

12回目 クラス

今日の講義で学ぶ内容

- クラスとは
- クラスの宣言と利用
- クラスの応用

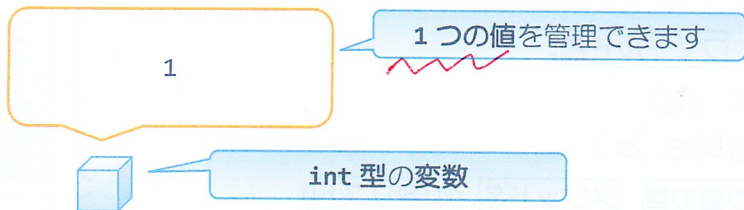
クラス

クラスとは

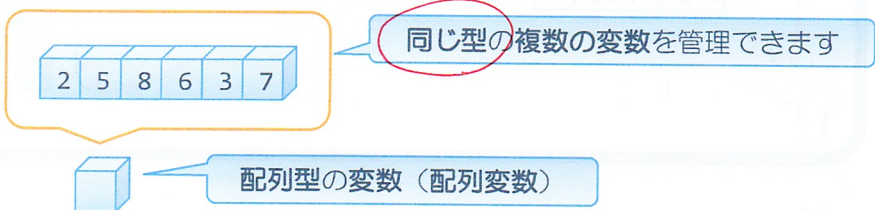
異なる複数の型の変数を内部にもつ型です

変数

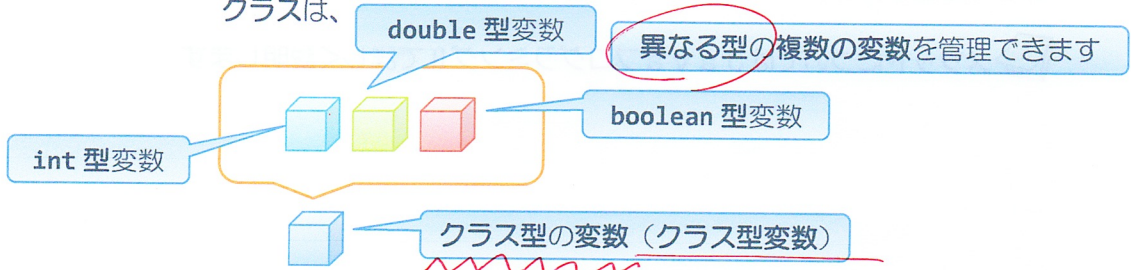
直観的に表現すると、  
int 型や double 型は



配列型は、



クラスは、



クラスは変数の他に、メソッドをもちます  
メソッドはC言語での関数に相当します

## クラスの宣言

### クラスの宣言

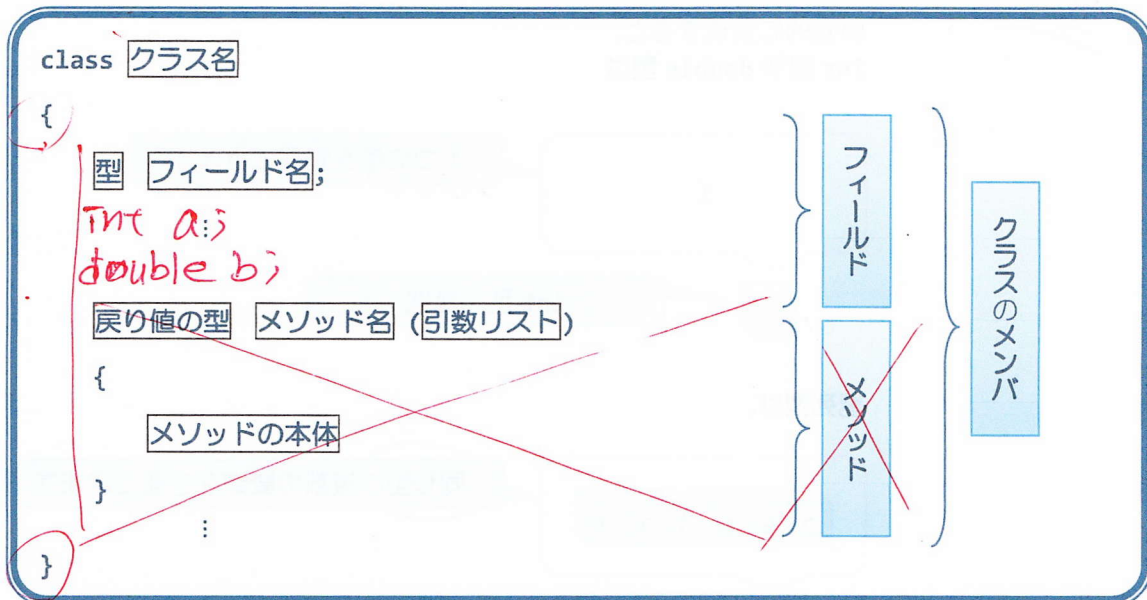
クラスのフィールド(変数のこと)とメソッド(関数のこと)を宣言して、新しい型として利用できるようにします

