プログラミング入門2

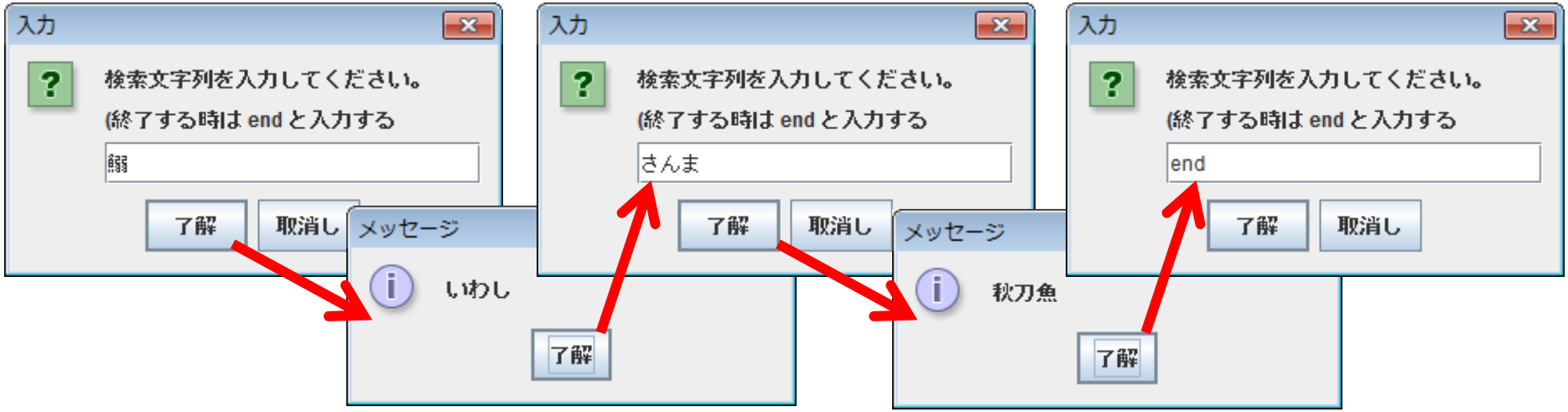
第10回　クラスメソッドとクラス変数

テーマ：クラスメソッドとクラス変数

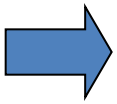
- クラスメソッドとクラス変数
  - クラスメソッド
  - クラス変数
  - 定数
- プログラムの整理とその応用
  - パッケージ
  - アクセスコントロール
  - main メソッドの意味

本日の主な目的(題材)

- クラスメソッドを作成して、インスタンスを配列で管理せよ。(例題23, 問題12)



テーマ：クラスメソッドとクラス変数



- クラスメソッドとクラス変数
  - クラスメソッド
  - クラス変数
  - 定数
- プログラムの整理とその応用
  - パッケージ
  - アクセスコントロール
  - main メソッドの意味

インスタンスメソッドの定義(第5回)

クラスAのインスタンスに保持させたいメソッドは、クラスAに記述する。
クラスAのインスタンスを生成した後に、インスタンスを操作するメソッドだけを
インスタンスメソッドとして定義できる。

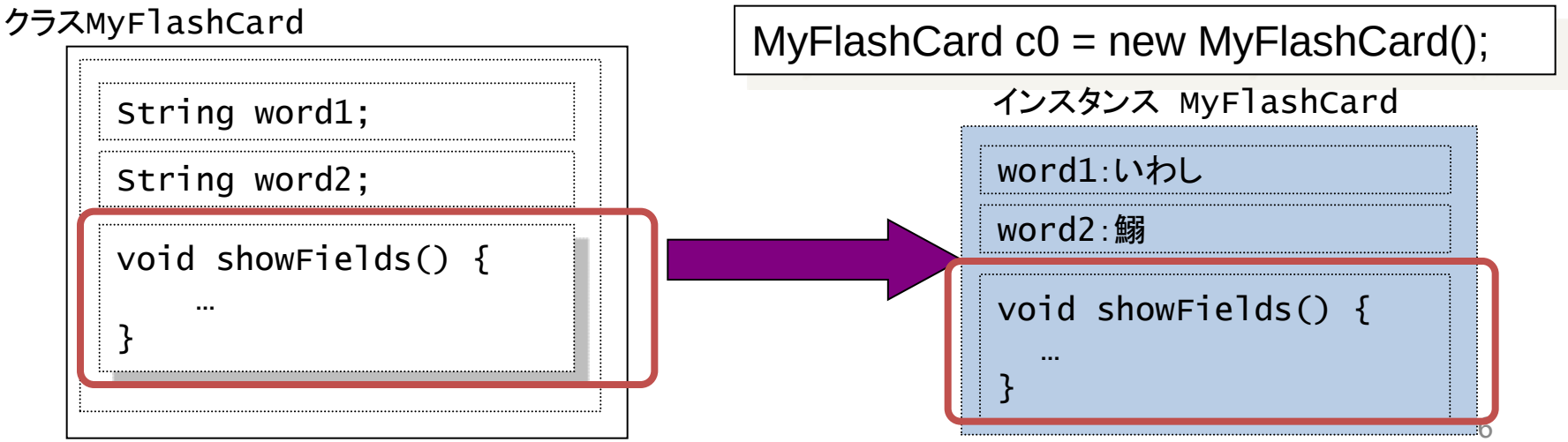
```
public class MyCircle2 {
  int x;
  int y;
  int diameter;

  void drawMyCircle(MyCircle c) {
    Canvas.drawOval(this.x, this.y, this.diameter,
                    this.diameter);
  }
}
```

```
public class クラス名 {
   変数の型 インスタンス変数名;

   返り値の型 インスタンスメソッド名(引数){
      …
   }
}
// 戻り値がない時には、戻り値の型はvoid にする
```

createMyCircle メソッドは、インスタンスを生成する前に呼ばれるメソッドなので、ここに定義できない!!

インスタンスメソッドの使い方

- クラス定義から生成された「ひとつの」インスタンスに対して、操作を行うためのメソッド
- インスタンスを生成した後でないと、呼び出すことはできない
- インスタンスメソッドの中では、対象となる「ひとつの」インスタンスのインスタンス変数に自由にアクセスできる



インスタンスメソッドに書けないもの

- createMyCircle のようなインスタンスを生成するメソッドは書けない
  - なぜならば、このメソッドは、インスタンスを生成する前に呼び出さなければいけないため。
  - そこで、コンストラクタを利用した。
- 複数のインスタンスを配列で管理するメソッドは書けない
  - なぜならば、インスタンスメソッドは「ひとつの」インスタンスの操作だけを対象とするため。
  - 配列で管理するメソッドは、ProductList のような別のクラスを作らないと書くことができなかった。

