プログラム設計

http://bit.ly/design4d

UMLの概要とクラス図

後期 第1週

2019/9/30

UMLとは

UML（Unified Modeling Language）は、OMGという団体により標準化された、ソフトウェアの仕様や設計を図として表現するために使用する言語です。

「クラス」の概念を用いたオブジェクト指向開発技法で利用されます。

現在はUML 2が利用されています。

ダイアグラムの種類

UML 2では、13種類の図（ダイアグラム）が定められています。

本講義では、その中でも使用頻度の高いダイアグラムを数種類を扱います。

参考文献

- 竹政照利「はじめて学ぶUML」(ナツメ社)
- マーチン・ファウラー「UMLモデリングのエッセンス」(翔泳社)
- 河村一樹「ソフトウェア工学入門」(近代科学社)
- 高橋 麻奈「やさしいJava オブジェクト指向編」(SBクリエイティブ)

まずは・・・

一番利用頻度の高いクラス図から学びます

クラス図とは、クラスの構造（フィールド, メソッドなど）とクラス間の関係（集約, 継承など）を表すためのダイアグラムです。

今回 → クラスの構造
次回 → クラスの間の関係

【課題の準備】

演習室で作業する前に、以下のコマンドを
入れるだけで準備が完了する

```
$   mygitclone4d   「自分のGitHubユーザ名」
$   cd prog4d-(ユーザ名)
$   ./myconf
```

※本体をシャットダウンするまでは、
上記「mygitclone」と「myconf」の設定は有効です

【課題の準備】

以下の流れで、課題のプログラムを作るためのフォルダを準備しましょう。

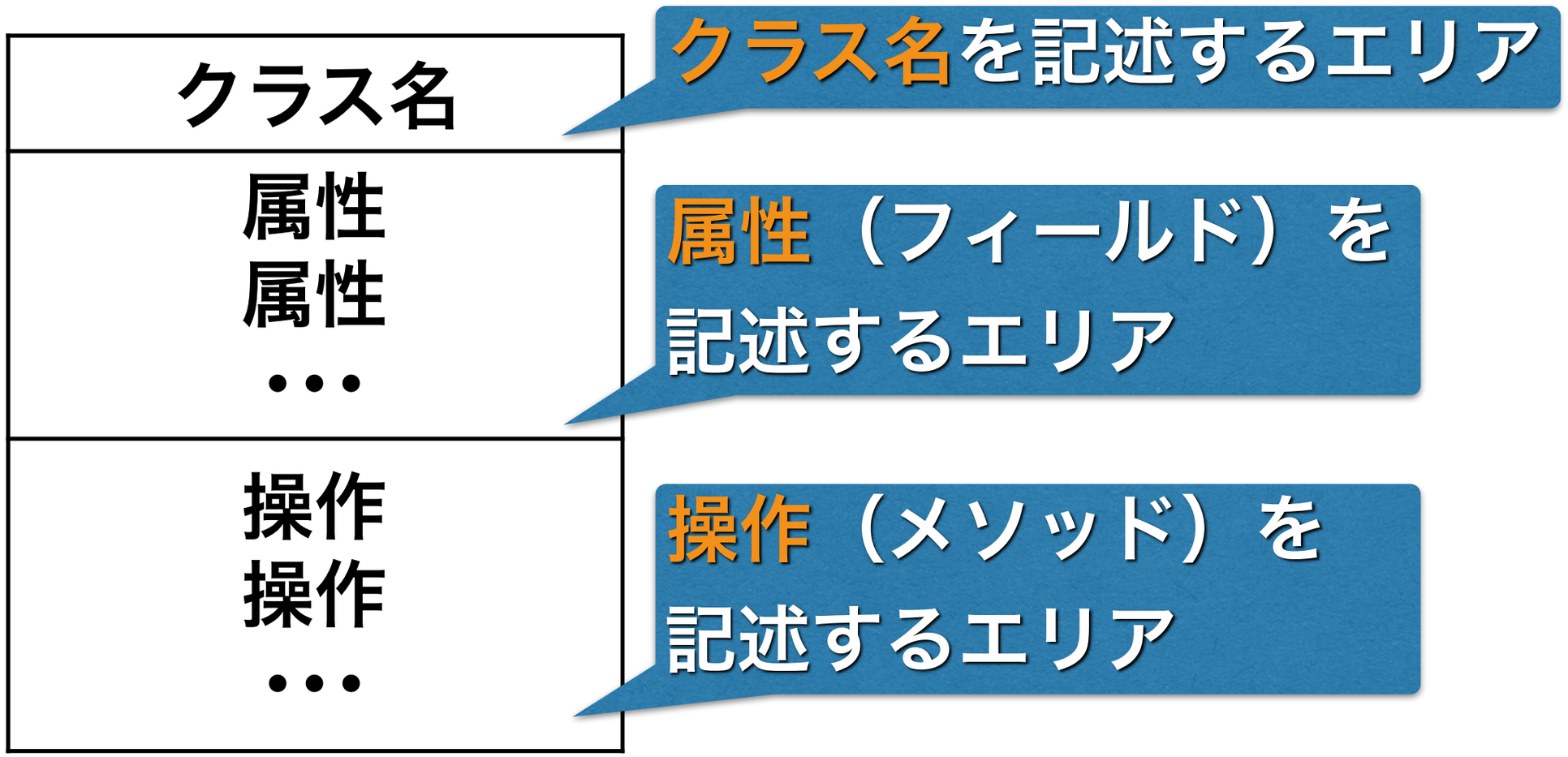
1. 端末を起動して、以下のコマンドを実行して後期第1週のフォルダを作る

```
$ cd prog4d-(ユーザ名)        (←既に移動しているなら不要)
$ mkdir week201
$ cd week201
```