フィールドとメソッドはクラスのメンバといいます

クラスの宣言はキーワード `class` を指定して次のように行います

```
class クラス名 {クラスのメンバ}
```

クラスのメンバは以下のように宣言します



📄 クラスのメンバはフィールドとメソッド以外のものを持つこともできます  
ここでは基本的なフィールドとメソッドをおさえておきましょう

📄 メソッドについてはJavaプログラミングIIで詳しく説明します

たとえば、  
車はナンバーとガソリン量をもっています

したがって、  
ナンバーとガソリン量をフィールドにもつクラス Car は次のように宣言することができます

ソースコード例

ソースファイル名 : Sample12\_1.java

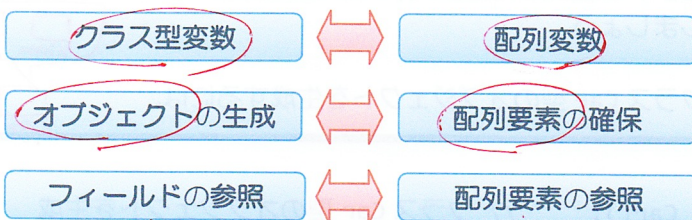
コピペ用

```
// クラス Car の宣言
class Car 型(新しい)
{
    int number; // ナンバーを格納する int 型の変数
    double gas; // ガソリン量を格納する double 型の変数
}
```

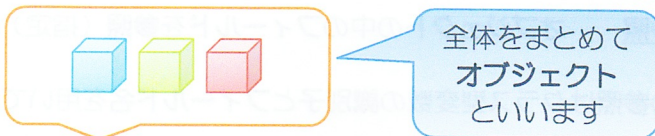
## クラスの利用

クラスの利用手順 クラス型変数の宣言 → オブジェクトの生成 → フィールドの参照

📄 クラスの利用手順を配列の利用手順と次のように  
対応させると分かりやすいでしょう



クラス型変数の宣言 クラス型変数とはオブジェクトを扱う（代入する）変数です



クラス型変数

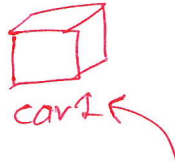
クラス型変数は通常の変数と同様に型と識別子を持ちます

型（クラス名）と識別子を指定して次のように行います

クラス名 識別子;

Car mycar;

int a;



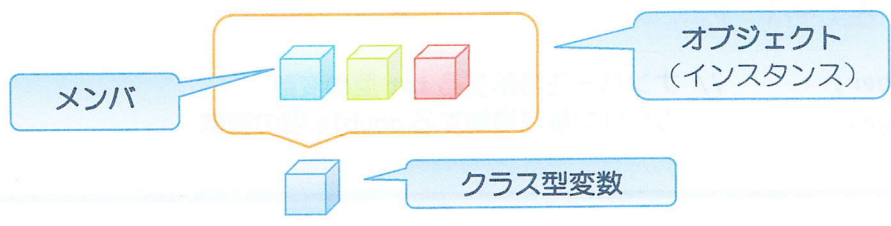
たとえば、

Sample12\_1.java で宣言したクラス Car 型の変数 (クラス型変数) を宣言するには、

```
Car car1; // クラス Car 型の変数 car1 を宣言
```

int a;

**オブジェクトの生成** オブジェクトとはクラスのメンバを格納するための領域です  
 クラスのオブジェクトはインスタンスとも呼ばれます



クラス名を指定して次のように行います

```
識別子 = new クラス名();  
car1 = new Car(); new int [2];
```

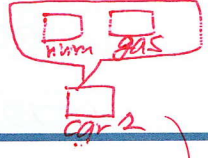
**new** 演算子は指定されたクラスのオブジェクトをコンピュータのメモリー上に作ります

オブジェクトの生成は丸括弧"()"で、配列要素の確保は角括弧"[]"です  
 違いに注意しましょう

new Inc [5];  
 配列要素aとb

たとえば、クラス Car 型のオブジェクトを生成するには、

```
Car car1;  
car1 = new Car(); // クラス Car 型のオブジェクトを生成
```



car1.num = 12345  
 (aやb)