クラスメソッド

- クラスに関係深い操作であるが、インスタンス生成前の操作や、複数のインスタンスを管理することを目的とした操作を実行するためのメソッド

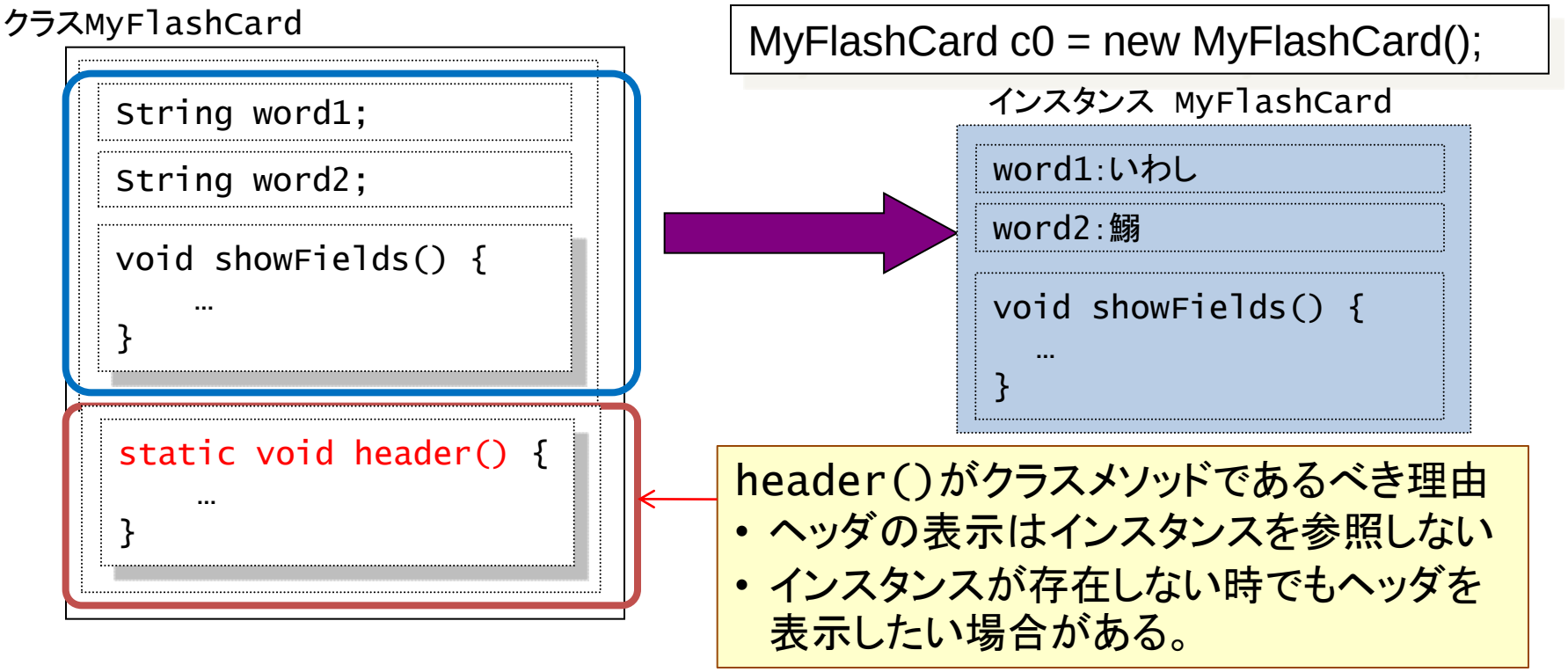
クラスメソッドの例

- インスタンスを生成する時の処理を記述するメソッド
- ひとつのクラスから生成されるインスタンスの集合を管理するメソッド
- クラスについて説明を与えるメソッド

クラスメソッドの定義

- メソッドに、「static 」という修飾子をつける
  - クラスメソッドのことを、static メソッドとも呼ぶ

```
static 返り値の型名 メソッド名(引数, …) { … }
```



■例題13

問題：クラス MyFlashcard のインスタンスメソッド header() をクラスメソッドに書き換え、例題12が空のCSVファイルでも動作できるように書き換えよ。

（クラス名： Ex13FlashCard）

例題13 (MyFlashCard)

```
1  package j2.lesson10;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5
6  public class MyFlashCard {
7      String word1;
8      String word2;
9
10     MyFlashCard(String word1, String word2) {
11         this.word1 = word1;
12         this.word2 = word2;
13     }
14
15     void showFields(int row) {
16         Spreadsheet.setString(row, 0, word1);
17         Spreadsheet.setString(row, 1, word2);
18     }
19
20     static void header() {
21         Spreadsheet.setString(0, 0, "単語1");
22         Spreadsheet.setString(0, 1, "単語2");
23     }
24
25     void showCard() {
26         Canvas.drawString(50, 50, word1);
27         Canvas.waitForCountdown(1000);
28         Canvas.clear();
29
30         Canvas.drawString(50, 50, word2);
31         Canvas.waitForCountdown(1500);
32         Canvas.clear();
33     }
34 }
```

クラスメソッドの呼び出し方

- クラス名に続いて、ピリオドとメソッド名を与えることで、クラスメソッドを呼び出す。

```
クラス名.メソッド名(引数, … ); // 返り値のない場合
変数 =クラス名.メソッド名(引数, … ); // 返り値のある場合
```

具体例:

```
MyFlashCard.header();
MyFlashCard c0 =
    MyFlashCard.createFlashCard(“いわし”, “鰯”);
```

例題13
(Ex13FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex13FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex13FlashCard().start();
    }

    void start() throws IOException {
        int maxRecords = 1000;
        MyFlashCard[] card = new MyFlashCard[maxRecords];

        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/10FlashCard.txt"));
        int index = 0;
        while (index < maxRecords) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            card[index] = new MyFlashCard(word1, word2);
            index++;
        }
        int numRecords = index;

        Spreadsheet.show();

        MyFlashCard.header();
        for (int i = 0; i < numRecords; i++) {
            card[i].showFields(i + 1);
        }
    }
}
```

クラス変数

- クラスメソッド内で利用する変数や、クラスに共通する変数をクラス変数(static 変数とも呼ぶ)という
- インスタンス変数が、インスタンスごとにインスタンスの数だけ生成されるのに対し、クラス変数は、クラスに対して唯一の変数が定義される。

クラス変数の定義

```
static 型名  変数名;
static 型名  変数名 = 初期値;
```

クラス変数の操作

```
変数 = クラス名.クラス変数名; // read操作
クラス名.クラス変数名 = 値;    // write操作
```

■例題21

問題：例題13を拡張して、クラス MyFlashCard2を作成する。このクラス MyFlashCard2 から生成された全てのインスタンスを配列に蓄積管理するクラスメソッド createFlashCard と、それに必要なクラス変数を定義し、CSVファイル(input/07FlashCard.txt)を利用して動作を確認せよ。

クラス変数	初期値	説明
MyFlashCard2[] contents	new MyFlashCard2[100]	生成したインスタンスを格納する配列
int maxRecords	100	格納できる最大数
int index	0	生成したインスタンス数

インスタンス変数	初期値	説明
String word1	無し	カードの表
String word2	無し	カードの裏

（クラス名： MyFlashCard2）

■例題21(続き)

コンストラクタ(引数)	機能
MyFlashCard2(String word1, String word2)	インスタンス変数の値がword1,word2であるMyFlashCard2クラスのインスタンスを作成する。

クラスメソッド

返り値の型	メソッド名(引数)	機能
MyFlashCard2	createFlashCard(String word1, String word2)	インスタンス変数の値がword1,word2であるMyFlashCard2クラスのインスタンスを作成しクラス変数のcontentsに格納する。
void	header()	Spreadsheet の先頭行にヘッダを表示する。

インスタンスメソッド

返り値の型	メソッド名(引数)	機能
void	showFields(int row)	インスタンスの内容をSpreadsheetのrow行に表示する。
void	showCard()	インスタンスのword1、word2の内容を第7回例題11(3)のようにキャンバスに描画する。

（クラス名： Ex11FlashCard2）

例題21 (MyFlashCard2)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import gpjava.Canvas;

public class MyFlashCard2 {
    // Ex21
    static int maxRecords = 100;
    static int index = 0;
    static MyFlashCard2[] contents = new MyFlashCard2[maxRecords];

    String word1;
    String word2;

    MyFlashCard2(String word1, String word2) {
        this.word1 = word1;
        this.word2 = word2;
    }

    // Ex21
    static MyFlashCard2 createFlashCard(String word1, String word2) {
        if(index < maxRecords) {
            MyFlashCard2 card = new MyFlashCard2(word1, word2);
            contents[index] = card;
            index++;
            return card;
        } else {
            return null;
        }
    }

    static void header() {...

    void showFields(int row) {...

    void showCard() {...
}
```

