8　ファイル入出力

6.2節では、プログラムの実行途中で、利用者がキーボードを用いて値を入力する方法について説明しました。しかし、プログラムが必要とする値が非常に多い場合、それを全てキーボード入力するのは手間が大きいでしょう。

また6.3節では、プログラムに用いられる値をディスプレイに出力する方法について説明しました。しかし、プログラムが出力する値が、ディスプレイに入りきらないくらい多い場合、それを全て表示して読むのは難しいでしょう。さらに、その出力した値を明日もう一度使いたい、というときにもディスプレイにしか出力できないのでは不便でしょう。

これらの問題を解決するために、本章ではファイルを操作する方法について解説します。

8.1　ファイル入力

本節では、プログラムが別のファイルの内容を読み込む処理について解説します。なお、Java言語を含む多くの言語では、ファイルの内容を読み込む処理を**ファイル入力**と呼びます。

まずテキストエディタでwh.txtというファイルを新規作成し、その中身を以下のように記述してください。

```
57.0   1.64
59.0   1.66
64.0   1.77
70.0   1.73
79.0   1.81
41.0   1.49
45.0   1.55
47.0   1.59
62.0   1.59
58.0   1.67
```

このファイルは、10人の体重（単位kg）と身長（単位m）を記述したもので、各行の左側に体重、右側に身長が記載されているものとします。

では、このファイルの内容をプログラムで読み込み、10人ぶんのBMIを一気に計算して表示するプログラムを紹介します。

まず以下のコマンドで、BmiMethod.java を BmiFileIO.java というファイルにコピーし、続いてそれをテキストエディタで表示してください。そして BmiFileIO.java の中身を、以下のように書き換えてください。

```
import java.io.File;
import java.io.FileReader;
import java.io.BufferedReader;
import java.util.StringTokenizer;

public class BmiFileIO {
   public static double calculateBmi(double weight, double height) {
     double result = weight / (height * height);
```

```
    return result;
  }

  public static void main(String[] args) {
    double weight = 67.0;
    double height = 1.78;
    String filename = "wh.txt";
    File file = new File(filename);
    try {
      FileReader freader = new FileReader(file);
      BufferedReader breader = new BufferedReader(freader);
      while(breader.ready()) {
        String line = breader.readLine();
        if(line == null) break;
        StringTokenizer token = new StringTokenizer(line);
        if(token.countTokens() < 2) continue;
        String w = token.nextToken();
        String h = token.nextToken();
        weight = Double.parseDouble(w);
        height = Double.parseDouble(h);
        double bmi = calculateBmi(weight, height);
        System.out.println("w=" + weight + " h=" + height + " BMI=" + bmi);
      }
      breader.close();
    } catch(Exception e) {
      e.printStackTrace();
    }
  }
}
```

では、このプログラムの中身について説明しましょう。

まず1〜4行目では、BmiFileIO クラスで用いる別のクラスとして、File, FileReader, BufferedReader, StringTokenizer クラスを import しています。ここでは明示的に 4 つのクラスの名前を記述していますが、7.2 節で説明したように、

```
import java.io.*;
import java.util.*;
```

というように記述しても構いません。

calculateBmi メソッドについては、4.3 節で紹介した BmiMethod クラスと同じですので、ここでは省略します。

続いて 16 行目に、**File** クラスが新しく出てきます。File クラスとは、特定のファイルを扱うためのクラスです。ここでは変数 filename に代入されている、wh.txt というファイル名に対して、File クラスのインスタンスを生成し、これを file という変数に代入します。

続いて 18 行目に、**FileReader** クラスが新しく出てきます。FileReader クラスは、特定のファイルの内容を読むためのクラスです。ここでは変数 file に代入されているファイルに対して、FileReader クラスのインスタンスを生成し、これを freader という変数に代入します。

続いて 19 行目に、BufferedReader クラスが出てきます。BufferedReader クラスの使い方は、6.2 節で紹介した通りです。このクラスは標準入力だけでなく、ファイル入力でも同様に扱えます。

続いて 20 行目に、breader.ready() というメソッドが出てきます。この行では、変数 breader が読み込み可能である状態を確認していて、読み込み可能である限り while 文による反復を実行する、という意味を持ちます。

続いて 21 行目にて、breader.readLine() というメソッドにより、ファイルの内容を 1 行読み、この結果を変数 line に代入します。この結果が null であるときには、ファイルの内容を最後まで読んだと判断し、break 文により while 文の反復を終了します。

続いて 23 行目にて、**StringTokenizer** クラスが新しく出てきます。このクラスは文字列を所定の文字で区切るものです。この場合には、変数 line に代入された文字列を、空白文字で区切ります。例えば、wh.txt の 1 行目

```
57.0 1.64
```

を、57.0 および 1.64 の 2 つの文字列に区切ります。この区切った結果は、nextToken() というメソッドによって、以下のように得ることができます。

- 1 回目の token.nextToken() メソッドの結果として、空白文字で区切った左側（この場合 57.0）を返し、これを String クラスの変数 w に代入します。
- 2 回目の token.nextToken() メソッドの結果として、空白文字で区切った右側（この場合 1.64）を返し、これを String クラスの変数 h に代入します。

ただし注意しないといけない点として、2 つの文字列に分割できない行が混じっていたときには、token.nextToken() メソッドを呼び出すとエラーになってしまいます。これを避けるために、24 行目に if(token.countTokens() < 2) continue; という行が混じっています。これは、2 つ以上の文字列に区切れなければ continue する、という意味です。