**フィールドの参照** オブジェクトの中のフィールドを参照 (指定) して値を代入します

各フィールドの参照はクラス型変数の識別子とフィールド名を用いて次のようにします

識別子.フィールド名

識別子とフィールド名の中のピリオドは、ooの中の△△と解釈するとよいでしょう

フィールドへの値の代入は、各フィールドを参照して次のように行います

```
識別子.フィールド名 = 値;
```

たとえば、

クラス Car 型のオブジェクト car1 のフィールド number と gas に値を代入するには、

```
car1.number = 9129;  
car1.gas=30.0;
```



フィールドはオブジェクトが生成されたときに予め以下のデフォルト値が代入されます  
(型) (デフォルト値)

boolean	false
char	0 ('����')
byte、short、int、long	0
float、double、	0.0
配列変数、クラス型変数	null

※null についてはJavaプログラミングIIで説明します

### ソースコード例

ソースファイル名 : Sample12\_2.java

```
// 車クラスの宣言とその利用
```

```
// クラス Car の宣言
```

```
class Car
```

```
{
```

```
    int number;    // ナンバー
```

```
    double gas;   // ガソリン量
```

```
}
```

```
class Sample12_2
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        Car car1;           // クラス Car 型の変数
```

```
        car1 = new Car();   // クラス Car 型のオブジェクトを生成
```

```
        // 上記を同時に行うこともできる
```

```
        // Car car1 = new Car();
```

```
        // 各フィールドに値を代入
```

```
        car1.number = 9129;
```

```
        car1.gas = 30.0;
```

```
        // 各フィールドの値を出力
```

```
        System.out.println("車のナンバーは" + car1.number + "です。");
```

```
        System.out.println("ガソリン量は" + car1.gas + "です。");
```

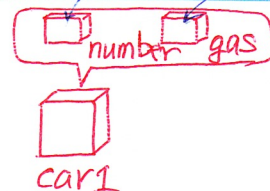
```
    }
```

```
}
```



CPad に main()メソッドを含むクラス名を知らせるため、main()メソッドを含むクラス名をファイル名と一致させましょう

9129 30.0



752  
Car

9129

30.0

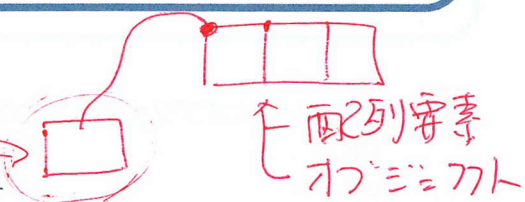
実行画面

車のナンバーは 9129 です。  
ガソリン量は 30.0 です。

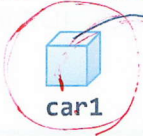
? クラス型変数は参照型変数? 基本型変数?

参照型変数には、配列変数とクラス型変数があります

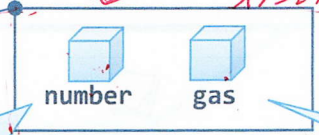
例題 Sample12\_2 のクラス型変数の振る舞いは図的に次のように理解できます



`Car car1;` // クラス Car 型の変数



`car1 = new Car();` // クラス Car 型のオブジェクトを生成



`car1.number` で参照します

`car1.gas` で参照します

`car1.num`

## ? クラスの配列を作ってみましょう

クラス型は int 型や double 型と同じで型の 1 つです

int 型の配列や double 型の配列と同様に、クラス型の配列を作成することができます

ソースコード例

ソースファイル名 : Sample12\_3.java

```
// 車クラスの配列
// クラス Car の宣言
class Car
{
    int number; // ナンバー
    double gas; // ガソリン量
}

class Sample12_3
{
    public static void main(String[] args)
    {
        Car[] cars; // クラス Car 型の配列型の変数 (配列変数)
        cars = new Car[2]; // クラス Car 型の変数 (配列要素) を 2 つ分

        cars[0] = new Car(); // クラス Car 型のオブジェクトを 1 つ生成
        cars[1] = new Car(); // 新たにクラス Car 型のオブジェクトを 1 つ生成

        // 各フィールドに値を代入
        cars[0].number = 9129;
        cars[0].gas = 30.0;

        cars[1].number = 1234;
        cars[1].gas = 15.5;

        // 各フィールドの値を出力 ← サイズ: 長さ
        for(int i=0; i<cars.length; i++)
        {
            System.out.println( i + "番目の車情報: ");
            System.out.println("車のナンバーは" + cars[i].number + "です。");
            System.out.println("ガソリン量は" + cars[i].gas + "です。");
        }
    }
}
```

