



4_C#カプセル化

アジェンダ (agenda : 目次)

• 今回のアジェンダ

- staticメンバとstaticクラス
- 演算子オーバーロード
- カプセル化
- 課題

• 今回の範囲

- かんたんC# : 1~334p
- Code : Qs2_0 ~ Qs2_3
- 課題 : Qs2_pro

Staticメンバ

インスタンスを生成せずに使えるメンバです。

```
ClassName.Method();
```

のような形で使用します。

メモリ上では実行時でなくコンパイル時に展開されます。
実行速度は速いですが、メモリの容量を取るのです。
頻繁に共通して使う機能などに使用します。

Staticクラス

C言語の構造体に近い振る舞いをします。

インスタンスを作らずに使う
便利な機能をまとめたりするのに使います。

Staticクラスでは
staticメンバ以外を定義できない

演算子オーバーロード

C++の物とほぼ同じです。
演算子を関数的に使えます

ゲーム開発では実数計算じゃない計算で多用しますので、
必ずマスターしてください。
(※ベクトルや論理値など)

**※ゲームではない一般的なソフトウェア開発だと
禁止される場合が多いです。**

カプセル化

カプセル化とは、『**隠ぺい**』という風に訳されます。

アクセス修飾子(private とか protected とか)
を用いて値のアクセス範囲を制限したり、
直接弄れなくすることをカプセル化と言います。

『getterとかsetterとか作るのがカプセル化』という認識で大丈夫です。

※コード上ではあまり実感できないでしょうが、
セキュリティ的にかなり重要です。

(※Debugでバグ探しをする際には役立ちます)

課題：Qs2_proを完成させよ。

実行結果が以下のようなクラスを作成せよ。

Main内は一切変更してはならない。

Microsoft Visual Studio デバッグ コンソール

```
【問題1(等速運動と等加速度運動)：】  
・ 初速speed(1.2f,1.4f)で3秒間等速運動：  
x:3.6000001, y:4.2, magnitude:5.5317264  
・ 初速speed(1.2f,1.4f), 加速度accel(0.1f,0.2f)で3秒間等加速度運動：  
x:4.05, y:5.1, magnitude:6.512488  
【問題2(放物線運動)：】  
物体が地面に付くのに2.45秒以上掛かりました。
```

まとめ

- ・ 頻繁に共通して使うものはStatic
- ・ Staticクラスはstaticメンバ以外は定義出来ない。
- ・ オーバーロードは必ず出来るように。
- ・ カプセル化はアクセス制限で行う。

▶ 今回の範囲

- ・ かんたんC# : 1~334p
- ・ Code : Qs2_0 ~ Qs2_3
- ・ 課題 : Qs2_pro

• アジェンダ

- ・ staticメンバとstaticクラス
- ・ 演算子オーバーロード
- ・ カプセル化
- ・ 課題