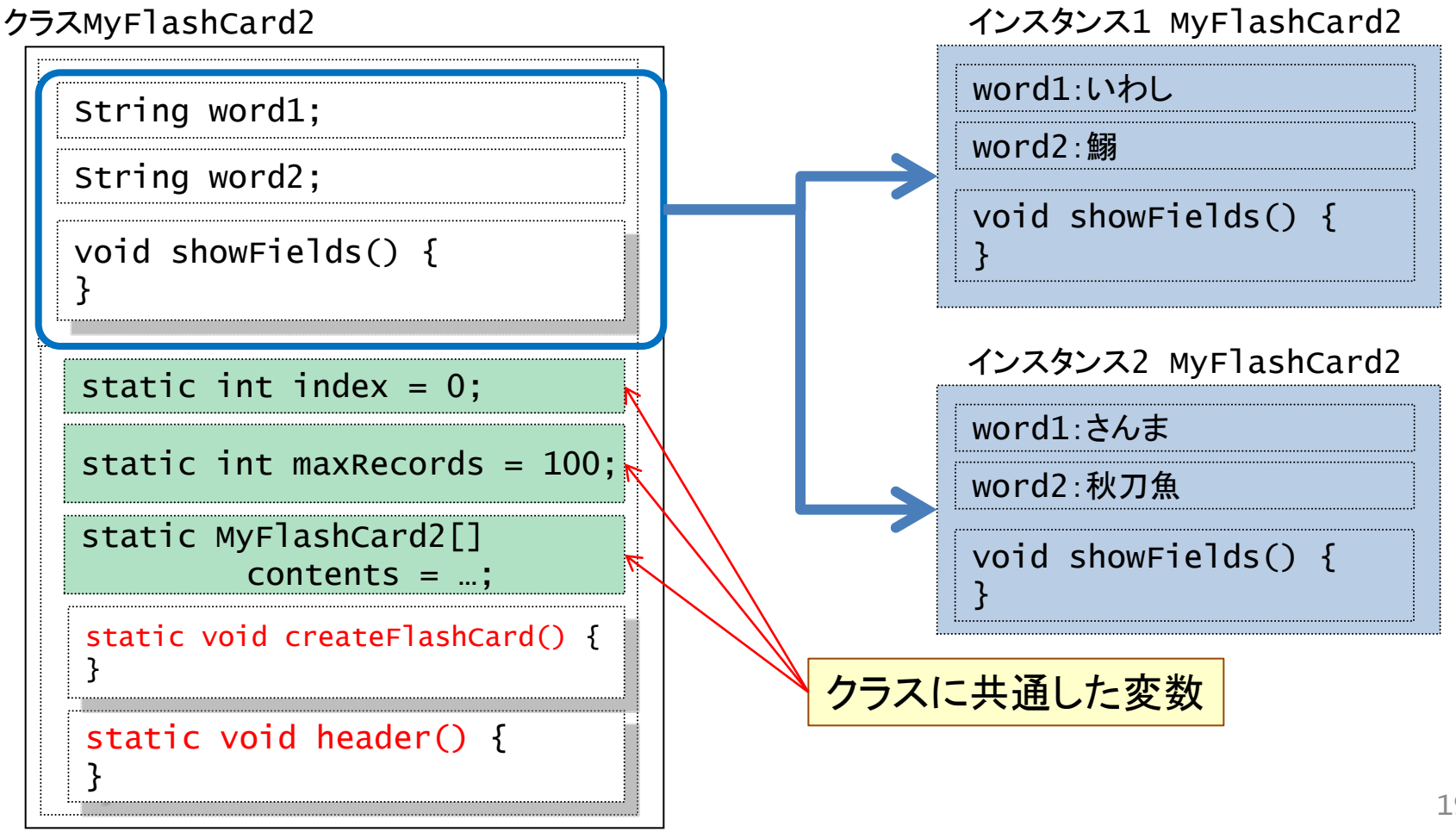
Ex13と同じ

例題21
(Ex21FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex21FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex21FlashCard().start();
    }

    void start() throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/07FlashCard.txt"));
        while (true) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            MyFlashCard2 card = MyFlashCard2.createFlashCard(word1, word2);
            if(card == null) {
                break;
            }
        }

        Spreadsheet.show();

        MyFlashCard2.header();
        for(int i = 0; i < MyFlashCard2.index; i++) {
            MyFlashCard2.contents[i].showFields(i + 1);
        }
    }
}
```

クラス変数とインスタンス変数

- クラス変数はクラスにひとつ作成
- インスタンス変数はインスタンスごとに作成



定数

- クラス変数に、final 修飾子をつけると、クラスに共通の定数を宣言できる
  - 定数名には、変数名と区別するために大文字を使うことが多い。

定数の宣言

```
static final 型名 定数名 = 値;
```

- プログラム中の文字列、意味のある数字などを、まとめて定数として宣言するとよい。

定数の例

```
static final String MESSAGE = “文字列を入力せよ”;
static final int SLEEP_TIME = 1000;
```

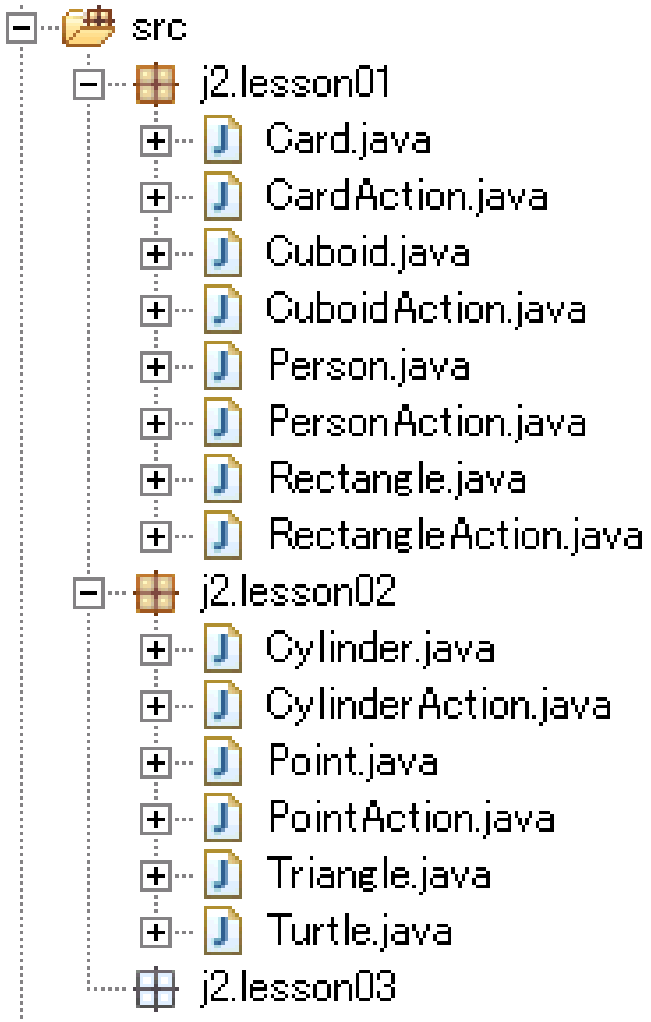
テーマ：クラスメソッドとクラス変数

- クラスメソッドとクラス変数
  - クラスメソッド
  - クラス変数
- プログラムの整理とその応用
  - パッケージ
  - アクセスコントロール
  - mainメソッドの意味

クラスやインスタンスを用いてデータやメソッドを整理することが、オブジェクト指向言語における基本的なプログラムの整理法であるが、より大規模なソフトウェアを開発しやすくするための機構がJava言語には備わっている。

パッケージとは

- クラスをグループ化して整理する仕組み
- j2.lesson02, j2.lesson03がパッケージの例。
- Windowsの「フォルダ」、Linuxの「ディレクトリ」と同じく階層の形式でグループを作成できる。



パッケージ機構の利点と必要性

- 大規模なソフトウェアでは、クラスの数も多くなり、それらを整理する必要がある。
  - 利用するクラスを見つけやすくする。
- クラスの名前が付けやすくなる。（名前の衝突を避けやすくなる）
  - 新たにクラスを作成する場合、パッケージを利用しない場合には、まだ他で使われていないクラス名を付けてやる必要がある。
  - 新たに設定する名前がすでに他で使われてしまっていることを「名前の衝突(conflict)」と言う。

→　別の場所で作成されたクラスライブラリ（後述）を、取り込むことが容易になる。

パッケージの宣言

- パッケージの宣言を行うと、「クラスがどのパッケージに含まれるか」ということをコンピュータに知らせることができる。
- パッケージの宣言はソースファイルの先頭で次のように行う。

