**第7章　Stringクラスを使った文字列処理**

この章ではStringクラスを使った文字列処理について説明します。

**Stringクラスを使った文字列処理**

Stringクラスとは、文字列を扱うために使用されているクラスです。文字列は、他の値(※1)とは違いオブジェクトと呼ばれます。オブジェクトの特徴は、データだけでなく文字列を操作するためのメソッド(機能)が含まれています。(※2)

Stringクラスでは、具体的な値(文字列)を **" "** (ダブルクォーテーション)で囲むことで、new演算子を使わなくてもオブジェクトを作ることができます。

(※1) 基本データ型(boolean,byte,char,short,int,long,float,double)

(※2)クラスやメソッドについては第9章で詳しく扱います。

* String str;
* str = "文字列";

**例1:**

* String str;
* str = "";

strには空文字(何もない文字列)が入る。

**例2:**

* String str;
* str = "hoge";

変数strには「hoge」という文字列が入る。

**1.文字列の処理**

* String str;
* str = "abc";
* str = str + "def";(※1)

(※1)3行目は、str += "def"　と書いても同じ意味になる。

strには「abcdef」という文字列が入る。

**例**

[JSample7\_1.java]

* package JSample;
* public class JSample7\_1 {
* public static void main(String[] args) {
* String str = "おはようございます。";
* str += "田中さん。";
* System.out.println(str);
* }
* }

**実行結果**

おはようございます。田中さん。

**2.Stringクラスのメソッド**

文字列が持つインスタンスメソッドは、Stringクラスで定義されています。

Stringクラスにはたくさんのメソッドがありますが、基本的なインスタンスメソッドには次のようなものがあります。

**2-1.equalsメソッド**

equalsメソッドは、「文字列が別の文字列strと等しい時trueを返す」メソッドです。

実際には次の25～29行目のように記述します。

**例**

[JSample7\_2.java]

* package JSample;
* public class JSample7\_2 {
* public static void main(String[] args) {
* String str1 = new String("abc");
* String str2 = new String("abc");
* String str3 = "ab";
* System.out.println("str1 = " + str1);
* System.out.println("str2 = " + str2);
* if (str1 == str2) {
* System.out.println("等しい");
* } else {
* System.out.println("等しくない");
* }
* str3 = str3 + "c";
* System.out.println("str1 = " + str1);
* System.out.println("str3 = " + str3);
* if (str1 == str3) {
* System.out.println("等しい");
* } else {
* System.out.println("等しくない");
* }
* System.out.println("equalsメソッドで比較");
* if (str1.equals(str3)) {
* System.out.println("等しい");
* } else {
* System.out.println("等しくない");
* }
* }
* }

**実行結果**

str1 = abc  
str2 = abc  
等しくない  
str1 = abc  
str3 = abc  
等しくない  
equalsメソッドで比較  
等しい

文字列と文字列を比較する場合、「==」演算子を使うと同じ文字列でも等しいと判定されたり、等しくないと判定されたりします。（8～15行目と17～23行目）

どういった時にどう判定されるのかはJavaの実装にも依存しているので、ここでは同じ文字列が格納さ えているかどうか調べる時には「==」演算子は使わないと覚えておいてください。

**2-2.指定の位置の文字を取得**

charAtメソッドは「先頭から（インデックス）番目の文字を返す」メソッドです。

* String型変数.charAt(インデックス)

**例**

[JSample7\_3.java]

* package JSample;
* public class JSample7\_3 {
* public static void main(String[] args) {
* String str = "日本語";
* char c1 = str.charAt(0);
* char c2 = str.charAt(1);
* char c3 = str.charAt(2);
* System.out.println(c1);
* System.out.println(c2);
* System.out.println(c3);
* }
* }

**実行結果**

日  
本  
語

**2-3.文字列の長さを取得**

lengthメソッドは「文字列の長さ（文字数）を返す」メソッドです。

* String型変数.length()

**例**

[JSample7\_4.java]

* package JSample;
* public class JSample7\_4 {
* public static void main(String[] args) {
* String str = "日本語";
* System.out.println("文字列："+str);
* System.out.println("文字列の長さは" + str.length());
* }
* }

