抽象クラス

不完全なクラス

あるオブジェクトの共通する基底クラスを生成しようとした場合
基底クラスは必要だが、インスタンス化するほど完成していない場合もあります

このような場合は、不確定要素を宣言のみにし、定義は派生クラスで行う方法があります
これを行うには、クラスに **abstract** 修飾子を指定します
これが指定されたクラスは**抽象クラス**となります

抽象クラスはインスタンス化できない不完全なクラスで
その実体は派生クラスで初めて活かされることになります

```
abstract class Kitty {
 public void Write() {
  System.Console.WriteLine("Kitty on your lap");
 }
}

class Taruto : Kitty {
 new public void Write() {
  System.Console.WriteLine("Magical nyan nyan TARUTO");
 }
}

class Test {
 static void Main() {
  Kitty obj = new Taruto();
  obj.Write();
 }
}
```

Kitty クラスは抽象クラスなのでインスタンスを生成することはできません
しかし、Kitty 型の変数は生成できるため派生クラスをキャストすることができます
上のプログラムのように隠蔽されたメンバにアクセスすることもできます
このプログラムを実行すると "Kitty on your lap" という文字列が表示されるでしょう

必要に応じて、抽象クラスは**抽象メソッド**を宣言することができます
抽象メソッドはメソッドの宣言のみを行い、その実体は派生クラスに委ねるものです
同時に、抽象メソッドは仮想メソッドの性質も持っています（そのため virtual は指定できない）

抽象メソッドを持つ基底クラスを継承した派生クラスは、抽象メソッドを再定義する必要があります
そのクラスが抽象メソッドを再定義できない場合は、そのクラスもまた抽象クラスにします

抽象メソッドの宣言も abstract 修飾子を指定します
この場合、メソッドは**本体 { } を定義できません**

```
abstract class Kitty {
 public abstract void Write();
}

class Taruto : Kitty {
 public override void Write() {
  System.Console.WriteLine("Magical nyan nyan TARUTO");
 }
}

class Test {
 static void Main() {
  Taruto obj = new Taruto();
  obj.Write();
 }
}
```

Kitty クラスの抽象メソッド Write() は実装を持ちません
これは仮想メソッドを発展させたもので、その実体は常に派生クラスにあります
抽象メソッドは抽象クラスにしか宣言することができません

抽象クラスの抽象メソッドを宣言することで、この派生クラスは必ず抽象メソッドをオーバーライドし
独自の再定義をしなければならないという約束が発生します
抽象メソッドを持つ派生クラスは、そのメソッドの実体を保証します

---

abstract

クラス、メソッド、プロパティに指定できる修飾子です
抽象クラス、抽象メソッド、抽象プロパティを宣言します

前のページへ　戻る　次のページへ