Share

10 文字列操作

Created by GT F
Last updated 2019-08-15

代表的な操作

以下文字列に対して常用な操作を**すべて覚える**必要があります。

メソッド	説明	例
char charAt(int index)	指定されたindexの文字を取得する	System.out.println("abc".charAt(0)) ※ 'a'
int compareTo(String another)	指定された文字列を比較する	System.out.println("abc".compareTo("abc")) ※ 0
int compareToIgnoreCase(String another)	指定された文字列を比較する（大小文字無視）	System.out.println("abc".compareTo("abc")) ※ 0
String concat(String another)	文字列を結合する	System.out.println("abc".concat("123")) ※ abc123
boolean equals(Object another)	文字列を比較する	System.out.println("abc".concat("123")) ※ false
boolean equalsIgnoreCase(String another)	文字列を比較する（大小文字無視）	System.out.println("abc".concat("ABC")) ※ true
byte[] getBytes()	文字列のバイト列を返し	-
int indexOf(String value)	指定された文字のindexを取得する	System.out.println("aaa".indexOf("a")) ※ 0
int lastIndexOf(String value)	指定された文字の最終indexを取得する	System.out.println("acc".lastIndexOf("c")); ※ 2
int length()	文字列の長さを取得する	System.out.println("123".length());

		※ 3
boolean matches(String regex)	正規表現で文字列をマッチするかを確認する	※正規表現は後で
String replace(String old, String newChar)	置換	System.out.println("123".replace("1", "a")); ※ a23
String[] split(String regex)	指定された区切り文字で分割する	String[] vs = "1,".split(",") ※vs.length = 1
String[] split(String regex, int limit)	指定された区切り文字で分割する（最小取得数を制限する）	String[] vs = "1,".split(",", -1) ※vs.length = 2
boolean startsWith(String prefix)	文字列はprefixを開始するかを確認する	System.out.println("123".startsWith("a")); ※ false
boolean endsWith(String suffix)	文字列はsuffixを終了するかを確認する	System.out.println("123".endsWith("a")); ※ false
String subString(int begin)	文字列の一部を取得する	System.out.println("123".subString(1)); ※ 23
String subString(int begin, int end)	文字列の一部を取得する	System.out.println("123".subString(1, 2)); ※ 2
String toLowerCase();	小文字に変換する	System.out.println("A".toLowerCase()); ※ a
String toUpperCase();	大文字に変換する	System.out.println("a".toUpperCase()); ※ A
String trim()	トリム（先頭と末尾の空白を削除）	System.out.println(" A A ".trim()); ※ A A
boolean empty()	空文字列を判断する	System.out.println("".empty()); ※ true

null or Empty

空文字とは長さ=0の文字列です。オブジェクト型（クラス型）は空の場合：変数XXは**null**と呼びます。null変数の属性またメソッドをアクセスする場合、Nullpointerエラーを発生します。

```
String value = ""; // 空文字列
String value1 = " "; // スペース
String value2 = null; // null
System.out.println(value.length()); // 0
System.out.println(value1.length()); // 1
System.out.println(value2.length()); // NG Nullpointer
```

Stringのフォーマット

JavaでStringクラスには静的のメソッドformatを利用して、数字、日付等をフォーマット可能です。キーワード`%`で、Javaにフォーマット指示します。

フォーマティング指示コード

```
%[引数][フラグ][長さ][精度]タイプ
```

※[]では省略可能。

タイプにて、以下種類があります。

タイプ	説明
d	10進数（整数）
f	浮動小数点数
x	16進数
c	文字（char型を数字に変換）

使用例

```
public static void main(String... args) {
    int one = 123456;
    float two = 123456.99999F;
    // one=123,456 two=123,457.00
    System.out.println(String.format("one=%,d two=%,.2f", one, two));
}
```

以下常用のフォーマットを覚えてください。

フォマード	説明	出力
String.format(“%, d”, 1000);	数字1000に「,」を追加する	1,000
String.format(“%05d”, 5)	左0をPaddingする。結果は５桁数になります。	00005

StringBuilder

Stringは文字の配列であり、サイズの変更（文字列内容変更）はかなりメモリコストします。文字列の編集連結等、できるだけStringBuilderを利用してください。一方、あまり変更しない文字列等、普通にStringを利用して問題ないです。

StringBuilder代表的のメソッド

メソッド	説明
append(Object obj)	文字列を追加する

使用例：

```
public static void main(String..args) {
    String abc = "0" + 1 + 2 + "3"; // あまりよくない
    // 以下StringBuilderを用い、文字列を構築する
    StringBuilder sb = new StringBuilder();
    sb.append("0").append(1).append(2).append("3");
    String abc2 = sb.toString();
}
```

質問

質問１：以下３行文字列を「**改行コード**」と「**,**」を分割して２つペットオブジェクトを作成してください。

1. **Pet**クラスを事前作成してください。
2. ペットのタイプは**列挙型**である：type = 0：猫、type = 1：犬

ペット病院用ペットマスタデータは以下通りです。（ファイルから読み取り不要）※トリム注意

```
name, age, type
みみ, 10, 1
レオ, 1, 0
マル,2,0
```

質問２：Windowsは各ファイルの拡張子を持っています。例 「新規ドキュメント.docx」の拡張子は「docx」である。メソッドを作成して、ファイルパス（String型）から拡張子を取得してください。

```
public static String getFileType(String path) {
    // return ??
}
public static void main(String...args) {
    System.out.println(getFileType("c://windows//aa.xlsx")); // xlsx
    System.out.println(getFileType("c://windows//a a.test")); // test
    System.out.println(getFileType("c://windows//aa.bb.docx")); // docx
    System.out.println(getFileType("c://windows//aaaaa")); // Empty
}
```

質問３：Javaには、すべての文字コードはUTF-8であり。日本語（全角文字）の場合１文字 = 3バイト。以下文字列のバイト数と桁数を求めてください。

```
public static void main(String...args) {
    String value = ”abced12345あいうえお１２３４５”;
    // バイト数は？
```

```
    // 桁数は？
}
```

質問４：各現場は文字列に null 又は empty 判断要共通メソッドを実装しています。似ているメソッドを実装してください。

```
public static boolean nullOrEmpty(String value) {
    // value は null or Empty 判断してください。
}
```

質問５：以下仕様のフォーマティング指示コードを作成してくだい。

1. 浮動小数点数
2. 長さは5桁数
3. 精度は小数点後2位

質問６：整数16の16進数をSystem.out.println()してください。

Like Be the first to like this

No labels