**実行結果**

文字列：日本語  
文字列の長さは3

**2-4.文字列の一部を取得**

substringメソッドは「元の文字列から一部分を取り出す」メソッドです。

取り出す範囲は、下の構文の「beginIndex文字目から、endIndex文字目の前まで」になります。

* String型変数.substring(int beginIndex, int endIndex)

**例**

[JSample7\_5.java]

* package JSample;
* public class JSample7\_5 {
* public static void main(String[] args) {
* String str1 = new String("Hello World!");
* String new\_str1 = str1.substring(2, 5);
* System.out.println(str1 + "のsubstring(2,5)は" + new\_str1 + "です");
* String str2 = new String("こんにちは");
* String new\_str2 = str2.substring(2, 4);
* System.out.println(str2 + "のsubstring(2,4)は" + new\_str2 + "です");
* }
* }

**実行結果**

Hello World!のsubstring(2,5)はlloです  
こんにちはのsubstring(2,4)はにちです

**2-5.大文字と小文字の変換**

toUpperCaseメソッドは「英字を大文字にした文字列を返す」メソッドで、

toLowerCaseメソッドは「英字を小文字にした文字列を返す」メソッドです。

* String型変数.toUpperCase()
* String型変数.toLowerCase()

**例**

[JSample7\_6.java]

* package JSample;
* public class JSample7\_6 {
* public static void main(String[] args) {
* String str1 = new String("Thank You");
* String upper\_str1 = str1.toUpperCase();
* String lower\_str1 = str1.toLowerCase();
* System.out.println("元の文字列 : " + str1);
* System.out.println("大文字へ変換 : " + upper\_str1);
* System.out.println("小文字へ変換 : " + lower\_str1);
* }
* }

**実行結果**

元の文字列 : Thank You  
大文字へ変換 : THANK YOU  
小文字へ変換 : thank you

**2-6.空白を削除する**

trimメソッドは「先頭と末尾の空白を削除した文字列を返す」メソッドです。

* String型変数.trim()

**例**

[JSample7\_7.java]

* package JSample;
* public class JSample7\_7 {
* public static void main(String[] args) {
* String str1 = new String(" Japan　");
* System.out.println("元の文字列「" + str1 + "」");
* System.out.println("文字数は" + str1.length() + "です");
* String new\_str1 = str1.trim();
* System.out.println("空白を取り除いた文字列「" + new\_str1 + "」");
* System.out.println("文字数は" + new\_str1.length() + "です");
* }
* }

**実行結果**

元の文字列「 Japan 」  
文字数は7です  
空白を取り除いた文字列「Japan」  
文字数は5です

**2-7.文字列を分割する**

splitメソッドは、「指定した文字で、指定文字列を分割し、分断した結果を文字列として返す」メソッドです。

* 配列 = String型変数.split(分割文字列)
* String str = "東京,大阪,京都,北海道";
* String[] city = str.split(",");

この例では「東京,大阪,京都,北海道」という文字列が指定した文字列（ｓｔｒ）にあたり、

「,」（コンマ）ごとに分割しそれぞれを一つの文字列として返します。

**例**

[JSample7\_8.java]

* package JSample;
* public class JSample7\_8 {
* public static void main(String[] args) {
* String str = "東京,大阪,京都,北海道";
* String[]city = str.split(",");
* for (int i = 0; i<city.length; i++) {
* System.out.println(i + "番目の要素 = :" + city[i]);
* }
* }
* }

**実行結果**

0番目の要素 = :東京  
1番目の要素 = :大阪  
2番目の要素 = :京都  
3番目の要素 = :北海道

**2-8.文字列を置換する**

**（１）最初に該当した文字列を置換する**

replaceFirstメソッドは「文字列の中から、最初に該当した部分文字列を別の文字列に置換する」メソッドです。

* String型変数.replaceFirst(置換される文字列, 置換文字列)

下の例では8行目がメソッドの仕様部分にあたります。

**例**

[JSample7\_9.java]

* package JSample;
* public class JSample7\_9 {
* public static void main(String[] args) {
* String str = "沖縄出身の私は沖縄の雰囲気が好きです。";
* System.out.println("変換前:");
* System.out.println(str);
* str = str.replaceFirst("沖縄", "東京");
* System.out.println("変換後:");
* System.out.println(str);
* }
* }

