



# 簡単なJAVAプログラム

オブジェクト指向プログラミング特論

只木進一:工学系研究科

# JAVAの特徴

- オブジェクト指向プログラミング言語
  - すべてクラスとして記述する
  - ファイル名そのものがクラス名になる
  - 文法はC++とほぼ同じ
- プラットフォーム依存が無い
  - JavaVMが動けば、どのOSでも動作
  - Windows、iOS、UNIX、Linuxなど
  - Jarファイルを持って、他のOSで実行可能
- 多数のライブラリが言語とともに配布
  - GUI、FileIO、Thread、DBなどなど



# 実習

- SimpleSampleプロジェクトの生成
- firstSampleパッケージの生成
- NoClass.javaの作成
- バブルソートの作成と実行



# 「構築」または「ビルド」

- コンパイル
  - Program.java → Program.class
- jarファイルへ
  - \*.class →プロジェクト名.jar
  - 必要なclassファイルをまとめる
- プロジェクトフォルダを見てみよう
- 保存時の自動コンパイル
- 整合性の確保のために
  - 「消去してビルド」
  - .classファイルを全て消して、再コンパイルとjar作成



# 実行

- NetBeansの中からの実行
  - プロジェクトのデフォルトの開始クラスから
  - `main()`のあるクラスを指定



## 実行:2

- コマンドラインから
- `java -cp クラスパス クラス名 オプション`
  - クラスパス:jarファイル
  - クラス名:main()メソッドのあるクラス名
- `java -jar jarファイル`
  - mainが定義されている場合



# クラスを使わない例

- メインのクラスしか無い例題
  - NoClass.java
- プログラムの開始は
  - `public static void main(String[] args)`
  - **main**に処理の詳細を書かないこと
- コンストラクタ: クラス名と同じメソッド
  - ここがinstance生成の場所



# 簡単な説明

- C++と共通な部分
  - 式の表記、forやwhile、ifやswitch
  - メソッドの書き方
- C++と違う部分
  - pointerが無い
  - クラスインスタンスは全て参照
  - headerファイルが無い
  - デストラクタが無い
    - 自動ガベージコレクション
  - 配列もクラスオブジェクト
  - 文字列はStringというクラス





## ○ package

- クラスをグループ化・階層化
- fieldへのアクセス制限で有効
  - 何も指定しないと、同一package内に対してpublic、他のpackageに対してprivate

