**第12章　java修飾子**

**修飾子の種類やその挙動について説明します。**

**java修飾子**

**1.メンバ変数定義時に付与できる修飾子**

※ 修飾子は上から定義する順に並んでいます。

|  |  |
| --- | --- |
| **修飾子** | **説明** |
| アクセス修飾子 | アクセス修飾子を指定することにより、メンバ変数を他のクラスから参照できないなどを指定できるようになります。 |
| static | メンバ変数がインスタンス変数ではなく、**クラス変数**であることを表します。 |
| final | finalをつけると変数に一度値が生成されるとその値を変更できず定数として扱われます。 |
| transient | シリアライズすべきでないメンバ変数を表します。 |
| volatile | メンバ変数に対して、コンパイラが何らかの最適化を施すことを防止します。 |

**2.メソッド定義時に付与できる修飾子**

※ 修飾子は上から定義する順に並んでいます。

|  |  |
| --- | --- |
| **修飾子** | **説明** |
| アクセス修飾子 | アクセス修飾子を指定することにより、メソッドを他のクラスから参照できないなどを指定できるようになります。 |
| static | メソッドがインスタンスメソッドではなく、**クラスメソッド**であることを表します。 |
| abstract | 処理を実装しないメソッド（抽象メソッド）を生成したい時に使用します。 |
| final | finalを使用した場合、そのメソッドはサブクラスでオーバーライドすることができません。 |
| native | C言語など他の言語で記述されたプログラムをJavaプログラム内で使用したい時に使います。使用したいプログラムにnativeを付け定義します。 |
| synchronized | スレッドを複数同時に走らせる際に同じデータに対し複数のメソッドが呼び出される場合があります。そのような場合、安全にメソッドが実行されるためにsynchronizedを使用します。 |

**3.アクセス修飾子**

**3-1.アクセス修飾子の種類**

指定できるアクセス修飾子を以下に記載します。

クラス定義時に指定できるアクセス修飾子は「public」と「指定無し（デフォルト）」のみです。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **アクセス修飾子** | **クラス** | **メンバ変数** | **メソッド** | **コンストラクタ** |
| public | ○ | ○ | ○ | ○ |
| protected | × | ○ | ○ | ○ |
| 指定無し（デフォルト） | ○ | ○ | ○ | ○ |
| private | × | ○ | ○ | ○ |

**3-2.アクセス修飾子の使用方法**

• **public**と指定すると、どこからでも参照できます。  
• **protected**と指定すると、そのクラスを継承したサブクラスからしか参照できません。  
• **修飾子を省略する**と、同じパッケージのクラスからしか参照できません。  
• **private**と指定すると、そのクラスの中からしか参照できません。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **アクセス修飾子** | **自パッケージ** | | | **他パッケージ** | |
| **自クラス** | **サブクラス** | **他クラス** | **サブクラス** | **他クラス** |
| public | ○ | ○ | ○ | ○ | ○ |
| protected | ○ | ○ | ○ | ○ | × |
| 指定無し（デフォルト） | ○ | ○ | ○ | × | × |
| private | ○ | × | × | × | × |

**例**

[Television.java]

* package JSample;
* public class Television{
* private int channelNo;
* private void setChannel(int newChannelNo){
* channelNo = newChannelNo;
* System.out.println("新しいChannelNo=" + channelNo);
* }
* }

[JSample12\_1.java]

* package JSample;
* public class JSample12\_1{
* public static void main(String args[]){
* Television tv1 = new Television();
* tv1.setChannel(1);
* }
* }

**実行結果**(コンパイルの段階でエラーとなります。)

Exception in thread "main" java.lang.Error: Unresolved compilation problem:  
メソッド setChannel(int) は型 Television で不可視です  
  
at JSample.JSample11*6.main(JSample11*6.java:7)

この例では、TelevisionクラスのprivateメソッドのためJSampleクラスからはアクセスできません。

そのためコンパイルエラーとなります。

**4.static**

メンバ変数、メソッドを定義する際、static修飾子が付与されたものを**クラス変数**、**クラスメソッド**、static修飾子が付与されていないものを**インスタンス変数**、**インスタンスメソッド**といいます。

**mainメソッド**はインスタンスを生成する前に実行する必要があるため、static修飾子を付けて定義しています。

**static変数**とはクラスに1つだけ存在する変数です。

オブジェクトごとに存在するインスタンス変数とは異なり、オブジェクトを生成しなくてもアクセスできます。  
static変数はクラス単位で管理したいときに便利な変数です。

**例**

クラス変数の参照方法を例示します。  
[ExClass.java]

* package JSample;
* class ExClass{
* static int standardSize;
* void setSize(int num) {
* standardSize = num;
* }
* }

[JSample12\_2.java]

* class JSample12\_2 {
* public static void main(String[] args) {
* ExClass ins1 = new ExClass();
* ins1.setSize(100);
* ExClass ins2 = new ExClass();
* ins2.setSize(300);
* System.out.println(ins1.standardSize);
* System.out.println(ins2.standardSize);
* System.out.println(ExClass.standardSize);
* }
* }

**実行結果**

300

300

300

クラス変数は、インスタンス間で値を共有するため、一番最後に設定された300が表示されます。

staticメンバーを利用する場合の注意点を、以下にまとめておきます。  
（1）staticメソッドからthisにはアクセスできない。  
staticメソッドでは、いわゆるインスタンスはありませんので、当然、インスタンスを示すthisにもアクセスできません。  
（2）staticフィールドの利用は限定的。  
staticフィールドは、インスタンスフィールドとは違い、内容を変更した場合にもすべてのインスタンスに影響します。

**例**

クラス変数、クラスメソッドの参照方法を例示します。  
[JSample12\_3.java]

* package JSample;
* class JSample12\_3 {
* static String classV = "クラス変数classV";
* static void classM() {
* System.out.println("クラスメソッドclassM");
* }
* public static void main(String[] args) {
* System.out.println(JSample12\_3.classV);
* JSample12\_3.classM();
* }
* }

**実行結果**

クラス変数classV

クラスメソッドclassM

**練習**

**問題** [AccessTest.java]

* package test;
* public class AccessTest{
* public void a(){
* System.out.println("public");
* }
* protected void b(){
* System.out.println("protected");
* }
* void c(){
* System.out.println("修飾子なし");
* }
* private void d(){
* System.out.println("private");
* }
* }

[JEx12\_1.java]をコンパイルしてみましょう。

* package JSample;
* import test.AccessTest;
* public class JEx12\_1{
* public static void main(String[] args){
* AccessTest at = new AccessTest();
* at.a();
* at.b();
* at.c();
* at.d();
* }
* }

**問題** [JEx12\_2.java]

* public class JEx12\_2 {
* public static void main(String[] args) {
* Dog dog1 = new Dog(3);
* Dog dog2 = new Dog(5);
* System.out.println(dog1.age);
* }
* }
* class Dog {
* static int age;
* public Dog(int age) {
* this.age = age;
* }
* }

**実行結果を選択してください**

1. 「3」と表示
2. 「5」と表示
3. 何も表示されない
4. コンパイルエラー
5. 実行時エラー