クラス Car

int 型の配列

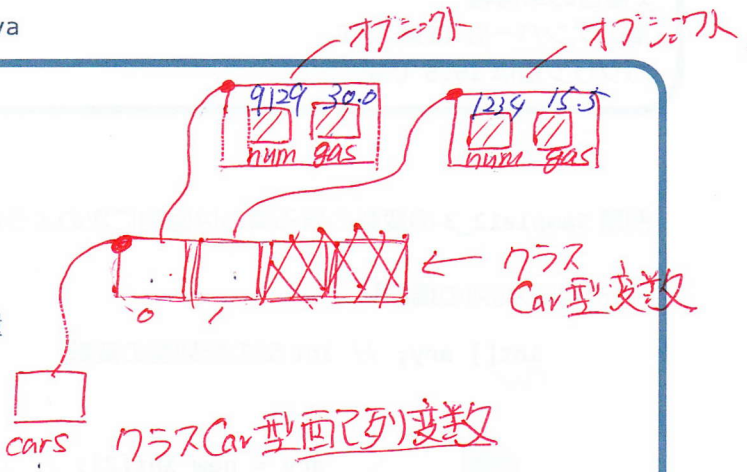
int[] arr;

arr = new int[2];

クラス Car 型の配列

Car[] arr;

arr = new Car[2];



{ [] 配列のとき  
( ) クラスのとき

1: 2 行のとき  
i  
① ②  
9129 1234  
30.0 15.5

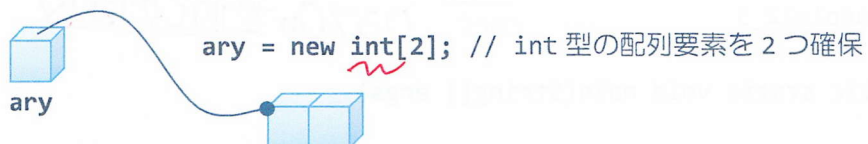
## 実行画面

0 番目の車情報：  
車のナンバーは 9129 です。  
ガソリン量は 30.0 です。  
1 番目の車情報：  
車のナンバーは 1234 です。  
ガソリン量は 15.5 です。

例題 **Sample12\_3** の変数の振る舞いは図的に次のように理解できます

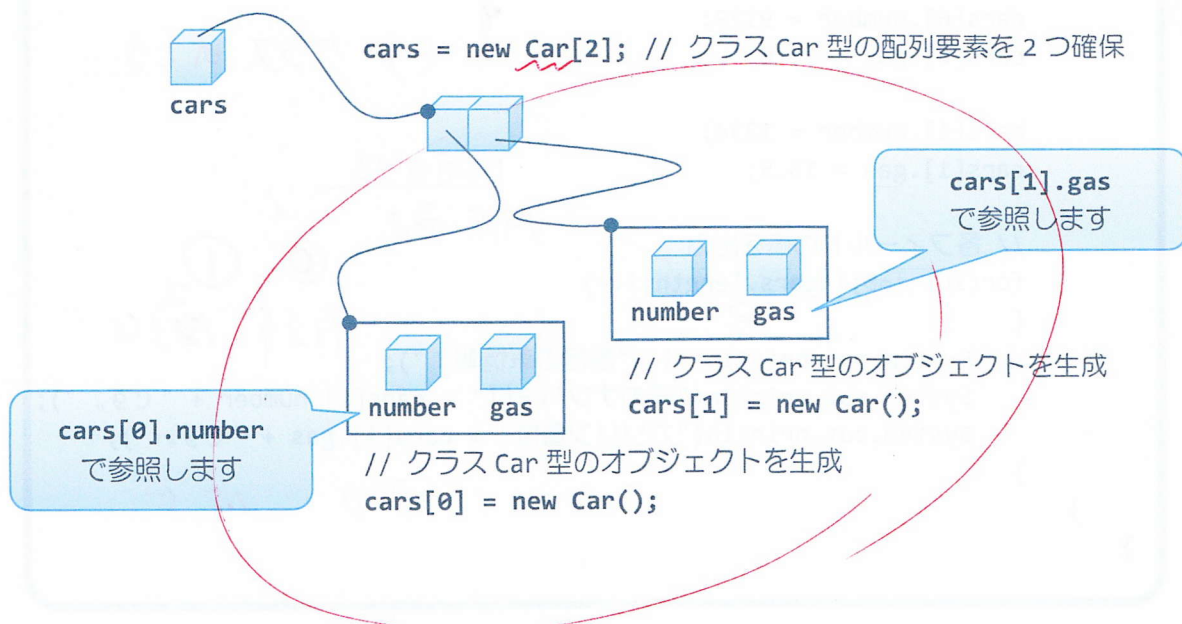
【int 型の配列の場合】

```
int[] ary; // int 型の配列型の変数
```



