commons-lang3

- description : commons-lang3 와 관련된 내용
- author : 주레피
- email : dhan@repia.com
- lastupdate : 2020-03-24

MavanRepository

```
<!-- https://mvnrepository.com/artifact/org.apache.commons/commons-lang3 -->
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-lang3</artifactId>
    <version>3.8</version>
</dependency>
```

MVN Repository

StringEscapeUtils

- [ ] StringEscapeUtils (할당된 사용자 천호동밤안개. 마감일 2020-04-06)

StringUtils

> StringUtils.abbreviate (문자열, 표시할 문자 수)

긴 문자열을 원하는 만큼만 보여주고 나머지는 말줄임표로 생략해준다.
표시할 문자 수는 4 이상이어야 한다.
표시할 문자 수는 자동으로 붙는 말줄임표 … (3자) 를 포함한 수이다.

```
String abbreviate = StringUtils.abbreviate("ShowMeTheMoney", 7);
System.out.println(abbreviate);
//출력결과 : Show...
```

> StringUtils.appendIfMissing (문자열, 추가할단어, 마지막단어)

마지막단어가 지정한 문자와 다를경우, 추가할단어를 붙여준다.
대소문자를 구분한다.

```
String appendIfMissing = StringUtils.appendIfMissing("ShowMeTheMoney", "-
```

```
From.Choi", "Money");
System.out.println(appendIfMissing);
//출력결과 : ShowMeTheMoney

appendIfMissing = StringUtils.appendIfMissing("ShowMeTheMoney", "-
From.Choi", "test");
System.out.println(appendIfMissing);
//출력결과 : ShowMeTheMoney-From.Choi
```

> StringUtils.appendIfMissingIgnoreCase (문자열, 추가할단어, 마지막단어)

StringUtils.appendIfMissing 과 같지만 대소문자를 구분하지 않는다.

```
String appendIfMissingIgnoreCase =
StringUtils.appendIfMissingIgnoreCase("ShowMeTheMoney", "-From.Choi",
"money");
System.out.println(appendIfMissingIgnoreCase);
//출력결과 : ShowMeTheMoney
```

> StringUtils.capitalize (문자열)

첫글자를 대문자로 변환한다.

```
String capitalize = StringUtils.capitalize("sss");
System.out.println(capitalize);
//출력결과 : Sss
```

> StringUtils.chomp (문자열)

마지막에 개행문자 \n, \r, \r\n 이 있을 경우 제거한다.

```
String chomp = StringUtils.chomp("sss\n");
System.out.println(chomp);
//출력결과 : sss
```

> StringUtils.chomp (문자열, 제거할문자)

마지막에 제거할 문자가 있을 경우 제거한다.

```
String chomp = StringUtils.chomp("sss#","#");
System.out.println(chomp);
//출력결과 : sss
```

> StringUtils.chop (문자열)

마지막 문자 하나를 제거한다.

```
String chop = StringUtils.chop("sss");
System.out.println(chop);
//출력결과 : ss
```

> StringUtils.center (문자열, 글자수)

문자열에서 부족한 만큼 공백이 추가된다.
문자열을 중심으로 오른쪽, 왼쪽 번갈아가며 추가된다.

```
String center = StringUtils.center("sss", 5);
System.out.println("'" + center + "'");
//출력결과 : ' sss '
```

> StringUtils.center (문자열, 글자수, 추가단어)

문자열에서 부족한 만큼 추가단어가 붙는다.
문자열을 중심으로 오른쪽, 왼쪽 번갈아가며 추가된다.

```
String center = StringUtils.center("sss", 6, "#");
System.out.println(center);
//출력결과 : #sss##
```

> StringUtils.compare (문자열, 비교할문자열)

두 문자열을 십진수로 변환 후 비교값을 출력한다.

```
//'가', '나' 의 십진수 변환 값
System.out.println((int) '가');
System.out.println((int) '나');
//출력결과 : 44032
//출력결과 : 45208

int compare = StringUtils.compare("가", "나");
System.out.println(compare);
//출력결과 : -1176
int compare2 = StringUtils.compare("나", "가");
System.out.println(compare2);
//출력결과 : 1176
```

> StringUtils.compareIgnoreCase (문자열, 비교할문자열)

StringUtils.compare 과 같지만 대소문자를 구분하지 않는다.

```
int compareIgnoreCase = StringUtils.compareIgnoreCase("A", "a");
System.out.println(compareIgnoreCase);
//출력결과 : 0
```

> StringUtils.contains (문자열, 비교할문자열)

문자열안에 비교할 문자열이 포함되어 있으면 'true'를 반환한다.

```
boolean contains = StringUtils.contains("Hello", "el");
System.out.println(contains);
//출력결과 : true
```

> StringUtils.containsAny (문자열, 비교할문자열, 비교할문자열)

문자열안에 비교할 문자열이 하나라도 포함되어있으면 'true'를 반환한다.

```
boolean containsAny = StringUtils.containsAny("안녕하세요", "안김");
System.out.println(containsAny);
//출력결과 : true
```

> StringUtils.containsIgnoreCase (문자열, 비교할문자열)