```
package パッケージ名;
```

例えば

```
package j2.lesson03;
```

publicとprivate

- 修飾子public を付けた要素（クラス、メソッド、コンストラクタ、インスタンス変数など）は他のパッケージ**からでも (全てのクラスから) 参照することができる**。
  - 全国の電話帳に名前を公開しているようなもの。publicとして宣言されたものには全国（全クラス）からアクセスがくる可能性がある。
- 修飾子private は逆に、それを指定するクラス内だけで（私的に）インスタンス変数やメソッドなどを用いるために付ける。すなわち、これを付けた要素は**他のクラスから参照できなくなる**。

アクセスコントロールの範囲

- Javaでは、アクセスコントロールの範囲（どの範囲で要素を公開するか）として次の４種類が用意されている。
  - private　：　それを指定したクラス内からのみ利用可能
  - package private(修飾子なし) : それを指定したパッケージ内からのみ利用可能
  - protected : それを指定したパッケージ内のクラスおよびサブクラスからのみ利用可能
  - public　: 全てのクラスから利用可能

アクセスコントロールの利用

- アクセスコントロールは、メソッドやインスタンス変数などを、どの範囲で公開するか（あるいは秘密にするか）を指定する。

- privateは、「見せたくない情報を隠す」ために用いる。利用者にとっては、本質的に必要でないことが明らかになるのでプログラムがしやすくなる。

- インスタンス変数をprivateとして指定すると、「外から書き換えてほしくない情報を利用者から勝手に書き換えられないようにする」ことができる。

■例題30

問題：j2.lesson07 で作成した PaintedCar に、インスタンスメソッドとコンストラクタの修飾子には public、インスタンス変数の修飾子には private を付与した PublicPaintedCar クラスを、j2.lesson07 の中に作成せよ。

インスタンス変数

修飾子	型名	変数名
private	int	bodyWidth, bodyHeight, wheelDiameter, red, green, blue

コンストラクタ

修飾子	コンストラクタ名(引数)
public	PublicPaintedCar(int bodyWidth, int bodyHeight, int wheelDiameter, int red, int green, int blue)

インスタンスメソッド

修飾子	返り値の型	メソッド名(引数)
public	void	showCar(int row), header(),drawCar(int x, int y)

（クラス名： `j2.lesson07.PublicPaintedCar`）

例題30 (PublicPaintedCar)

j2.lesson07 の中に作成

```
1  package j2.lesson07;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5
6  public class PublicPaintedCar {
7      private int bodyWidth;
8      private int bodyHeight;
9      private int wheelDiameter;
10     private int red;
11     private int green;
12     private int blue;
13
14     public PublicPaintedCar(int width, int height, int diameter, int red, int green, int blue) {..
22
23     public void showCar(int row) {..
31
32     public void header() {..
40
41     public void drawCar(int x, int y) {..
52 }
```

private なインスタンス変数

public なコンストラクタ

public なインスタンスメソッド

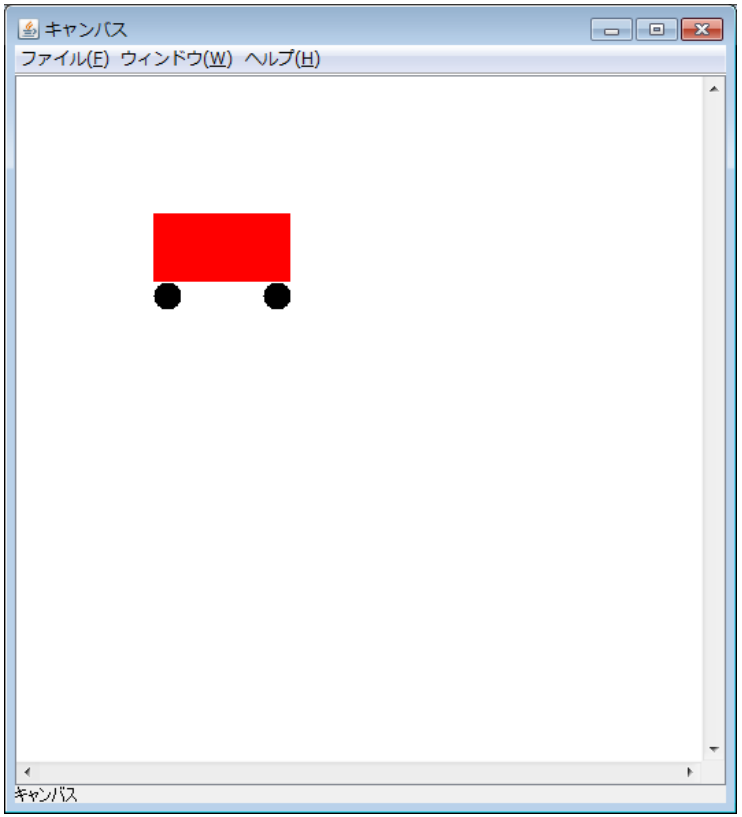
コンストラクタ、メソッドの中身は PaintedCar と同じ

他のパッケージにあるクラスを使用する

- 他のパッケージにあるクラスを使用する場合、クラス名を「**パッケージ名.クラス名**」で指定してやる必要がある。
- 「パッケージ名.クラス名」の形で指定したクラス名を**完全限定名** (または**完全修飾名**) と呼ぶ。また、パッケージ名を省略した「クラス名」の形で指定したクラス名を**単純名**と呼ぶ。

■例題31

問題：j2.lesson07 のPublicPaintedCar クラスを完全修飾のクラス名で利用して、車を表示するプログラムを作成せよ。



（クラス名： Ex31DrawCar）

例題31(Ex31DrawCar)

```
1  package j2.lesson10;
2
3  import gpjava.Canvas;
4
5  public class Ex31DrawCar {
6      public static void main(String[] args) {
7          new Ex31DrawCar().start();
8      }
9
10     void start() {
11         Canvas.show();
12         j2.lesson07.PublicPaintedCar car
13             = new j2.lesson07.PublicPaintedCar(100, 50, 20, 255, 0, 0);
14         car.drawCar(100, 100);
15     }
16 }
```

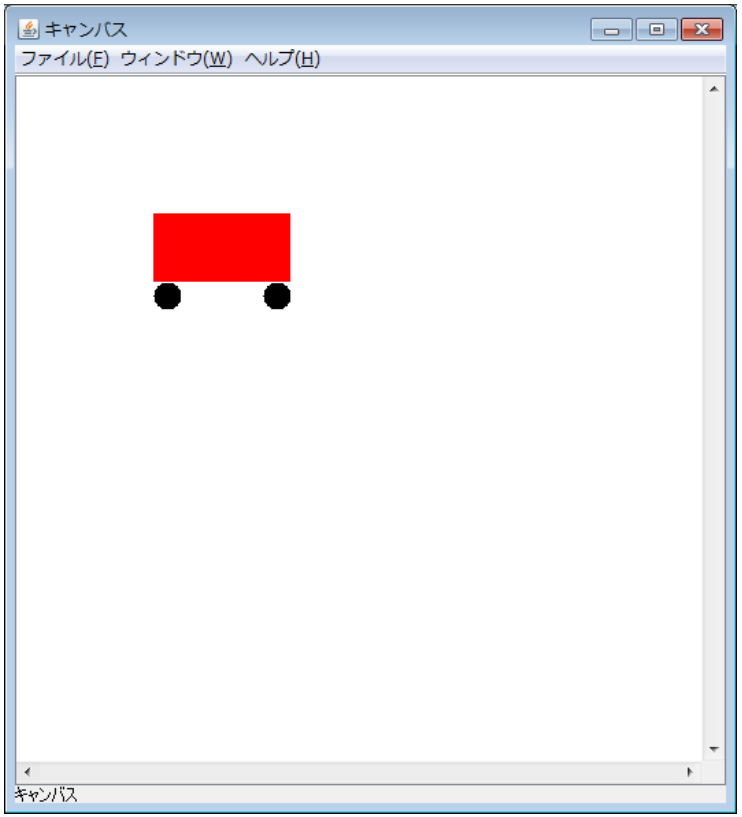
import 機能

```
import 完全限定名;
```

- import 宣言を行うことによって、対象のクラスを単純名でアクセスできるようになる。
- 「import パッケージ名.*; 」と書くことによって、対象のパッケージに含むクラスを全て単純名でアクセスできるようにする機能もある。

■例題32

問題：j2.lesson07 のPublicPaintedCar クラスを import して、クラス名を利用して、車を表示するプログラムを作成せよ。



（クラス名： Ex32DrawCar）

例題32(Ex32DrawCar)

```
1  package j2.lesson10;
2
3  import gpjava.Canvas;
4  import j2.lesson07.PublicPaintedCar;
5
6  public class Ex32DrawCar {
7      public static void main(String[] args) {
8          new Ex32DrawCar().start();
9      }
10
11     void start() {
12         Canvas.show();
13         PublicPaintedCar car = new PublicPaintedCar(100, 50, 20, 255, 0, 0);
14         car.drawCar(100, 100);
15     }
16 }
```

暗黙の import