続いて 27,28 行目にて、Double.parseDouble() というメソッドが出てきます。このメソッドの意味については 6.1 節を参照してください。

29 行目の calculateBmi() メソッド、および 30 行目の System.out.println() メソッドについては、今まで何度も出てきたので省略します。32 行目の catch 以下についても省略します。

さて、このプログラムを実行してみましょう。以下のように表示されましたでしょうか？

```
w=57.0 h=1.64 BMI=21.192742415229034
w=59.0 h=1.66 BMI=21.410944984758313
w=64.0 h=1.77 BMI=20.428357113217785
w=70.0 h=1.73 BMI=23.38868655818771
w=79.0 h=1.81 BMI=24.114038033027075
w=41.0 h=1.49 BMI=18.467636592946263
w=45.0 h=1.55 BMI=18.730489073881373
w=47.0 h=1.59 BMI=18.59103674696412
w=62.0 h=1.59 BMI=24.524346347059055
w=58.0 h=1.67 BMI=20.796729893506402
```

8.2 ファイル出力

本節では、プログラムが別のファイルに内容を書き出す処理について解説します。なお、Java 言語を含む多くの言語では、ファイルに内容を書き出す処理を**ファイル出力** と呼びます。

ここで、先ほどの BmiFileIO.java を改良して、画面出力している BMI 値を bmi.txt というファイルに出力する、ということを考えます。

では BmiFileIO.java の中身を、以下のように書き換えてください。

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.util.StringTokenizer;

public class BmiFileIO {
   public static double calculateBmi(double weight, double height) {
     double result = weight / (height * height);
     return result;
   }

   public static void main(String[] args) {
     double weight = 67.0;
     double height = 1.78;
     String filename = "wh.txt";
     String filename2 = "bmi.txt";
     File file = new File(filename);
     File file2 = new File(filename2);
     try {
       FileReader freader = new FileReader(file);
       BufferedReader breader = new BufferedReader(freader);
       FileWriter fwriter = new FileWriter(file2);
       BufferedWriter bwriter = new BufferedWriter(fwriter);
       while(breader.ready()) {
         String line = breader.readLine();
         if(line == null) break;
         StringTokenizer token = new StringTokenizer(line);
         if(token.countTokens() < 2) continue;
         String w = token.nextToken();
         String h = token.nextToken();
         weight = Double.parseDouble(w);
         height = Double.parseDouble(h);
         double bmi = calculateBmi(weight, height);
         String line2 = "w=" + weight + " h=" + height + " BMI=" + bmi;
         bwriter.write(line2, 0, line2.length());
         bwriter.flush();
```

```
          bwriter.newLine();
        }
        breader.close();
        bwriter.close();
      } catch(Exception e) {
        e.printStackTrace();
      }
    }
}
```

では、このプログラムの中身について説明しましょう。

まず 1〜6 行目では、BmiFileIO クラスで用いる別のクラスとして、File, FileReader, FileWriter, BufferedReader, BufferedWriter, StringTokenizer クラスを import しています。

19,20 行目に、**File** クラスが 2 回出てきます。1 回目は、変数 filename に代入された wh.txt というファイルのインスタンスを、変数 file に代入します。2 回目は、変数 filename2 に代入された bmi.txt というファイルのインスタンスを、変数 file2 に代入します。前者は読み込むファイル、後者は書き出すファイルですが、ここでは両者を区別せずに、同様に new 命令でインスタンスを生成します。

続いて 24 行目に、**FileWriter** クラスが新しく出てきます。FileWriter クラスは、特定のファイルに内容を書くためのクラスです。ここでは変数 file2 に代入されているファイルに対して、FileWriter クラスのインスタンスを生成し、これを fwriter という変数に代入します。

続いて 25 行目に、**BufferedWriter** クラスが新しく出てきます。BufferedWriter クラスは、標準出力およびファイル出力において、内容を 1 行単位で書き出すなどの機能を有するクラスです。

続いて 36 行目に、String line2= で始まる代入文があります。これは、引用符で囲まれた w=,h=,BMI= といった文字列や、変数 weight, height, bmi の値を、一続きに並べてできる文字列を、変数 line2 に代入する、というものです。この結果として変数 line2 では、BmiFileIO.java が標準出力していたときと同じ内容の文字列を格納することになります。

続いて 37〜39 行目にて、変数 bwriter のメソッドが続々と呼び出されます。まず write メソッドによって、変数 line2 に代入された文字列をファイルに書き出すことが指定されます。続いて flush メソッドによって、変数 bwriter に保持されていた文字列が実際にファイルに書きだされ、そして newLine メソッドによって改行が指定されます。

さて、このプログラムを実行した後に、テキストエディタで bmi.txt の中身を表示してみてください。ファイルの中身は、以下のようになっていますでしょうか？

```
w=57.0 h=1.64 BMI=21.192742415229034
w=59.0 h=1.66 BMI=21.410944984758313
w=64.0 h=1.77 BMI=20.428357113217785
w=70.0 h=1.73 BMI=23.38868655818771
w=79.0 h=1.81 BMI=24.114038033027075
w=41.0 h=1.49 BMI=18.467636592946263
w=45.0 h=1.55 BMI=18.730489073881373
w=47.0 h=1.59 BMI=18.59103674696412
w=62.0 h=1.59 BMI=24.524346347059055
w=58.0 h=1.67 BMI=20.796729893506402
```