StringUtils 工具类的常用方法

Apache Commons 通用扩展包基本上是每个项目都会使用的，只是使用的多少不同而已，一般情况下 lang 包用作 jdk 的基础语言扩展。考虑到 commons 的名气很响，在实际开发中用到的地方也很多，故在此略作介绍，以备项目组在实际开发中使用。

org.apache.commons.lang.StringUtils.StringUtils 方法的操作对象是 java.lang.String 类型的对象，是对 JDK 提供的 String 类型操作方法的补充，并且是 null 安全的(即如果输入参数 String 为 null 则不会抛出 NullPointerException ，而是做了相应处理，例如，如果输入为 null 则返回也是 null 等，具体可以查看源代码)。

除了构造器，StringUtils 中一共有 130 多个方法，并且都是 static 的，所以我们可以这样调用 StringUtils.xxx()

下面分别对一些常用方法做简要介绍：

1. public static boolean isEmpty(String str)
   判断某字符串是否为空，为空的标准是 str==null 或 str.length()==0
   下面是 StringUtils 判断是否为空的示例：

```
StringUtils.isEmpty(null) = true
StringUtils.isEmpty("") = true
StringUtils.isEmpty(" ") = false //注意在 StringUtils 中空格作非空处理
StringUtils.isEmpty("   ") = false
StringUtils.isEmpty("bob") = false
StringUtils.isEmpty(" bob ") = false
```

2. public static boolean isNotEmpty(String str)
   判断某字符串是否非空，等于 !isEmpty(String str)
   下面是示例：

```
StringUtils.isNotEmpty(null) = false
StringUtils.isNotEmpty("") = false
StringUtils.isNotEmpty(" ") = true
StringUtils.isNotEmpty("         ") = true
StringUtils.isNotEmpty("bob") = true
StringUtils.isNotEmpty(" bob ") = true
```

3. public static boolean isBlank(String str)
   判断某字符串是否为空或长度为 0 或由空白符(whitespace) 构成
   下面是示例：

```
StringUtils.isBlank(null) = true
```

StringUtils.isBlank("") = true
StringUtils.isBlank(" ") = true
StringUtils.isBlank(" ") = true
StringUtils.isBlank("\t \n \f \r") = true //对于制表符、换行符、换页符和回车符

StringUtils.isBlank() //均识为空白符
StringUtils.isBlank("\b") = false //"\b"为单词边界符
StringUtils.isBlank("bob") = false
StringUtils.isBlank(" bob ") = false

4. public static boolean isNotBlank(String str)
判断某字符串是否不为空且长度不为 0 且不由空白符(whitespace) 构成，等于 !isBlank(String str)
下面是示例：

StringUtils.isNotBlank(null) = false
StringUtils.isNotBlank("") = false
StringUtils.isNotBlank(" ") = false
StringUtils.isNotBlank(" ") = false
StringUtils.isNotBlank("\t \n \f \r") = false
StringUtils.isNotBlank("\b") = true
StringUtils.isNotBlank("bob") = true
StringUtils.isNotBlank(" bob ") = true

5. public static String trim(String str)
去掉字符串两端的控制符(control characters, char <= 32) ， 如果输入为 null 则返回 null
下面是示例：
StringUtils.trim(null) = null
StringUtils.trim("") = ""
StringUtils.trim(" ") = ""
StringUtils.trim(" \b \t \n \f \r ") = ""
StringUtils.trim(" \n\tss \b") = "ss"
StringUtils.trim(" d d dd ") = "d d dd"
StringUtils.trim("dd ") = "dd"
StringUtils.trim(" dd ") = "dd"

6. public static String trimToNull(String str)
去掉字符串两端的控制符(control characters, char <= 32) ,如果变为 null 或""，则返回 null
下面是示例：
StringUtils.trimToNull(null) = null
StringUtils.trimToNull("") = null
StringUtils.trimToNull(" ") = null

```
StringUtils.trimToNull("     \b \t \n \f \r    ") = null
StringUtils.trimToNull("     \n\tss   \b") = "ss"
StringUtils.trimToNull(" d   d dd     ") = "d   d dd"
StringUtils.trimToNull("dd     ") = "dd"
StringUtils.trimToNull("     dd       ") = "dd"
```