**実行結果**

変換前:  
沖縄出身の私は沖縄の雰囲気が好きです。  
変換後:  
東京出身の私は沖縄の雰囲気が好きです。

ここでは、8行目でメソッドを使い、最初の「沖縄」という部分文字列を「東京」という文字列に置換する指示を出しています。

5行目の文字列を見ると、文字列の中に「沖縄」という言葉は2つ含まれています。

replaceFirstメソッドでは、最初に該当した「沖縄」のみを「東京」に置換するため、実行結果の4行目のように出力されます。

**（２）該当する全ての文字列を置換する**

replaceAllメソッドは「文字列の中から、該当した部分文字列を別の文字列に置換する」メソッドです。

* String型変数.replaceAll("置換される文字列", 置換文字列)

下の例では8行目がメソッドの仕様部分にあたります。

**例**

[JSample7\_10.java]

* package JSample;
* public class JSample7\_10 {
* public static void main(String[] args) {
* String str = "沖縄出身の私は沖縄の雰囲気が好きです。";
* System.out.println("変換前:");
* System.out.println(str);
* str = str.replaceAll("沖縄", "東京");
* System.out.println("変換後:");
* System.out.println(str);
* }
* }

**実行結果**

変換前:  
沖縄出身の私は沖縄の雰囲気が好きです。  
変換後:  
東京出身の私は東京の雰囲気が好きです。

この例では(1)とは異なり、5行目の文字列内の「沖縄」と部分文字列をすべて「東京」という文字列に置換するため、実行結果の4行目のように出力されます。

**2-9.文字列の大小を比較する**

compareToメソッドは「辞書順にみて、引数の文字列sより前なら負の数、同じなら0、後なら正の数を返す」メソッドです。

* String型変数1.compareTo(String型引数s)
* String型変数1.compareToIgnoreCase(String型引き数s)

compareToIgnoreCaseメソッドは、文字列の大文字と小文字を区別せずに比較する事ができます。

また、辞書順で引数より前のことを「値が小さい」、引数よりも後のことを「値が大きい」と表現することもあります。

**例**

[JSample7\_11.java]

* package JSample;
* public class JSample7\_11 {
* public static void main(String[] args) {
* String str = "MNO";
* System.out.println(str.compareTo("ABC")); // 結果：12
* System.out.println(str.compareTo("MNO")); // 結果：0
* System.out.println(str.compareTo("XYZ")); // 結果：-11
* System.out.println(str.compareToIgnoreCase("mno")); // 結果：0
* }
* }

**実行結果**

12  
0  
-11  
0

この例では、メソッドのString型変数1の部分は5行目の「MNO」にあたります。

6～8行目でcompareToメソッドを使い、文字列の値を比較しています。

9行目では比較する引数が「mno」と小文字で指定されていますが、compareToIgnoreCaseメソッドを使っているため、大文字と小文字を区別せず比較することができています。

**3.toStringメソッドとは**

toStringメソッドは、引数にString型の文字列に変換したい変数を指定します。戻り値として、String型の文字列を返します。

toString()メソッドとは、

java.lang.Objectクラスで定義されている文字列表現を返すメソッドです。全てのクラスはこのjava.lang.Objectクラスを継承しているため、必然的に

どのクラスでもtoString()メソッドが定義されていることになります。

**例**

[JSample7\_12.java]

* package JSample;
* public class JSample7\_12 {
* public static void main(String[] args) {
* int num1 = 111;
* int num2 = 222;
* System.out.println(num1 + num2);
* String str1 = Integer.toString(num1);
* String str2 = Integer.toString(num2);
* System.out.println(str1 + str2);
* }
* }

**実行結果**

333  
111222

このプログラムでは、+演算子で数値を足した結果と、文字列を結合した結果をそれぞれ表示しています。Integerはint型のラッパークラスで、toString メソッドを呼び出すために使用しています。

**練習**

**問題1**[JEx7\_1.java]

String型で定義された文章中のカンマの取り除いた文章を表示するプログラムを作成せよ 例) abc,123,あいう → abc123あいう

**問題2**[JEx7\_2.java]

文字列の中から1文字を検索するプログラムを作成せよ。 ヒットすればその文字のインデックスを、ヒットしなければ-1を出力すること。 検索文字が複数ある場合は、最も小さいインデックスを出力すること。 例) abcabcからcを検索すると2と出力される。