StringUtils.contains 과 같지만 대소문자를 구분하지 않는다.

```
boolean containsIgnoreCase = StringUtils.containsIgnoreCase("HELLO", "he");
System.out.println(containsIgnoreCase);
//출력결과 : true
```

> StringUtils.containsNone (문자열, 비교할문자열)

문자열안에 비교할 문자열이 하나라도 포함되어있으면 'false'를 반환한다.

```
boolean containsNone = StringUtils.containsNone("hello", "aaah");
System.out.println(containsNone);
//출력결과 : false
```

> StringUtils.containsOnly (문자열, 비교할문자열)

문자열안에 비교할 문자열이 모두 포함되어있을때만 'true'를 반환한다.

```
boolean containsOnly = StringUtils.containsOnly("hello", "olleh");
System.out.println(containsOnly);
//출력결과 : true

boolean containsOnly2 = StringUtils.containsOnly("hello", "hell");
System.out.println(containsOnly2);
//출력결과 : false
```

> StringUtils.containsWhitespace (문자열, 비교할문자열)

문자열안에 공백이 포함되어 있으면 'true'를 반환한다.

```
boolean containsWhitespace = StringUtils.containsWhitespace("안녕 하세요");
System.out.println(containsWhitespace);
//출력결과 : true
```

> StringUtils.countMatches (문자열, 비교할문자열)

문자열안에 비교할 문자열과 같은 문자열이 있으면 1을, 없다면 0을 반환한다.

```
int countMatches = StringUtils.countMatches("ABCD", "A");
System.out.println(countMatches);
//출력결과 : 1
int countMatches2 = StringUtils.countMatches("ABCD", "AB");
System.out.println(countMatches2);
//출력결과 : 1
int countMatches3 = StringUtils.countMatches("ABCD", "AC");
System.out.println(countMatches3);
//출력결과 : 0
```

> StringUtils.defaultIfBlank (문자열, 대체문자열)

문자열이 space(띄어쓰기),공백, null 로만 이루어져있을 경우 대체문자열을 반환한다.

```
String defaultIfBlank1 = StringUtils.defaultIfBlank(null, "A");
System.out.println(defaultIfBlank1);
//출력결과 : A
String defaultIfBlank2 = StringUtils.defaultIfBlank("", "A");
System.out.println(defaultIfBlank2);
//출력결과 : A
String defaultIfBlank3 = StringUtils.defaultIfBlank("  ", "A");
System.out.println(defaultIfBlank3);
//출력결과 : A
```

> StringUtils.defaultIfEmpty (문자열, 대체문자열)

문자열이 공백, null 로만 이루어져있을 경우 대체문자열을 반환한다.

```
String defaultIfEmpty1 = StringUtils.defaultIfEmpty(null, "B");
System.out.println(defaultIfEmpty1);
//출력결과 : B
String defaultIfEmpty2 = StringUtils.defaultIfEmpty("", "B");
System.out.println(defaultIfEmpty2);
//출력결과 : B
String defaultIfEmpty3 = StringUtils.defaultIfEmpty("  ", "B");
System.out.println(defaultIfEmpty3);
//출력결과 : 없음
```

> StringUtils.defaultString (문자열, 대체문자열)

문자열이 null 로만 이루어져있을 경우 대체문자열을 반환한다.

```
String defaultString1 = StringUtils.defaultString(null, "C");
System.out.println(defaultString1);
//출력결과 : C
String defaultString2 = StringUtils.defaultString("", "C");
System.out.println(defaultString2);
//출력결과 : 없음
String defaultString3 = StringUtils.defaultString("  ", "C");
System.out.println(defaultString3);
//출력결과 : 없음
```

> StringUtils.deleteWhitespace (문자열)

문자열에 공백이 있을 경우 제거한다.

```
String deleteWhitespace = StringUtils.deleteWhitespace("안녕 하세요");
System.out.println(deleteWhitespace);
//출력결과 : 안녕하세요
```

> StringUtils.difference (문자열1, 문자열2)

두 문자열을 비교하여 문자열2에서 다른 문자부터 이후를 전부 반환한다.

```
String difference = StringUtils.difference("레피아입니다.", "레피아랍니다.");
System.out.println(difference);
//출력결과 : 랍니다.
```

> StringUtils.endsWith (문자열, 비교문자열)

문자열의 마지막 문자와 비교문자열을 비교하여 같다면 'true', 다르면 'false'를 반환한다.

```
boolean endsWith = StringUtils.endsWith("안녕하세요.", ".");
System.out.println(endsWith);
//출력결과 : true
boolean endsWith = StringUtils.endsWith("안녕하세요", ".");
System.out.println(endsWith);
//출력결과 : false
```

> StringUtils.endsWithAny (문자열, 비교문자열1, 비교문자열2, …)

StringUtils.endsWith 와 같지만 비교문자열을 여러개 쓸 수 있다.
여러개의 비교문자열 중 하나라도 문자열의 마지막 문자와 같다면 'true'를 반환한다.

```
boolean endsWithAny = StringUtils.endsWithAny("AAA", "a");
System.out.println(endsWithAny);
//출력결과 : false
boolean endsWithAny = StringUtils.endsWithAny("AAA", "a", "A");
System.out.println(endsWithAny);
//출력결과 : true
```