7. public static String trimToEmpty(String str)

去掉字符串两端的控制符(control characters, char <= 32),如果变为 null 或 "" ,则返回 ""

下面是示例:

```
StringUtils.trimToEmpty(null) = ""
StringUtils.trimToEmpty("") = ""
StringUtils.trimToEmpty(" ") = ""
StringUtils.trimToEmpty("     \b \t \n \f \r    ") = ""
StringUtils.trimToEmpty("     \n\tss   \b") = "ss"
StringUtils.trimToEmpty(" d   d dd     ") = "d   d dd"
StringUtils.trimToEmpty("dd     ") = "dd"
StringUtils.trimToEmpty("     dd       ") = "dd"
```

8. public static String strip(String str)

去掉字符串两端的空白符(whitespace) ,如果输入为 null 则返回 null

下面是示例(注意和 trim() 的区别):

```
StringUtils.strip(null) = null
StringUtils.strip("") = ""
StringUtils.strip(" ") = ""
StringUtils.strip("     \b \t \n \f \r    ") = "\b"
StringUtils.strip("     \n\tss   \b") = "ss   \b"
StringUtils.strip(" d   d dd     ") = "d   d dd"
StringUtils.strip("dd     ") = "dd"
StringUtils.strip("     dd       ") = "dd"
```

9. public static String stripToNull(String str)

去掉字符串两端的空白符(whitespace) ,如果变为 null 或"",则返回 null

下面是示例(注意和 trimToNull() 的区别):

```
StringUtils.stripToNull(null) = null
StringUtils.stripToNull("") = null
StringUtils.stripToNull(" ") = null
StringUtils.stripToNull("     \b \t \n \f \r    ") = "\b"
StringUtils.stripToNull("     \n\tss   \b") = "ss   \b"
StringUtils.stripToNull(" d   d dd     ") = "d   d dd"
StringUtils.stripToNull("dd     ") = "dd"
StringUtils.stripToNull("     dd       ") = "dd"
```

10. public static String stripToEmpty(String str)

去掉字符串两端的空白符(whitespace) ，如果变为 null 或"" ，则返回""
下面是示例(注意和 trimToEmpty() 的区别)：

```
StringUtils.stripToNull(null) = ""
StringUtils.stripToNull("") = ""
StringUtils.stripToNull(" ") = ""
StringUtils.stripToNull("     \b \t \n \f \r    ") = "\b"
StringUtils.stripToNull("     \n\tss   \b") = "ss   \b"
StringUtils.stripToNull(" d   d dd     ") = "d   d dd"
StringUtils.stripToNull("dd     ") = "dd"
StringUtils.stripToNull("     dd       ") = "dd"
```

以下方法只介绍其功能，不再举例：

11. public static String strip(String str, String stripChars)
去掉 str 两端的在 stripChars 中的字符。
如果 str 为 null 或等于"" ，则返回它本身；
如果 stripChars 为 null 或"" ，则返回 strip(String str) 。

12. public static String stripStart(String str, String stripChars)
和 11 相似，去掉 str 前端的在 stripChars 中的字符。

13. public static String stripEnd(String str, String stripChars)
和 11 相似，去掉 str 末端的在 stripChars 中的字符。

14. public static String[] stripAll(String[] strs)
对字符串数组中的每个字符串进行 strip(String str) ，然后返回。
如果 strs 为 null 或 strs 长度为 0，则返回 strs 本身

15. public static String[] stripAll(String[] strs, String stripChars)
对字符串数组中的每个字符串进行 strip(String str, String stripChars) ，然后返回。
如果 strs 为 null 或 strs 长度为 0，则返回 strs 本身

16. public static boolean equals(String str1, String str2)
比较两个字符串是否相等，如果两个均为空则也认为相等。

17. public static boolean equalsIgnoreCase(String str1, String str2)
比较两个字符串是否相等，不区分大小写，如果两个均为空则也认为相等。

18. public static int indexOf(String str, char searchChar)
返回字符 searchChar 在字符串 str 中第一次出现的位置。
如果 searchChar 没有在 str 中出现则返回-1，
如果 str 为 null 或 "" ，则也返回-1

