

質問1 voidとはどんな型なのか？

回答

void “戻り値無し”を表わすプレースホルダであり、データ型ではない

*void is a placeholder indicating no return type.*¹⁾

*¹⁾著書「The Java Programming Language 4th edition」サンマイクロシステムズ社より

質問2 なぜコンストラクタは戻り値がないのに void が指定できないのか？

回答

メソッドとコンストラクタはその宣言の形やオーバーロードが可能であることから大変に類似している。しかし、次に示す定義によりこれらは明確に区別される。

メソッドの定義

修飾子 戻り値の型 メソッド名 (引数リスト)

```
{  
    本 体  
}
```

修飾子 public, private など

戻り値 メソッドが返す戻り値の型
値を返さない場合は void を指定

メソッド名 メソッドの名前

引数リスト 引数とその型の組からなるリスト
引数がない場合は空にする (void も指定しない)

本 体 処理コード

コンストラクタの定義

修飾子 クラス名 (引数リスト)

```
{  
    本 体  
}
```

修飾子 public, private など

引数リスト 引数とその型の組からなるリスト
引数がない場合は空にする (void も指定しない)

本 体 処理コード

コンストラクタの定義より、その宣言には戻り値の宣言を含めることができないことが分かる。この理由であるが、文献1で調べたところ関連する記述は見当たらなかった。次のサンプルを見てほしい。コンストラクタの宣言に戻り値の宣言を含めるとメソッドとして認識される。戻り値の型の宣言の有る／無しがこれらを明確に区別するための働きをしている。

```

class Test{
    public Test(){
        System.out.println("This is Constructor!!");
    }
    public void Test(){
        System.out.println("This is not Constructor!!");
    }
}

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

```

```

>java Main
This is Constructor!!
This is not Constructor!!
-- Press any key to exit (Input "c" to continue) --

```

質問3 クラスのメンバとして宣言される変数とメソッドのブロック内で宣言される変数ってどう違うの？

回答

変数は、次の3種類に分けられる。

1. フィールド
2. ローカル変数
3. パラメータ

各変数の特徴は、次の表1に示す通りである。

表1 変数の種類と特徴

変数の種類	宣言の位置	変数にアクセスできる領域(スコープ)
フィールド	クラスのメンバ インタフェース ^{※1} のメンバ内	アクセス制御やクラス変数・クラスメソッド、継承 ^{※1} など様々な要素に依存
ローカル変数	各ブロック内 ・メソッドブロック ・コンストラクタブロック ・if文ブロック ・while文ブロック など 特別な場合：for文の初期化の式 ^{※2}	宣言された位置から そのブロックの最後まで

パラメータ	メソッドの仮引数リスト コンストラクタの仮引数リスト try-catch 文の仮引数リスト※ ¹	対応するブロックの内側
-------	---	-------------

(※1) 以後の講義で解説
(※2) 前期の for 文の回を参照

フィールドは、メンバとして宣言される変数である。public や private 修飾子（アクセス制限）や static 修飾子（クラス変数）などとともに宣言される。ローカル変数は、ブロック内で宣言される変数である。ローカル変数へのアクセスは宣言されたブロック内で可能であり、ブロック外からはアクセスできない。パラメータは、仮引数リストで宣言される変数である。ローカル変数と同様の性質をもち、該当するブロック内でのみアクセスが可能である。

上に示したように、変数はその宣言される位置で区別される。これらの位置のおおまかな把握のために、下のサンプルを参考にして欲しい。

```
class A{
    // フィールド
    private int i;

    public void Func(int n){ // パラメータ
        // ローカル変数
        int j;
    }
}
```