【クラス型の配列の場合】

```
Car[] cars; // クラス Car 型の配列型の変数
```





## ? クラスのメンバにクラス型変数を宣言してみましょう

クラス型は `int` 型や `double` 型と同じで型の 1 つです

`int` 型や `double` 型の変数をメンバにできるようにクラス型の変数をメンバにできます

ソースコード例

ソースファイル名 : `Sample12_4.java`

```
// 車クラスを別のクラスのメンバにする
// クラス Car の宣言
class Car
{
    int number;    // ナンバー
    double gas;   // ガソリン量
}

// クラス Car をメンバに持つクラス Owner の宣言
class Owner
{
    String name;
    int age;
    Car mycar; // クラス Car 型の変数をメンバにもつ
}

class Sample12_4
{
    public static void main(String[] args)
    {
        Owner owner1;           // クラス Owner 型の変数
        owner1 = new Owner();   // クラス Owner 型のオブジェクトを生成

        owner1.name = "Java";
        owner1.age = 21;
        owner1.mycar = new Car(); // クラス Car 型のオブジェクトを生成
        owner1.mycar.number = 9129;
        owner1.mycar.gas = 30.0;

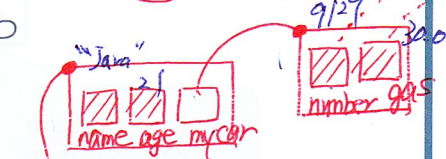
        // 各フィールドの値を出力
        System.out.println("所有者");
        System.out.println("名 前:"+owner1.name);
        System.out.println("年 齢:"+owner1.age);
        System.out.println("車");
        System.out.println("車ナンバー:"+owner1.mycar.number);
        System.out.println("ガソリン量:"+owner1.mycar.gas);
    }
}
```

クラス  
Car

クラス  
オーナー

メンバに他のクラス型をもつ

基本型



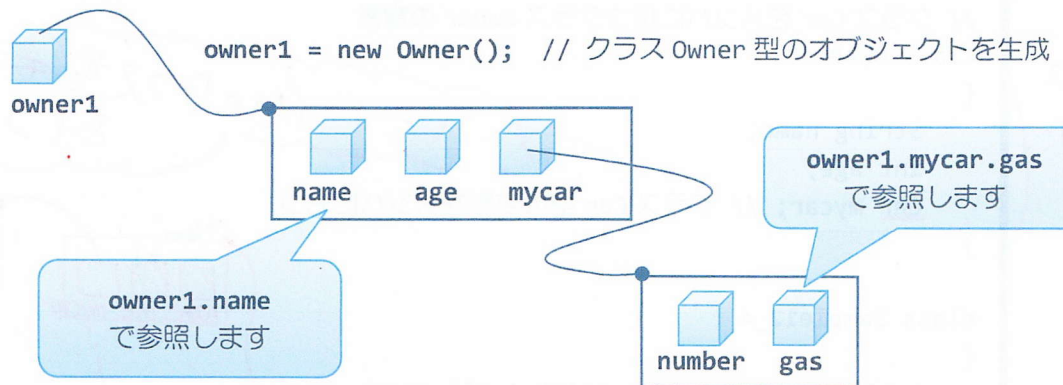
参照型

## 実行画面

```
所有者  
名前:Java  
年齢:21  
車  
車ナンバー:9129  
ガソリン量:30.0
```

例題 `Sample12_4` の変数の振る舞いは図的に次のように理解できます

```
Owner owner1; // クラス Owner 型の変数
```



```
// クラス Car 型のオブジェクトを生成  
owner1.mycar = new Car();
```

### ■ 今日の講義のまとめ ■

- クラスを用いることにより、異なる複数の型の変数を一括して管理できます。
- クラスが持つフィールド（変数）やメソッドをクラスのメンバといいます。
- 指定されたメンバをもつ新しいクラス（クラス型）の準備はクラスの宣言により行います。
- クラスを利用するときは、クラス型の変数を宣言し、オブジェクトを確保します。オブジェクトはクラスのメンバを格納するためのメモリ領域です。メンバへのアクセスは、ピリオドを用いて行います。
- クラス型の変数は参照型の変数です。
- クラス型の変数は `int` 型や `double` 型と同じように配列を構成できます。

