オブジェクト指向プログラミング

OO編　第1回

クラス/インスタンスとデータ抽象の概念

- 目的
  - クラス，インスタンスの仕組みを理解する
  - 描画ライブラリを用いたGUIの仕組みを理解する
- キーワード
  - クラス，インスタンス，オブジェクト，インスタンス化，コンストラクタ，カプセル化

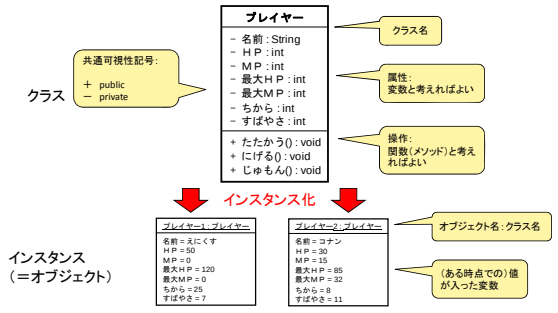
オブジェクト指向とは

- （現実）世界をオブジェクトとその関連するシステムとしてモデル化しようとする考え方
- ソフトウェアが現実世界の写像だとすれば，現実世界のモデル化によりソフトウェアが作成できる
- オブジェクト指向モデルの採用によって，現実とソフトウェアモデルをうまく融合できる可能性がある
  - 我々はよく，現実世界をオブジェクト指向的な見方で捉えている
  - ソフトウェア工学の研究ではオブジェクト指向（データ抽象）がモジュラリティーと再利用性を高めるために有効

クラスモデル

- オブジェクト指向の概念に基づく世界のモデル
  - クラス
    - 概念的には：”もの or こと”の分類をする
    - ソフトウェア的には：抽象的なデータを表現する
  - インスタンス
    - クラスから出来たオブジェクトのことを言う

クラス/インスタンス（UML表記）



オブジェクト指向の利点（カプセル化）

- クラスのないプログラム
  - データ構造が変わると，利用するアルゴリズム全てに変更必要
- 例：時刻のデータ

```
struct TimeA{
  int minute;
  int second;
}

int getMinute(TimeA* time){
  return time->minute;
}

int getSecond(TimeA* time){
  return time->second;
}
```

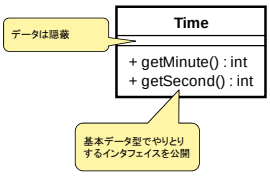
```
struct TimeB{
  int second;
}


int getMinute(TimeB* time){
  return time->second/60;
}

int getSecond(TimeB* time){
  return time->second%60;
}
```

ソフトウェアモジュールとしてのクラス

- クラス：データを抽象化し，内部データをカプセル化したもの



C++での表記

```
class Time{
   private:
      int time;
   public:
      int getMinute();
      int getSecond();
}

int Time::getMinute(){
  return time->second/60;
}

int Time::getSecond(){
  return time->second%60;
}
```

クラスの概念と書法

クラス＝オブジェクトのひな形で，データを抽象化する

名前＝「合成物に適切な名前をつける」ことが抽象化
（大文字で始めること）

```
class Point{
  int x;
  int y;

  void addX(int x){
  }
}
```

属性＝オブジェクトの持つデータ
＝インスタンス変数
（クラスは基本データを合成する）

操作＝オブジェクトの持つメソッド
（データに依存するアルゴリズムの
インターフェイスを提供する）

インスタンス化

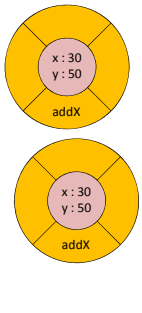
クラス

```
class Point{
  int x;
  int y;

  void addX(int x){
  }
}
```

インスタンス
＝クラスから出来たオブジェクト

インスタンス化
(new)



Mainクラスの意味

```
class Main{

  public static void main(String[] args){
          Main main = new Main();
          main.main();
  }

  void main(){
          Point p = new Point();
          p.addX(100);
  }
}
```

Mainクラスの動作

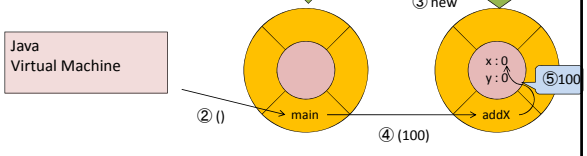
```
public static void  main
          (String[] args){
① Main main = new Main();
② main.main();
}
```

```
class Main{

  void main(){
③  Point p = new Point();
④  p.addX(100);
  }
}
```

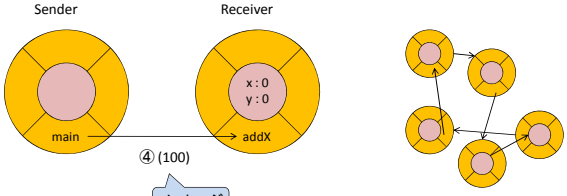
```
class Point{
  int x;
  int y;

  void addX(int x){
⑤  this.x = x;
  }
}
```



オブジェクト指向システム

オブジェクト（群）とメッセージ交換で目的を達成するシステム
＝抽象化されたデータと操作するメソッド呼び出し



メッセージ＝Javaではメソッドcallで実現

クラス設計，もう少し詳しく

```
public class Point{
  private int x;
  private int y;

  public Point(int x, int y){
    this.x = x;
    this.y = y;
  }

  public void addX(int x){
  }
}
```

アクセス修飾子＝他のオブジェクトに公開するか否か
（private 非公開，public 公開）

コンストラクタ
＝インスタンス生成メソッド
（戻り値なし，クラスと同名）

this ポインタ＝自分への参照
（引数のxと属性のxを区別する場合に使う）

デフォルトコンストラクタ

```
public class Point{
  private int x;
  private int y;

  public Point(){
  }

  public void addX(int x){
  }
}
```

コンストラクタを明示しなければ，
引数なし，何もしないコンストラクタ
が自動的に生成されている