```
package j2.lesson14;

public class FullyQualifiedJavaLang {
    public static void main(java.lang.String[] args) {
        java.lang.System.out.println(java.lang.Math.sqrt(2.0));
    }
}
```

- Systemクラス, Stringクラス, Mathクラスはそれぞれ Java 標準ライブラリ（後述）の java.lang パッケージ内に格納されている。

```
package j2.lesson14;

public static void main(String[] args) {
    System.out.println(Math.sqrt(2.0));
}
```

- 本来であれば、このように単純名で書く場合は、import java.lang.*;のようなインポート宣言が必要であるが、java.lang パッケージ内のクラスはよく使用されるため、**import しなくても単純名を使用できる**ことになっている。

public static void main() の意味

- これまで、実行クラスには、必ず public static void main() メソッドを習慣として書いてきた。
  - これの意味は、パッケージ外に公開する返り値のないクラスメソッド main である。
  - 他のクラスメソッドと意味的な違いはなく、プログラム起動時に、外部から呼ばれる点だけが特殊である。

- new クラス名().start() の意味は、クラスのインスタンスを生成し、そのインスタンスに対して、インスタンスメソッド start()を呼び出すことである。
  - 実は、起動クラスは、インスタンス変数をもたないクラスであり、変数を持たないインスタンスを生成した後、インスタンスメソッド start() を呼び出していた。

- start() メソッドを呼び出すルールは、プログラミング入門1のルールであって、Javaのルールではない。
  - main メソッドの中に、直接プログラムを書いてもよい。ただし、クラスメソッドは、インスタンスを指定しないかぎり、(同じクラスの)クラスメソッドしか呼び出せないことに注意。

例題32(Ex32DrawCar)

```java
package j2.lesson10;

import gpjava.Canvas;
import j2.lesson07.PublicPaintedCar;

public class Ex32DrawCar {
    public static void main(String[] args) {
        new Ex32DrawCar().start();
    }

    void start() {
        Canvas.show();
        PublicPaintedCar car = new PublicPaintedCar(100, 50, 20, 255, 0, 0);
        car.drawCar(100, 100);
    }
}
```

例題33(1) (Ex33DrawCar)

```java
package j2.lesson10;

import gpjava.Canvas;
import j2.lesson07.PublicPaintedCar;

public class Ex33DrawCar {
    public static void main(String[] args) {
        Canvas.show();
        PublicPaintedCar car = new PublicPaintedCar(100, 50, 20, 255, 0, 0);
        car.drawCar(100, 100);
    }
}
```

まとめ：クラスメソッドとクラス変数

- クラスメソッドとクラス変数
  - クラスメソッド
  - クラス変数
  - 定数
- プログラムの整理とその応用
  - パッケージ
  - アクセスコントロール
  - main メソッドの意味

本日の例題と問題

- MyFlashCard の拡張(クラスメソッドの演習)
  - Ex10, Ex11, Ex12, Ex13, Ex21, Ex22, Ex23, Q11, Q12
- PublicPaintedCar (public, private 修飾子の利用)
  - Ex30, Ex31, Ex32, Ex33
- Prefecture の拡張
  - Q21, Q22, Q23*

(Ex:例題, Q:問題, *は少し手間のかかる問題)

各自に適した順番で解けばよいが、上記の順番が自然な流れとなるよう構成されている。

例題集

パッケージ「j2.lesson10」を作成する。

パッケージやクラスの作成，実行の仕方の説明は省略する。
作り方を忘れた場合は過去のスライドや
http://java2010.cis.k.hosei.ac.jp/01/material-01/
を参考にせよ

■例題10

問題：第7回例題11で作成したフラッシュカード(単語帳)のクラスMyFlashCardを作成せよ。

インスタンス変数	初期値	説明
String word1	無し	カードの表
String word2	無し	カードの裏

コンストラクタ(引数)	機能
MyFlashCard(String word1, String word2)	インスタンス変数の値がword1,word2であるMyFlashCardクラスのインスタンスを作成する。

インスタンスメソッド

返り値の型	メソッド名(引数)	機能
void	showFields(int row)	インスタンスの内容をSpreadsheetのrow行に表示する。
void	header()	Spreadsheet の先頭行にヘッダを表示する。
void	showCard()	インスタンスのword1、word2の内容を第7回例題11(3)のようにキャンバスに描画する。

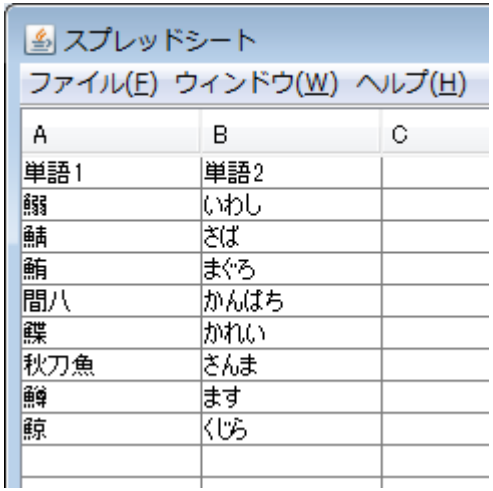
（クラス名： MyFlashCard）

例題10 (MyFlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import gpjava.Canvas;

public class MyFlashCard {
    String word1;
    String word2;

    MyFlashCard(String word1, String word2) {
        this.word1 = word1;
        this.word2 = word2;
    }

    void showFields(int row) {
        Spreadsheet.setString(row, 0, word1);
        Spreadsheet.setString(row, 1, word2);
    }

    void header() {
        Spreadsheet.setString(0, 0, "単語1");
        Spreadsheet.setString(0, 1, "単語2");
    }

    void showCard() {
        Canvas.drawString(50, 50, word1);
        Canvas.waitForCountdown(1000);
        Canvas.clear();

        Canvas.drawString(50, 50, word2);
        Canvas.waitForCountdown(1500);
        Canvas.clear();
    }
}
```

■例題11

問題：第7回例題21(3)で作成したプログラムと同じ内容の Ex11FlashCard を作成し、CSVファイル(inputFiles/07Flashcard.txt)を読み込み、フラッシュカードの内容を表示せよ。



（クラス名： Ex11FlashCard）

例題11
(Ex11FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex11FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex11FlashCard().start();
    }

    void start() throws IOException {
        int maxRecords = 1000;
        MyFlashCard[] card = new MyFlashCard[maxRecords];

        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/07FlashCard.txt"));
        int index = 0;
        while (index < maxRecords) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            card[index] = new MyFlashCard(word1, word2);
            index++;
        }
        int numRecords = index;

        Spreadsheet.show();

        card[0].header();
        for (int i = 0; i < numRecords; i++) {
            card[i].showFields(i + 1);
        }
    }
}
```

■例題12

問題：例題11のCSVファイルの指定を
InputFiles/10Flashcard.txt に変更してファイルを読み込み、フラッシュカードの内容を表示せよ。

ただし、事前に、InputFiles/10Flashcard.txt にはからのファイルを作成しておく。

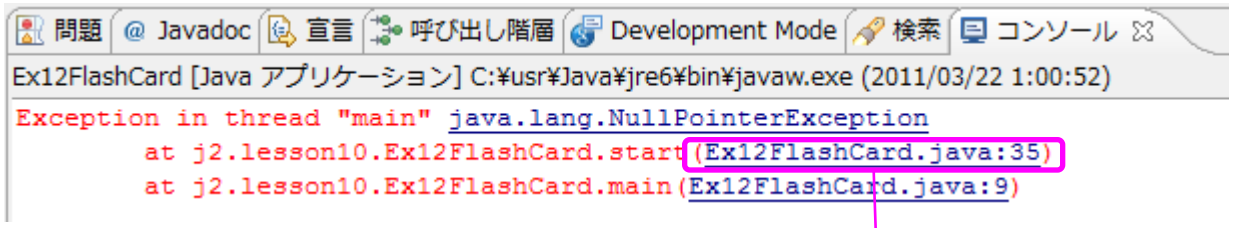
何が起きるか。

（クラス名： Ex12FlashCard）

例題12
(Ex12FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex12FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex12FlashCard().start();
    }

    void start() throws IOException {
        int maxRecords = 1000;
        MyFlashCard[] card = new MyFlashCard[maxRecords];

        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/10FlashCard.txt"))
        int index = 0;
        while (index < maxRecords) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            card[index] = new MyFlashCard(word1, word2);
            index++;
        }
        int numRecords = index;

        Spreadsheet.show();

        card[0].header();
        for (int i = 0; i < numRecords; i++) {
            card[i].showFields(i + 1);
        }
    }
}
```

