

# Java講座

---

~第2回~

情報科学部コンピュータ科学科  
2年 竹中 優

# 今回の内容

- ◆ Eclipseの便利なショートカットキー
- ◆ メソッド
  - ◆ メソッドの定義
  - ◆ メソッドのオーバーロード
  - ◆ メソッドを再帰呼び出し
  - ◆ 可変長引数を持つメソッド

# Eclipseの便利なショートカットキー

## ◆ オススメ順に記述

ショートカットキー	効果
Ctrl+F11	プログラムの実行
F11	デバッグの開始
Ctrl+Space	補完(予測リストの表示)
Alt+Shift+r	リファクタリング(変数、メソッド、クラスの名前変更)
Ctrl+Shift+o	自動でimport文の挿入
Fn+Ctrl+矢印上下	開いているプログラムの表示切り替え
Alt+矢印上下	プログラムの選択中の部分を上下に移動させる
Ctrl+/ または Ctrl+7	選択中の行とカーソル行のコメントアウト

# Eclipseの便利なショートカットキー

## ◆ オススメ順に記述

ショートカットキー	効果
Alt+Shift+l	ローカル変数の抽出
Alt+Shift+m	メソッドの抽出
Ctrl+l	エラー箇所ですぐにどうすればよいかの候補を表示
Ctrl+Shift+f	コードの自動フォーマット(インデントなど)
Alt+Shift+z	選択箇所を色々な構文で囲む

# Eclipseの便利なショートカットキー

- ◆ mainと入力→ [Ctrl+Space]→Enter→mainメソッドの自動入力
- ◆ [Ctrl+左右矢印キー]→単語単位で移動
- ◆ [Fn+左右矢印キー]→記述してある部分の最初と最後に移動
- ◆ [Alt+上下矢印キー]→現在の(ドラッグしている)行を上下に移動
- ◆ [Ctrl+Shift+o]→自動インポート
- ◆ [Ctrl+Shift+f]→フォーマット(自動でインデント)
- ◆ [Shift+Alt+r]→リファクタリング(名前変更)
- ◆ [Ctrl+1]→色々。使ううちに覚えましょう。

---

# メソッド

# メソッド

メソッドとは、何回も使うような一連の命令などをまとめておくためのもの。  
あとは、数学の関数と同じ意味。

例えば、 $f(x) = 2x + 5$  という関数。

Javaに置き換えると、 $f$ という名前の関数が引数 $x$ を受け取り、その値を2倍し、5を足した値を戻り値として返す(`return`する)。

# メソッド

メソッド(C言語では関数という)には、大きく分けて4種類ある。

- ◆ 引数あり、戻り値あり(void以外の型)
- ◆ 引数なし、戻り値あり(void以外の型)
- ◆ 引数あり、戻り値なし(void型)
- ◆ 引数なし、戻り値なし(void型)

ちなみにメソッドの引数は255個が上限らしい...

# メソッド

## メソッドの定義フォーマット

```
戻り値の型 メソッド名([引数の型 引数名, ...]){  
    処理;  
}
```

例

- ◆ `void voidTypeMethod(int a){ }//returnしなくても良い`
- ◆ `int intTypeMethod(){return (int型の値);}`
- ◆ `String[] stringArrayTypeMethod(int length){  
 return (String型配列); }`

# メソッド

## メソッド呼び出しフォーマット

```
メソッド名(引数);
```

### 例

- ◆ `voidTypeMethod(100);`//戻り値がないため変数に格納できない
- ◆ `int a = intTypeMethod();`
- ◆ `stringArrayTypeMethod(10);`//戻り値を格納しなくても良い

# メソッド

```
void start(){
    sayHello();
    getInt();
    String str = “メソッドから取得:” + getString();
}
void sayHello(){
    System.out.println(“Hello world!”);
    return;//省略可
}
int getInt(){
    .....
    return int型の値;
}
String getString(){
    .....
    return “文字列”;
}
```