## ○ static宣言

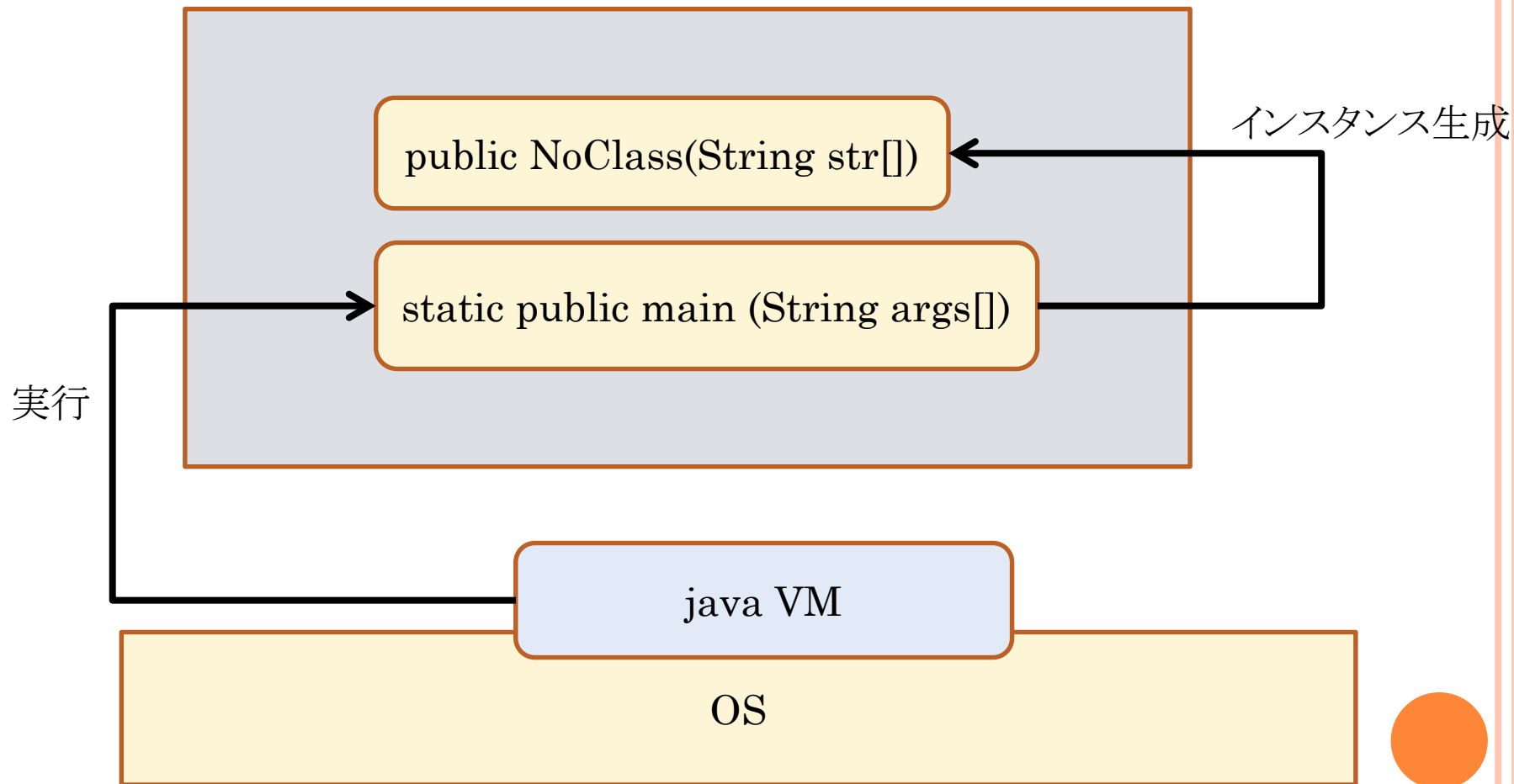
- クラスに属するメソッドやフィールド
  - インスタンスを作らなくても存在する
  - インスタンスを複数作っても、一つしか無い

## ○ import

- ライブラリの名前空間を導入




# 実行の仕組み



# 課題

- 以下のEntryクラスの要素をscoreでソートする方法を考えよ。

```
public class Entry {  
    final private String name;//名前  
    final private int score;//点数  
  
    public Entry(String name, int score) {  
        this.name = name;        this.score = score;}  
  
    public Entry(final Entry entry) {  
        this.name = entry.getName();    this.score = entry.getScore();    }  
  
    public String getName() { return name;}  
    public int getScore() {return score;}  
    public String toString() {return getName() + ":" + getScore();}  
}
```





NoClass.java

```
/*
 * 属するパッケージを宣言
 */
package firstSample;

/**
 *
 * @author tadaki
 */
public class NoClass {
    //データを保存する整数配列

    private final int data[];

    /**
     * コンストラクタ
     *
     * @param data データ
     */
    public NoClass(int data[]) {
        this.data = data;
    }

    /**
     * ソート
     *
     * @return
     */
    public int[] sort() {
        return bubble(data);
    }

    private int [] bubble(int d[]) { //泡立ち法
        for (int j = d.length - 1; j >= 1; j--) { //後ろからループを回す
            for (int i = 0; i < j; i++) {
                if (d[i] > d[i + 1]) { //順序が逆の場合
                    int c = d[i];
                    d[i] = d[i + 1];
                    d[i + 1] = c;
                }
            }
        }
        return d;
    }

    /**
```

NoClass.java

```

    *   ここから実行が始まる
    *
    *   @param args the command line arguments
    */
    public static void main(String[] args) {
        int data[] = {4, 2, 5, 8, 3, 1};
        //インスタンスの生成
        NoClass noClass = new NoClass(data);
        int d[] = noClass.sort();
        for (int i=0; i<d.length; i++) {
            System.out.print(d[i]);
            System.out.print(" ");
        }
        System.out.println();
    }
}
```