例題12の結果



配列が空なので、card[0] は null になり、header を表示できない

```
33          Spreadsheet.show();
34
35          card[0].header();
36          for (int i = 0; i < numRecords; i++) {
37              card[i].showFields(i + 1);
38          }
```

■例題13

問題：クラス MyFlashcard のインスタンスメソッド header() をクラスメソッドに書き換え、例題12が空のCSVファイルでも動作できるように書き換えよ。

（クラス名： Ex13FlashCard）

例題13 (MyFlashCard)

```
1  package j2.lesson10;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5
6  public class MyFlashCard {
7      String word1;
8      String word2;
9
10     MyFlashCard(String word1, String word2) {
11         this.word1 = word1;
12         this.word2 = word2;
13     }
14
15     void showFields(int row) {
16         Spreadsheet.setString(row, 0, word1);
17         Spreadsheet.setString(row, 1, word2);
18     }
19
20     static void header() {
21         Spreadsheet.setString(0, 0, "単語1");
22         Spreadsheet.setString(0, 1, "単語2");
23     }
24
25     void showCard() {
26         Canvas.drawString(50, 50, word1);
27         Canvas.waitForCountdown(1000);
28         Canvas.clear();
29
30         Canvas.drawString(50, 50, word2);
31         Canvas.waitForCountdown(1500);
32         Canvas.clear();
33     }
34 }
```

例題13
(Ex13FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex13FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex13FlashCard().start();
    }

    void start() throws IOException {
        int maxRecords = 1000;
        MyFlashCard[] card = new MyFlashCard[maxRecords];

        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/10FlashCard.txt"));
        int index = 0;
        while (index < maxRecords) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            card[index] = new MyFlashCard(word1, word2);
            index++;
        }
        int numRecords = index;

        Spreadsheet.show();

        MyFlashCard.header();
        for (int i = 0; i < numRecords; i++) {
            card[i].showFields(i + 1);
        }
    }
}
```

■例題21

問題：例題13を拡張して、クラス MyFlashCard2を作成する。このクラス MyFlashCard2 から生成された全てのインスタンスを配列に蓄積管理するクラスメソッド createFlashCard と、それに必要なクラス変数を定義し、CSVファイル(input/07FlashCard.txt)を利用して動作を確認せよ。

クラス変数	初期値	説明
MyFlashCard2[] contents	new MyFlashCard2[100]	生成したインスタンスを格納する配列
int maxRecords	100	格納できる最大数
int index	0	生成したインスタンス数

インスタンス変数	初期値	説明
String word1	無し	カードの表
String word2	無し	カードの裏

（クラス名： MyFlashCard2）

■例題21(続き)

コンストラクタ(引数)	機能
MyFlashCard2(String word1, String word2)	インスタンス変数の値がword1,word2であるMyFlashCard2クラスのインスタンスを作成する。

クラスメソッド

返り値の型	メソッド名(引数)	機能
MyFlashCard2	createFlashCard(String word1, String word2)	インスタンス変数の値がword1,word2であるMyFlashCard2クラスのインスタンスを作成しクラス変数のcontentsに格納する。
void	header()	Spreadsheet の先頭行にヘッダを表示する。

インスタンスメソッド

返り値の型	メソッド名(引数)	機能
void	showFields(int row)	インスタンスの内容をSpreadsheetのrow行に表示する。
void	showCard()	インスタンスのword1、word2の内容を第7回例題11(3)のようにキャンバスに描画する。

（クラス名： MyFlashCard2）

例題21 (MyFlashCard2)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import gpjava.Canvas;

public class MyFlashCard2 {
    // Ex21
    static int maxRecords = 100;
    static int index = 0;
    static MyFlashCard2[] contents = new MyFlashCard2[maxRecords];

    String word1;
    String word2;

    MyFlashCard2(String word1, String word2) {
        this.word1 = word1;
        this.word2 = word2;
    }

    // Ex21
    static MyFlashCard2 createFlashCard(String word1, String word2) {
        if(index < maxRecords) {
            MyFlashCard2 card = new MyFlashCard2(word1, word2);
            contents[index] = card;
            index++;
            return card;
        } else {
            return null;
        }
    }

    static void header() {...

    void showFields(int row) {...

    void showCard() {...
}
```

Ex13と同じ

例題21
(Ex21FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex21FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex21FlashCard().start();
    }

    void start() throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/07FlashCard.txt"));
        while (true) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            MyFlashCard2 card = MyFlashCard2.createFlashCard(word1, word2);
            if(card == null) {
                break;
            }
        }

        Spreadsheet.show();

        MyFlashCard2.header();
        for(int i = 0; i < MyFlashCard2.index; i++) {
            MyFlashCard2.contents[i].showFields(i + 1);
        }
    }
}
```

■例題22

問題：例題21を拡張して、クラス MyFlashCard2に蓄積管理されたインスタンスを全て Spreadsheet に表示する showContents を作成し、動作を確認せよ。

新しく追加するクラスメソッド

返り値の型	メソッド名(引数)	機能
void	showContents()	蓄積管理されたインスタンスをヘッダ行と共に、Spreadsheet に表示する。

（クラス名： MyFlashCard2）

例題22 (MyFlashCard2)

Ex21と同じ

Ex13と同じ

```
1  package j2.lesson10;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5
6  public class MyFlashCard2 {
7      // Ex21
8      static int maxRecords = 100;
9      static int index = 0;
10     static MyFlashCard2[] contents = new MyFlashCard2[maxRecords];
11
12     String word1;
13     String word2;
14
15     MyFlashCard2(String word1, String word2) {..
19
20     // Ex21
21     static MyFlashCard2 createFlashCard(String word1, String word2) {..
31
32     // Ex22
33     static void showContents() {
34         header();
35         for(int i = 0; i < index; i++) {
36             contents[i].showFields(i + 1);
37         }
38     }
39
40     static void header() {..
44
45     void showFields(int row) {..
49
50     void showCard() {..
59 }
```

例題22
(Ex22FlashCard)

```java
package j2.lesson10;

import gpjava.Spreadsheet;
import java.io.*;

public class Ex22FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex22FlashCard().start();
    }

    void start() throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/07FlashCard.txt"));
        while (true) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            MyFlashCard2 card = MyFlashCard2.createFlashCard(word1, word2);
            if(card == null) {
                break;
            }
        }