# メソッドのオーバーロード

# メソッド

```
void print(String str){
    System.out.println("文字列:" + str);
}
void print (int num){
    System.out.println("整数:" + num);
}
void print (String str, double dbl){
    System.out.println("文字列:" + str);
    System.out.println("小数:" + dbl);
}
void print (){
    System.out.println("Hello world.");
}
```

上記のプログラム中で、

**show(100);**

と記述したら、どの**show**メソッドが呼ばれるだろう？

# メソッド

もちろん、`show(int num)`が呼ばれる。

メソッド名が同じでも引数の型、あるいは個数などが違っていれば、つまり呼び出す側が区別できれば、問題なく定義できる。

このようにメソッドを同じ名前前で定義することをメソッドのオーバーロードという。

オーバーロード.....over-loading(多重定義)

# メソッド

- ◆ オーバーロードはクラスを扱う際にも使うので、ここで覚えておこう。

サンプルコード: `Sample_overloading.java`

# デフォルト引数

- ◆ C言語などでは、関数の引数にデフォルト値を与えることができる。
- ◆ Javaでは、そのような機能はないので、デフォルト値を与えたい場合は引数ありとなしのメソッドをオーバーロードする必要がある。

```
void print(){
    print("Hello world!");
}
void print(String str){
    System.out.println(str);
}
```

# 問題1

- ◆ 次のメソッドを定義しなさい。1つの引数を受け取り、その引数の型の名前を返すメソッド  
String getType(引数)

例えば、

```
int a = 10;
```

```
String type = getType(a); //type="int"となる
```

```
String s = "ssssss";
```

```
String type2=getType(s); //type2="String"
```

# メソッドを再帰呼び出し

# メソッド

- ◆ メソッドの再帰呼び出しについて  
あるメソッド中において、メソッドが自分自身を  
呼び出すこと。  
これが出来るとスマート。

わかりにくいので、  
次頁の例を見てみよう。

# メソッド

```
void countdown(int n){  
    System.out.println(n);  
    if(1 <= n)  
        countdown(n-1);  
}
```

上記のcountdown()メソッドは第1回資料問題1の解答の1つである。  
引数に5を渡すと、

5

4

3

2

1

0

とコンソールに出力される。

サンプルコード: `Sample_recursion01.java`

# メソッド

```
int fibo(int n){  
    if(1 < n)  
        return fibo(n-1) + fibo(n-2);  
    else  
        return 1;  
}
```

- ◆ フィボナッチ数列の第n項の値を返すメソッドfibo()
- ◆  $a = \{1, 1, 2, 3, 5, 8, 13, 21, 34, \dots\}$
- ◆ 数列aの第n項の値は  $a_n = a_{n-1} + a_{n-2}$

サンプルコード: Sample\_recursion02.java

# 問題2

- ◆  $n$ の階乗を求めるメソッド`factrial()`を再帰呼び出しを用いて、定義せよ。  
ただし、引数は`int`型の $n$ のみ、戻り値の型を`int`とする。  
また、 $n < 0$ の場合は-1を返す。

これ以降、余裕のある人向け

# 可変長引数を持つメソッド

# メソッド

- ◆ 可変長引数はメソッドに同じ型の引数をいくつか渡したい場合などに使用する。
- ◆ メソッド側で配列を引数の型とするのと同じ意味になる。  
(むしろ制限が多くなるが便利な場合もある。)

# メソッド

- ◆ 例えば、  
渡した引数(整数値)のうちの最大値を返すメソッドなどを定義したいとする。  
比較する値が2つの場合、  
`int Math.max(値1, 値2)`  
というメソッドが使用できるが、3つ以上の場合は?

# メソッド

可変長引数を持つメソッドの記述例(前頁の最大値計算メソッド)

```
int max(int v1, int v2, int ... values){
    int max = Math.max(v1, v2);
    for(int i = 0; i < values.length; i++){
        if(values[i] > max)
            max = values[i];
    }
    return max;
}
```

- ◆ メソッド中では、可変長引数の値を配列のように使用する。
- ◆ 可変長引数の後ろに他の引数は定義できない(前はOK)。

サンプルコード: `Ans_variableArgs.java`

終わり