



3_C#基礎

アジェンダ (agenda : 目次)

• 本日のアジェンダ

- 基本的な進め方
- コメントで学習
- 基本文法
- 関数
- クラスの基本
- 構造体

• 今回の範囲

- かんたんC# : 1~334p
- Code : Qs1_0 ~Qs_4

基本的な進め方

授業

コードを書きながら解説していきます。
授業時間は限られていますので、
細かいところは省く場合があります。
(※その際は解説コード参照)

また、

- ・コード上だと理解が難しい場合
- ・要点をまとめたい場合

スライドで解説します。

復習と予習と課題

解説コメントが沢山書かれている
『**解説コード**』を配ります。

基本的には
それで予習と復習を行ってください。

動画は休んだ時以外は
あまりおススメしません。
(※タイパが悪い)

課題がある場合は
課題も行ってください。
(※**提出を求めるかもしれません**)

コメントで学習

クリエイターとなることを選んだ皆さんは、一生勉強し続ける必要があります。
(※統計出てますが、他業種の7倍以上勉強することになります)

特にプログラマになる人は、
教科書じゃなくて

他人の ソースコードとコメント で学習する事が殆どです。
(※特に第二言語はその比率が非常に高くなります)

皆さんは第二言語としてC#を学ぶので、
ソースコードに直接書かれたコメントで学習する事に慣れましょう。

エントリーポイント

```
static void Main(){}
```



コイツがエントリーポイント

プログラムの開始位置が、**エントリーポイント**です！

C++やC#に限らず、
プログラミングと呼ばれる物には全てエントリーポイントがあります。

他人のソースコードを読むときは、
まずはエントリーポイントを探しましょう。

※エントリーポイントから読み出せば理解できるような
コードorコメントを書くのが大切です。

コーディング規約

ソースコードを書く際の書き方のルールの事です。

会社・組織・プロジェクトで変わりますが、
最近は言語毎のルールに合わせるのが主流です。

C#はマイクロソフトが規約を作ってます。

<https://learn.microsoft.com/ja-jp/dotnet/csharp/fundamentals/coding-style/coding-conventions>

この規約に沿って書きますので、
とりあえず、

- ・ キャメルケース (例: temp string なら tempString)
- ・ パスカルケース (例: show string なら ShowString)

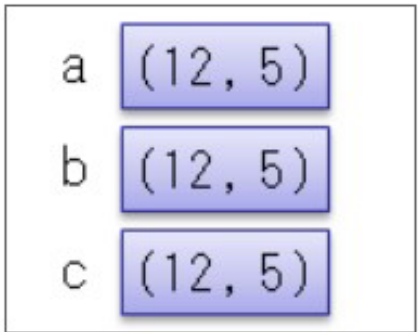
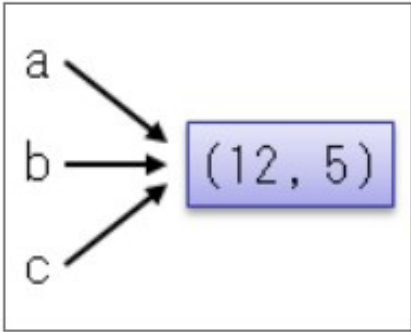
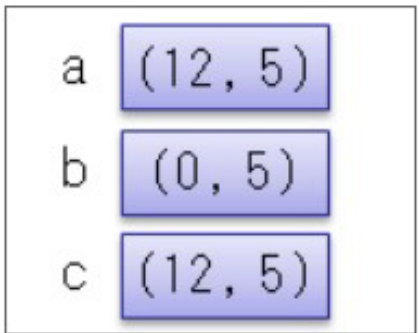
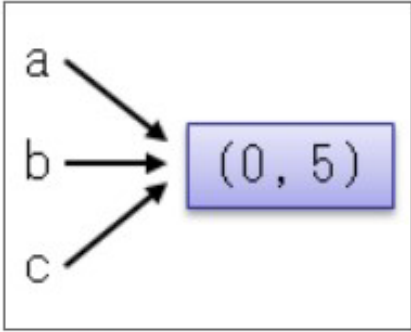
は覚えましょう。

ポインタじゃなくて参照型 1



C# の型の分類

ポインタじゃなくて参照型 2

	値型	参照型
代入時	 <p>それぞれの変数は値のコピーを保持。</p>	 <p>値の実体は別のところ※にあり、それぞれの変数は実体への参照のみを持つ。</p>
b の値変更時	 <p>b の値のみ変更される。</p>	 <p>b が参照している実体の値が変更される。同じ実体を参照している a と c も変更されたかのように見える。</p>

※ 「ヒープ」と呼ばれる領域に実体を確保します。

ポインタじゃなくて参照型 3

値型・参照型の特徴

	値型	参照型
代入時	値の複製が生じる	値は複製しない
利点	間接参照が生じないので、メンバーアクセスが高速	複製が生じないので、変数への代入・引数渡しが高速
	<u>スタック</u> を使うのでメモリ確保が早い	
欠点	型のサイズが大きいとき、複製のコストが大きい	間接参照が生じて、メンバーアクセス時に少しコストがかかる
	継承・多態的ふるまいができない	<u>ヒープ</u> を使うのでメモリ確保が少し遅い

このような特徴があるため、通常は データのサイズが小さく、継承の必要のないものは構造体として定義し、それ以外のものはクラスとして定義します。

ポインタじゃなくて参照型4

C#の型の分類

C#には組込み型、クラス、構造体など、さまざまな型がありますが、これらは以下のように分類されます。

値型	構造体型	ユーザー定義構造体(<code>struct</code>)		
		数値型	整数型	<code>byte, sbyte, char, short, ushort, int, uint, long, ulong</code>
			浮動小数点型	<code>float, double</code>
			<code>decimal</code>	
		<code>bool</code>		
	列挙型(<code>enum</code>)			
参照型	クラス(<code>class</code>)			
	インターフェース(<code>interface</code>)			
	デリゲート(<code>delegate</code>)			
	<code>object</code>			
	<code>string</code>			
	配列			

注: 色付き文字で書かれているものは組込み型

デストラクタは殆ど無意味

C#では、メモリ管理は**GC(ガベージコレクション)**が自動で行います。

その影響を最も受けるのが、デストラクタです。

メモリ解放がいつ行われるのか分からないので、
デストラクタに書かれた処理が何時行われるのか分かりません。

なので、**基本的にデストラクタは使いません。**

機能はありますが、使わないでください。

値型は殆ど構造体

C#において、`int`や`float`などの値型の型は
`enum`(列挙型)を除き全て構造体で作られている。
だからフィールド(メンバ変数)とかメソッド(メンバ関数)とか持つ。

例：`int.Parse()`，`配列名.Length` など。

C++のような純粋な値型はありません。

全部メンバとフィールドを持つので、すごく便利です。

まとめ

- ・コメントで学習
- ・エントリーポイント
- ・コーディング規約
- ・ポインタじゃなくて参照型
- ・デストラクタは殆ど無意味
- ・値型は殆ど構造体

▶ 今回の範囲

- ・かんたんC# : 1~334p
- ・Code : Qs1_0 ~ Qs_4

• アジェンダ

- ・基本文法
- ・関数
- ・クラスの基本
- ・構造体