        Spreadsheet.show();
        MyFlashCard2.showContents();
    }
}
```

■例題23

問題：例題22を拡張して、クラス MyFlashCard2に蓄積管理されたインスタンスから、key 文字列に対するペアとなる文字列を返却する get メソッドと、このgetメソッドを使って、ダイアログで検索機能を提供する getFromInput メソッド作成し、動作を確認せよ。

新しく追加するクラスメソッド

返り値の型	メソッド名(引数)	機能
String	get(String key)	蓄積管理されたインスタンスの中から引数 key となる文字列を探し、ペアとなる文字列を返却する。key と比較する文字列は、word1 と word2 の両者であり、ペアとなる文字列とは、比較した文字列の他方の文字列である。
void	getFromInput()	ダイアログから入力したキーワードとペアになる単語を get() メソッドを使って返却し、ダイアログに表示する。end が入力するまで続ける。

（クラス名： MyFlashCard2）

例題23 (MyFlashCard2)

Ex21と同じ

Ex22と同じ

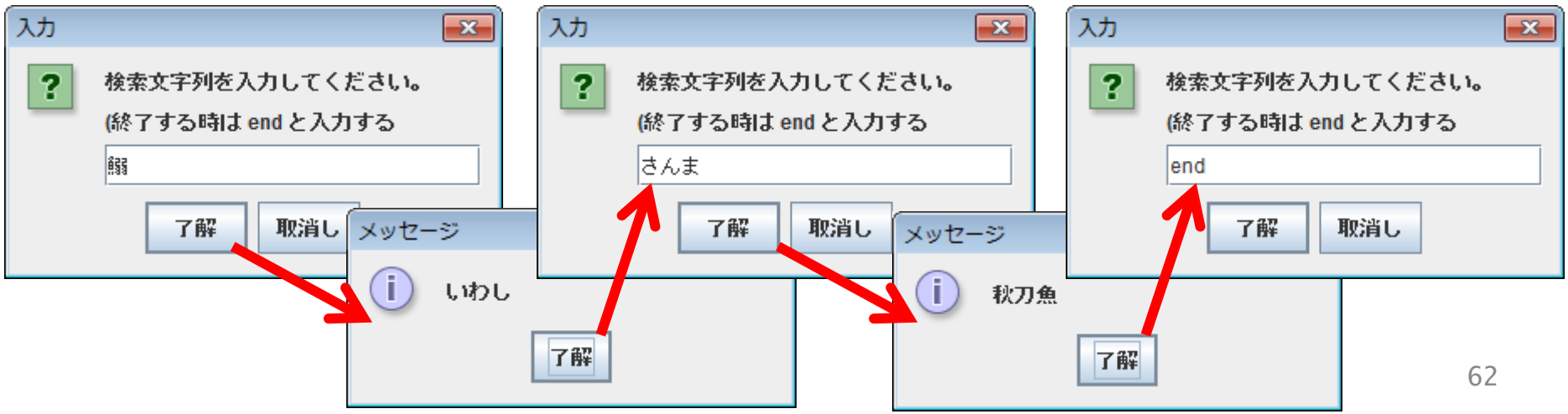
```
1  package j2.lesson10;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5  import javax.swing.JOptionPane;
6
7  public class MyFlashCard2 {
8      // Ex21
9      static int maxRecords = 100;
10     static int index = 0;
11     static MyFlashCard2[] contents = new MyFlashCard2[maxRecords];
12
13     String word1;
14     String word2;
15
16     MyFlashCard2(String word1, String word2) {
20
21     // Ex21
22     static MyFlashCard2 createFlashCard(String word1, String word2) {
32
33     // Ex22
34     static void showContents() {
40
41     // Ex23
42     static String get(String key) {
43         for(int i = 0; i < index; i++) {
44             MyFlashCard2 card = contents[i];
45             if(card.word1.equals(key)) {
46                 return card.word2;
47             } else if(card.word2.equals(key)) {
48                 return card.word1;
49             }
50         }
51         return null;
52     }
53
```

次ページへ続く

例題23(続き)
(MyFlashCard2)

```
54    static void getFromInput() {
55        while(true) {
56            String message = "検索文字列を入力してください。\n(終了する時は end と入力する";
57            String input = JOptionPane.showInputDialog(message);
58
59            if(input.equals("end")) {
60                break;
61            }
62
63            String word = get(input);
64            if(word == null) {
65                message = "該当する単語は存在しません";
66            } else {
67                message = word;
68            }
69            JOptionPane.showMessageDialog(null, message);
70        }
71    }
72
73    static void header() {
77
78    void showFields(int row) {
82
83    void showCard() {
92  }
```

Ex13と同じ



例題23
(Ex23FlashCard)

```java
package j2.lesson10;

import java.io.*;

public class Ex23FlashCard {

    public static void main(String[] args) throws IOException {
        new Ex23FlashCard().start();
    }

    void start() throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(
                "inputFiles/07FlashCard.txt"));
        while (true) {
            String line = reader.readLine();
            if (line == null) { // null なら終了
                break;
            }
            String[] strs = line.split(",");
            String word1 = strs[0]; // 第1列(カード表)
            String word2 = strs[1]; // 第2列(カード裏)

            MyFlashCard2 card = MyFlashCard2.createFlashCard(word1, word2);
            if(card == null) {
                break;
            }
        }

        MyFlashCard2.getFromInput();
    }
}
```

■例題30

問題：j2.lesson07 で作成した PaintedCar に、インスタンスメソッドとコンストラクタの修飾子には public、インスタンス変数の修飾子には private を付与した PublicPaintedCar クラスを、j2.lesson07 の中に作成せよ。

インスタンス変数

修飾子	型名	変数名
private	int	bodyWidth, bodyHeight, wheelDiameter, red, green, blue

コンストラクタ

修飾子	コンストラクタ名(引数)
public	PublicPaintedCar(int bodyWidth, int bodyHeight, int wheelDiameter, int red, int green, int blue)

インスタンスメソッド

修飾子	返り値の型	メソッド名(引数)
public	void	showCar(int row), header(),drawCar(int x, int y)

（クラス名： `j2.lesson07.PublicPaintedCar`）

例題30 (PublicPaintedCar)

j2.lesson07 の中に作成

```
1  package j2.lesson07;
2
3  import gpjava.Spreadsheet;
4  import gpjava.Canvas;
5
6  public class PublicPaintedCar {
7      private int bodyWidth;
8      private int bodyHeight;
9      private int wheelDiameter;
10     private int red;
11     private int green;
12     private int blue;
13
14     public PublicPaintedCar(int width, int height, int diameter, int red, int green, int blue) {..
22
23     public void showCar(int row) {..
31
32     public void header() {..
40
41     public void drawCar(int x, int y) {..
52 }
```

private なインスタンス変数

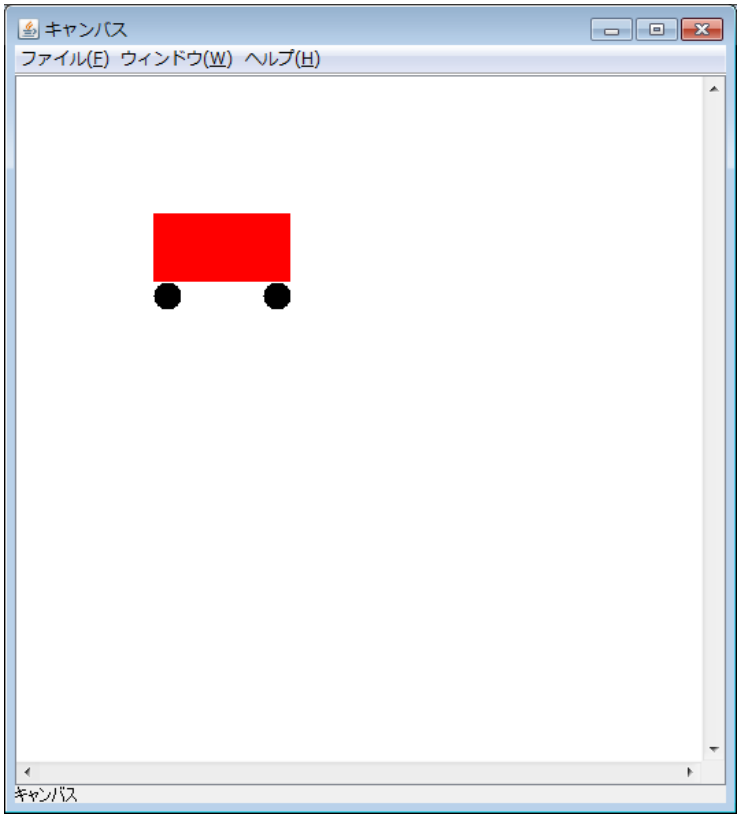
public なコンストラクタ

public なインスタンスメソッド

コンストラクタ、メソッドの中身は PaintedCar と同じ

■例題31

問題：j2.lesson07 のPublicPaintedCar クラスを完全修飾のクラス名で利用して、車を表示するプログラムを作成せよ。



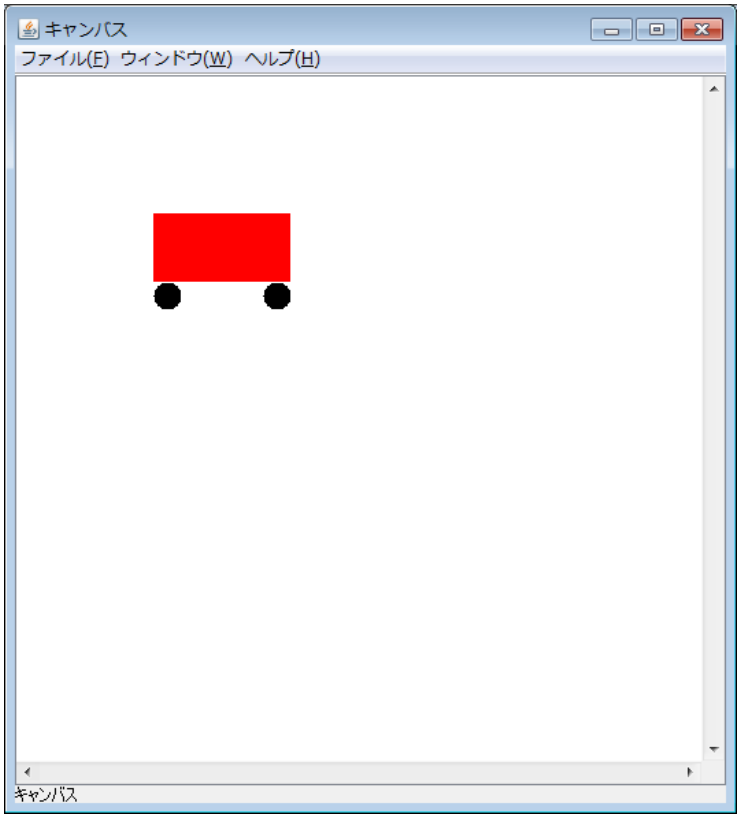
（クラス名： Ex31DrawCar）

例題31(Ex31DrawCar)

