this キーワード

暗黙的なインスタンス

インスタンスメソッドやコンストラクタは、実行時にインスタンスが必要です
このインスタンスにメソッドやコンストラクタ内でアクセスしたい場合、どうするのでしょうか？

インスタンスのメンバにアクセスする場合
インスタンスメソッド内ではインスタンスを省略してメンバ名だけでアクセスできましたが
実は、これは暗黙的にインスタンスを参照しているのです

インスタンスメソッドは、暗黙的にインスタンスの参照 **this** ポインタを持ちます
インスタンスメソッドが実行時に持っているインスタンスを **カレントインスタンス** と呼び
this ポインタはこのカレントインスタンスを表します
そのため、静的メソッドは this ポインタを持っていません

```
class Kitty {
 public string str;
 public void Write() {
  str = "Kitty on your lap";
  System.Console.WriteLine(this.str);
 }
}

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

Kitty クラスの Write() メソッドはカレントインスタンスへの参照 this を持ちます
str メンバ変数にアクセスする時、単純に str としてもアクセスすることができますが
これは this.str の省略であるということがこのプログラムを実行するとわかるでしょう

this ポインタを使用することで、メソッドの変数とメンバ変数の問題を解決できます
メンバ変数とメソッド内の変数名が衝突 することがあると思いますが
この場合は、コンパイルエラーになることはありません

```
class A {
 public int x;
 public void method(int x) {
  x = 10;
 }
}
```

メソッドの中でメンバ変数と名前が衝突している変数にアクセスした場合
メソッドの中で定義された変数（ローカル変数）が優先となります
上の例の場合では、method() メソッドは 仮パラメータ x にアクセスしています
そこで、メンバ変数には this を使って明示的にアクセスすることができるようになります

```
class Kitty {
 public string name;
 public Kitty(string name) {
  this.name = name;
 }
}

class Test {
 static void Main() {
  Kitty obj = new Kitty("RENA");
  System.Console.WriteLine(obj.name);
 }
}
```

Kitty クラスのコンストラクタ Kitty() は、仮引数で name を受け取ります
name 変数は Kitty.name メンバ変数と名前が衝突しています
そのため、メンバ変数にアクセスする時は this ポインタを使って明示的にアクセスします

this

カレントインスタンスへの参照です
インスタンスメソッド、コンストラクタ、インスタンスアクセッサで使えます
ただし、静的メンバは this ポインタを保有しません

前のページへ　戻る　次のページへ