> StringUtils.endsWithIgnoreCase (문자열, 비교문자열)

StringUtils.endsWith 와 같지만 대소문자를 구분하지 않는다.

```
boolean endsWithIgnoreCase = StringUtils.endsWithIgnoreCase("BBB", "b");
System.out.println(endsWithIgnoreCase);
//출력결과 : true
```

> StringUtils.equals (문자열1, 문자열2)

두 문자열을 비교하여 완전히 같다면 'true'를 반환한다.

```
boolean equals = StringUtils.equals("abcd", "abcd");
System.out.println(equals);
//출력결과 : true
boolean equals = StringUtils.equals("abcd", "a bcd");
System.out.println(equals);
//출력결과 : false
```

> StringUtils.equalsAny (문자열, 비교문자열1, 비교문자열2, …)

StringUtils.equals 와 같지만 비교문자열을 여러개 쓸 수 있다.
여러개의 비교문자열 중 하나라도 문자열과 완전히 같다면 'true'를 반환한다.

```
boolean equalsAny = StringUtils.equalsAny("abcd", "a", "ab", "abcd");
System.out.println(equalsAny);
//출력결과 : true
```

> StringUtils.equalsAnyIgnoreCase (문자열, 비교문자열1, 비교문자열2, …)

StringUtils.equalsAny 와 같지만 대소문자를 구분하지 않는다.

```
boolean equalsAnyIgnoreCase = StringUtils.equalsAnyIgnoreCase("abc",
"abcde", "ABC");
System.out.println(equalsAnyIgnoreCase);
//출력결과 : true
```

> StringUtils.equalsIgnoreCase(문자열,비교할문자열)

문자열과 비교할문자열과 같다면
true을 반환한다.
(대/소문자를 구분안한다)

```
boolean equalsIgnoreCase = StringUtils.equalsIgnoreCase("ss", "sS");
System.out.println(equalsIgnoreCase);
//출력결과 : true
```

> StringUtils.firstNonBlank(문자열,문자열,문자열)

null, 공백, 띄어쓰기를 제외한 첫번째 문자열을 가져온다
StringUtils 3.8 버전부터 사용가능

```
String firstNonBlank = StringUtils.firstNonBlank(null, "", " ", "xyz");
System.out.println(firstNonBlank);
출력결과 : xyz
```

> StringUtils.getCommonPrefix(new String[] {“문자열 ”, “ 문자열 ” })

모든 문자열에서 공통된 첫부분을 출력한다.

```
String getCommonPrefix = StringUtils.getCommonPrefix("sss","ssS","ssa");
System.out.println(getCommonPrefix);
//출력결과 : ss
```

> StringUtils.getDigits(“문자열 ”)

Unicode Number(숫자)만 출력한다.

```
String getDigits = StringUtils.getDigits("123제거된다456");
System.out.println(getDigits);
//출력결과 : 123456
```

> StringUtils.indexOf(“문자열 ” , “ 찾을문자열 ”)

문자열중에 찾을 문자열이
포함된 첫번째 인덱스를 반환한다.

```
int indexOf = StringUtils.indexOf("sS", "S");
System.out.println(indexOf);
//출력결과 : 1
```

> StringUtils.indexOfAny(“문자열”,new String[] {“문자열 ” , “ 문자열 ” })

문자열과 배열을 비교하고(vararg형식)
첫번째 인덱스를 반환한다.

```
int indexOfAny = StringUtils.indexOfAny("안녕하세요",new String[] {"안", "녕"});
System.out.println(indexOfAny);
//출력결과 : 0
```

> StringUtils.indexOfAnyBut(“문자열”,new char[] {'문자', '문자'})

문자열과 문자 배열을 비교 후
배열에 없는 첫번째 문자인덱스를 반환한다.

```
int indexOfAnyBut = StringUtils.indexOfAnyBut("문자열",new char[] {'문', '자'});
System.out.println(indexOfAnyBut);
//출력결과 : 2
```

> StringUtils.indexOfDifference(“문자열 ” , “ 비교문자열 ”);

문자열과 비교문자열을 비교하고 달라지는
부분의 인덱스를 반환한다.

```
int indexOfDifference = StringUtils.indexOfDifference("안녕하세요","안녕하시
오");
System.out.println(indexOfDifference);
```

```
//출력결과 : 3
```

> StringUtils.indexOfIgnoreCase(“문자열 ” , “ 찾을문자열 ”)

문자열중에 찾을 문자열이 포함된 첫번째 인덱스를 반환한다.
(대/소문자를 구분 안한다)

```
int indexOfIgnoreCase = StringUtils.indexOfIgnoreCase("sS", "S");
System.out.println(indexOfIgnoreCase);
//출력결과 : 0
```

Ref

- StringUtils

주레피, eleven, stringutils, commonslang, commonslang3

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:java:commons-lang&rev=1586764102

Last update: **2022/03/10 19:52**