```
1  package j2.lesson10;
2
3  import gpjava.Canvas;
4
5  public class Ex31DrawCar {
6      public static void main(String[] args) {
7          new Ex31DrawCar().start();
8      }
9
10     void start() {
11         Canvas.show();
12         j2.lesson07.PublicPaintedCar car
13             = new j2.lesson07.PublicPaintedCar(100, 50, 20, 255, 0, 0);
14         car.drawCar(100, 100);
15     }
16 }
```

■例題32

問題：j2.lesson07 のPublicPaintedCar クラスを import して、クラス名を利用して、車を表示するプログラムを作成せよ。



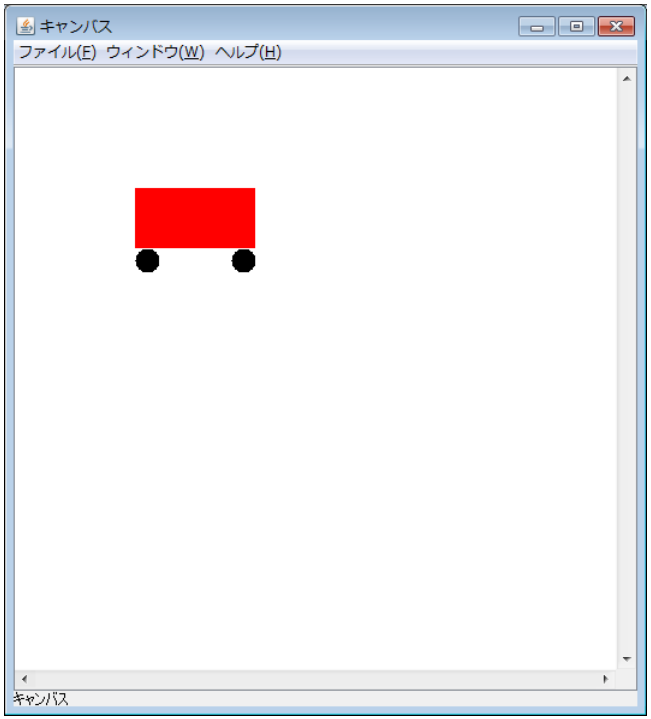
（クラス名： Ex32DrawCar）

例題32(Ex32DrawCar)

```
1  package j2.lesson10;
2
3  import gpjava.Canvas;
4  import j2.lesson07.PublicPaintedCar;
5
6  public class Ex32DrawCar {
7      public static void main(String[] args) {
8          new Ex32DrawCar().start();
9      }
10
11     void start() {
12         Canvas.show();
13         PublicPaintedCar car = new PublicPaintedCar(100, 50, 20, 255, 0, 0);
14         car.drawCar(100, 100);
15     }
16 }
```

■例題33(1)

問題：j2.lesson07 のPublicPaintedCar クラスをフルパスのクラス名で利用して、車を表示するプログラムを作成せよ。ただし、start() メソッドは作成せずに、全てを main メソッド内に記述してみよ。



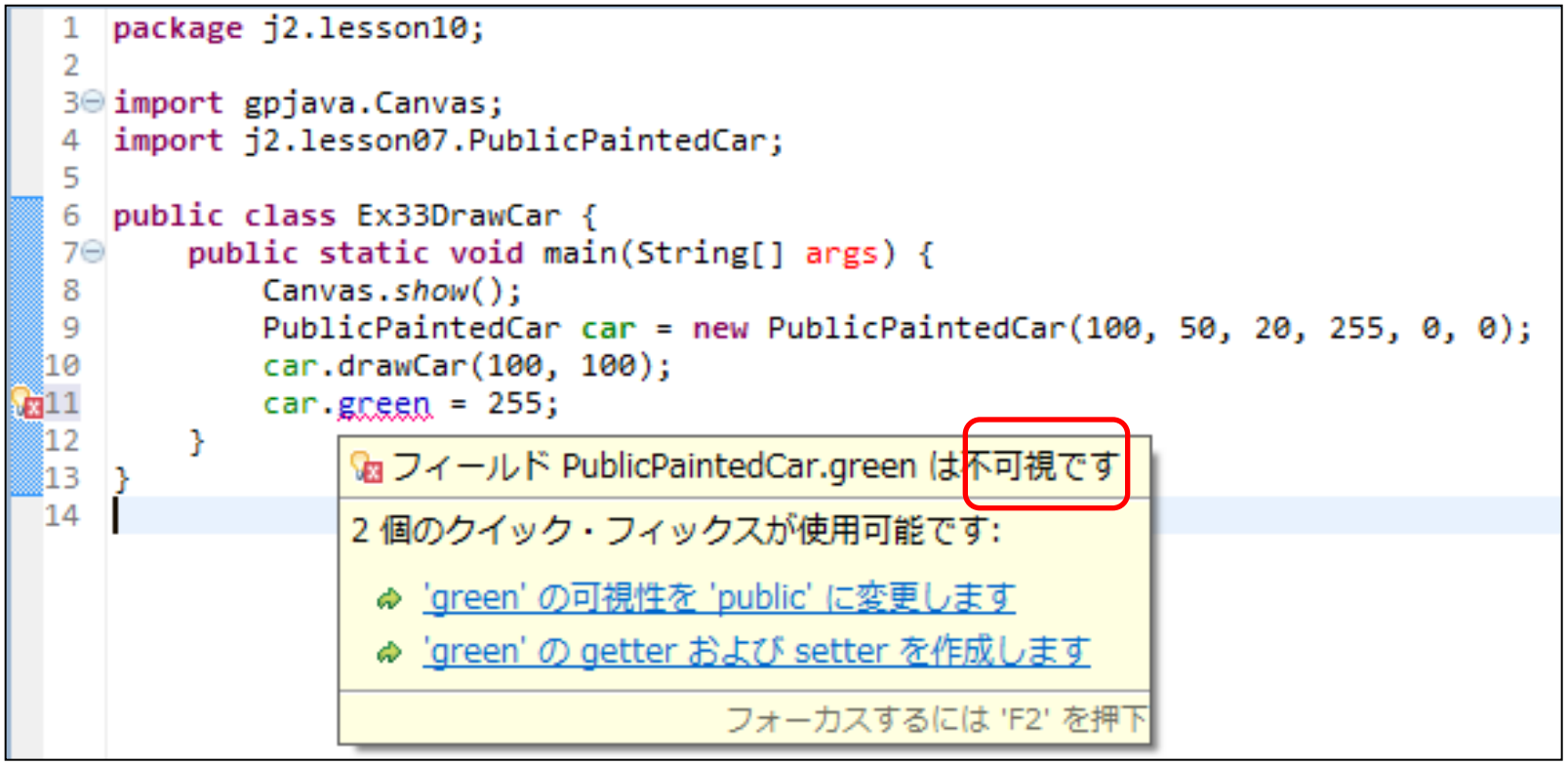
（クラス名： Ex33DrawCar）

例題33(1) (Ex33DrawCar)

```
1  package j2.lesson10;
2
3  import gpjava.Canvas;
4  import j2.lesson07.PublicPaintedCar;
5
6  public class Ex33DrawCar {
7      public static void main(String[] args) {
8          Canvas.show();
9          PublicPaintedCar car = new PublicPaintedCar(100, 50, 20, 255, 0, 0);
10         car.drawCar(100, 100);
11     }
12 }
```

■例題33(2)

問題：例題33のEx33DrawCar で、main の最後の行に、
car.green = 255; と追加したら何が起こるか確認せよ。



（クラス名： Ex33DrawCar）