19. public static int indexOf(String str, char searchChar, int startPos)

返回字符 searchChar 从 startPos 开始在字符串 str 中第一次出现的位置。
如果从 startPos 开始 searchChar 没有在 str 中出现则返回-1，
如果 str 为 null 或 "" ，则也返回-1

20. public static int indexOf(String str, String searchStr)
返回字符串 searchStr 在字符串 str 中第一次出现的位置。
如果 str 为 null 或 searchStr 为 null 则返回-1，
如果 searchStr 为 "" ,且 str 为不为 null ，则返回 0，
如果 searchStr 不在 str 中，则返回-1

21. public static int ordinalIndexOf(String str, String searchStr, int ordinal)
返回字符串 searchStr 在字符串 str 中第 ordinal 次出现的位置。
如果 str=null 或 searchStr=null 或 ordinal<=0 则返回-1
举例(*代表任意字符串)：
```
StringUtils.ordinalIndexOf(null, *, *) = -1
StringUtils.ordinalIndexOf(*, null, *) = -1
StringUtils.ordinalIndexOf("", "", *) = 0
StringUtils.ordinalIndexOf("aabaabaa", "a", 1) = 0
StringUtils.ordinalIndexOf("aabaabaa", "a", 2) = 1
StringUtils.ordinalIndexOf("aabaabaa", "b", 1) = 2
StringUtils.ordinalIndexOf("aabaabaa", "b", 2) = 5
StringUtils.ordinalIndexOf("aabaabaa", "ab", 1) = 1
StringUtils.ordinalIndexOf("aabaabaa", "ab", 2) = 4
StringUtils.ordinalIndexOf("aabaabaa", "bc", 1) = -1
StringUtils.ordinalIndexOf("aabaabaa", "", 1) = 0
StringUtils.ordinalIndexOf("aabaabaa", "", 2) = 0
```

22. public static int indexOf(String str, String searchStr, int startPos)
返回字符串 searchStr 从 startPos 开始在字符串 str 中第一次出现的位置。
举例(*代表任意字符串)：
```
StringUtils.indexOf(null, *, *) = -1
StringUtils.indexOf(*, null, *) = -1
StringUtils.indexOf("", "", 0) = 0
StringUtils.indexOf("aabaabaa", "a", 0) = 0
StringUtils.indexOf("aabaabaa", "b", 0) = 2
StringUtils.indexOf("aabaabaa", "ab", 0) = 1
StringUtils.indexOf("aabaabaa", "b", 3) = 5
StringUtils.indexOf("aabaabaa", "b", 9) = -1
StringUtils.indexOf("aabaabaa", "b", -1) = 2
StringUtils.indexOf("aabaabaa", "", 2) = 2
StringUtils.indexOf("abc", "", 9) = 3
```

23. public static int lastIndexOf(String str, char searchChar)
基本原理同 18

24. public static int lastIndexOf(String str, char searchChar, int startPos)
    基本原理同 19

25. public static int lastIndexOf(String str, String searchStr)
    基本原理同 20

26. public static int lastIndexOf(String str, String searchStr, int startPos)
    基本原理同 22

另附：

String 的 split(String regex) 方法的用法
如果我们需要把某个字符串拆分为字符串数组，则通常用 split(String regex) 来实现。

例如：

Java 代码

```
1. String str = "aa,bb,cc,dd";
2. String[] strArray = str.split(",");
3. System.out.println(strArray.length);
4.   for (int i = 0; i < strArray.length; i++) {
5.         System.out.println(strArray[i]);
6. }
```

结果为：
4
aa
bb
cc
dd

如果，
String str = "aa.bb.cc.dd";
String[] strArray = str.split(".");

则结果为：0

为什么结果不是我们所想的呢，原因是参数 String regex 是正则表达式 (regular expression) 而不是普通字符串，而 "." 在正则表达式中有特殊含义，表示匹配所有单个字符。如果要那样拆分，我们必须给 "." 进行转义，String[] strArray = str.split(".") 修改为 String[] strArray = str.split("\\.") 即可。

另外有关 StringUtils 的详细 API 请参见官方网站 。