split():

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    System.out.println
       (java.util.Arrays.toString("  s".split(" ")));
    // output : [, , s]

    System.out.println
       (java.util.Arrays.toString("".split("")));
    // output : []

    System.out.println
       (java.util.Arrays.toString("  ".split(" ")));
    // output : []

    System.out.println
       (java.util.Arrays.toString("      ".split(" ")));
    // output : []

    System.out.println
       (java.util.Arrays.toString(" s ".split(" ")));
    // output : [, s]
  }
}
```

It's important to note that an invocation like:

```
param = req.getParam(...);
String[] words = param.split(" ");
String firstWord = words[0];
```

will generate a NullPointerException if param.equals(" ").

Using *split()* with a space can be a problem. Consider the following :

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real  How To";  // extra space

    System.out.println
       (java.util.Arrays.toString(testString.split(" ")));
    // output : [Real, , How, To]
  }
}
```

We have an extra element. The fix is to specify a regular expression to match one or more spaces.

```
public class StringSplit {
  public static void main(String args[]) throws Exception{
    String testString = "Real  How To";
    System.out.println
       (java.util.Arrays.toString(testString.split("\\s+")));
    // output : [Real, How, To]
  }
}
```

Since String.split() is based on regular expression, you can make some *complex* operations with a simple call!

```
String testString = "{RealHowto}{java-0438.html}{usage of String.split()}";
System.out.println
   (java.util.Arrays.toString(testString.split("[{}]")));
// output : [, RealHowto, , java-0438.html, , usage of String.split()]
// note : extra empty elements :-(
```

To split a long string into into fixed-length parts. In this example, we split in groups of 3 characters :

```
String testString = "012345678901234567890";
System.out.println
   (java.util.Arrays.toString(testString.split("(?<=\\G.{3})")));
// output : [012, 345, 678, 901, 234, 567, 890]
```

To split but keep the separator :

```
 String testString = "RealHowto!java-0438.html!usage of String.split()!";
 System.out.println
    (java.util.Arrays.toString(testString.split("(?<=[!])")));
 // output : [RealHowto!, java-0438.html!, usage of String.split()!]
```

If you find this article useful, consider making a small donation to show your support for this Web site and its content.