※課題で作るファイル名は各自で決めて構いません。

クラスの表記

クラスは、以下の3つのエリアに分けて描きます。



属性の表記

属性はクラスが持つデータを表現します。必須なのは「名前」のみで、その他は省略可能です。

【書式】

可視性　属性名: 型　＝　初期値

- **可視性**: 他のクラスから参照可能かを表す
  「+」(public),「-」(private),「#」(protected),「~」(package)
- **属性名**: 属性を表す名前
- **型**: 属性の型
- **初期値**: 属性が最初に持つ値

操作の表記

操作はクラスが持つ振る舞いを表現します。必須なのは「名前」のみで、その他は省略可能です。

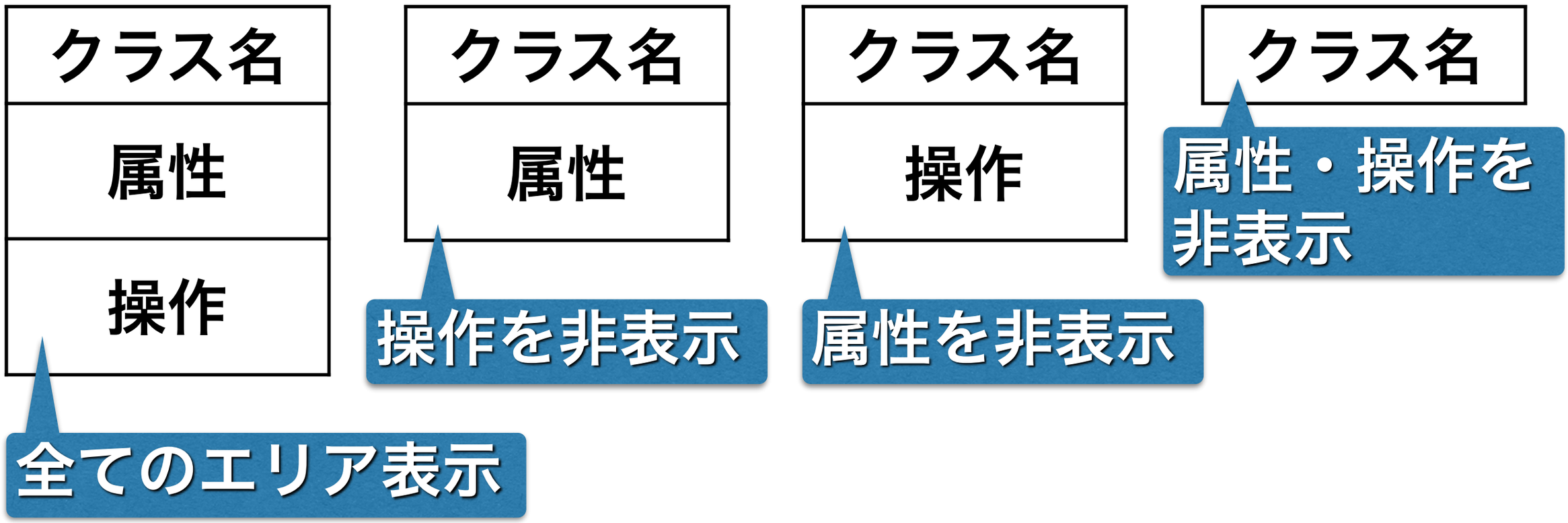
【書式】

可視性　操作名(引数名: 引数の型, ...) : 戻り値の型

- 可視性: 他のクラスから参照可能かを表す
  「+」(public),「-」(private),「#」(protected),「~」(package)
- 操作名: 操作を表す名前
- 引数名: 引数を表す名前
- 型: 引数と戻り値の型

クラス表記のバリエーション

クラスは属性および操作の表示領域のどちらか、または両方を非表示にすることができます。（つまり、非表示になっていても、属性・操作が定義されている場合があるので注意して下さい。）



astahの参考サイト

- UML初学者向けチュートリアル
  http://astah.change-vision.com/ja/tutorial/tutorial-community.html
- 基本操作ガイド
  http://astah.change-vision.com/ja/files/astah_Basic_Operation_Guide.pdf

【練習1-1】

astahを使って、次のJavaプログラムのクラス定義から、UMLクラス図を作成しましょう。

```
class MyClass {
  private int num1;
  private int num2;
  public void setNum1(int n) {}  // num1の値を設定する
  public void setNum2(int n) {}  // num2の値を設定する
  public int sum() {}            // num1+num2を返す
}
```

MyClass
- num1 : int - num2 : int
+ setNum1(n : int) : void + setNum1(n : int) : void + sum( ) : int

完成したら、属性名と操作名のみの表示に変更してみましょう。

MyClass
num1 num2
setNum1( ) setNum1( ) sum( )

【練習1-2】

練習1-1で作成したクラスMyClassに対して、先に示したようなクラス表記のバリエーションの表示に切り替えてみましょう。

【課題1-1】

別紙に示すプログラム中のクラス定義から、クラスDate, Time, Schedule, ScheduleMainを、UMLのクラス図として作成してください。

【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
$ git commit -m “課題1-1提出”
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザ名)