



Язык программирования C# 6.0 модуль 2

Основы синтаксиса C#

- Язык регистрозависимый.
- Рекомендуемая кодировка для исходных файлов UTF-8
- Признаком завершения команды является - ;
- Для начала и окончания блока кода используется пара фигурных скобок – { }

Языковые лексемы

- Пробельные символы.
- Комментарии

// - однострочный комментарий

/* - многострочный комментарий

*/

/// документирующий комментарий

Список рекомендованных тегов

<c>

<code>

<example>

<exception>*

<include>*

<list>

<para>

<param>*

<paramref>

<permission>*

<remarks>

<returns>

<see>*

<seealso>*

<summary>

<typeparam>*

<typeparamref>

<value>

* - компилятор проверяет
синтаксис

Вопрос. Как гарантировать единый стиль в оформлении
исходников?

- Идентификаторы

буквы, _ , цифры

Пример: abc, _abc, if, да

- Ключевые слова

abstract as basebool break byte case catch char checked class
const continue decimal default delegate do double else enum
event explicit extern falsefinally fixed float for foreach
goto if implicit in int interface internal is lock long namespace
new null object operator out override params private
protected public readonly refreturn sbyte sealed short
sizeof stackalloc static string struct switch this throw true try
typeof uint ulong unchecked unsafe ushort using virtual void
volatile while

- Литералы

true, false, null, 123, 2.5m

's' , '\n', "test string\n",

@"test string \n" (точная или буквальная строка)

- Операции и пунктуация

{ } [] () . , : ; + - * / % & | ^ ! ~ =
< > ? ?? :: ++ -- && ||
-> == != <= >= += -= *= /= %= &= |= ^= <<
<<= ==> >> >>=

- Команды препроцессора

#define, #undef

#if, #elif, #else, и #endif

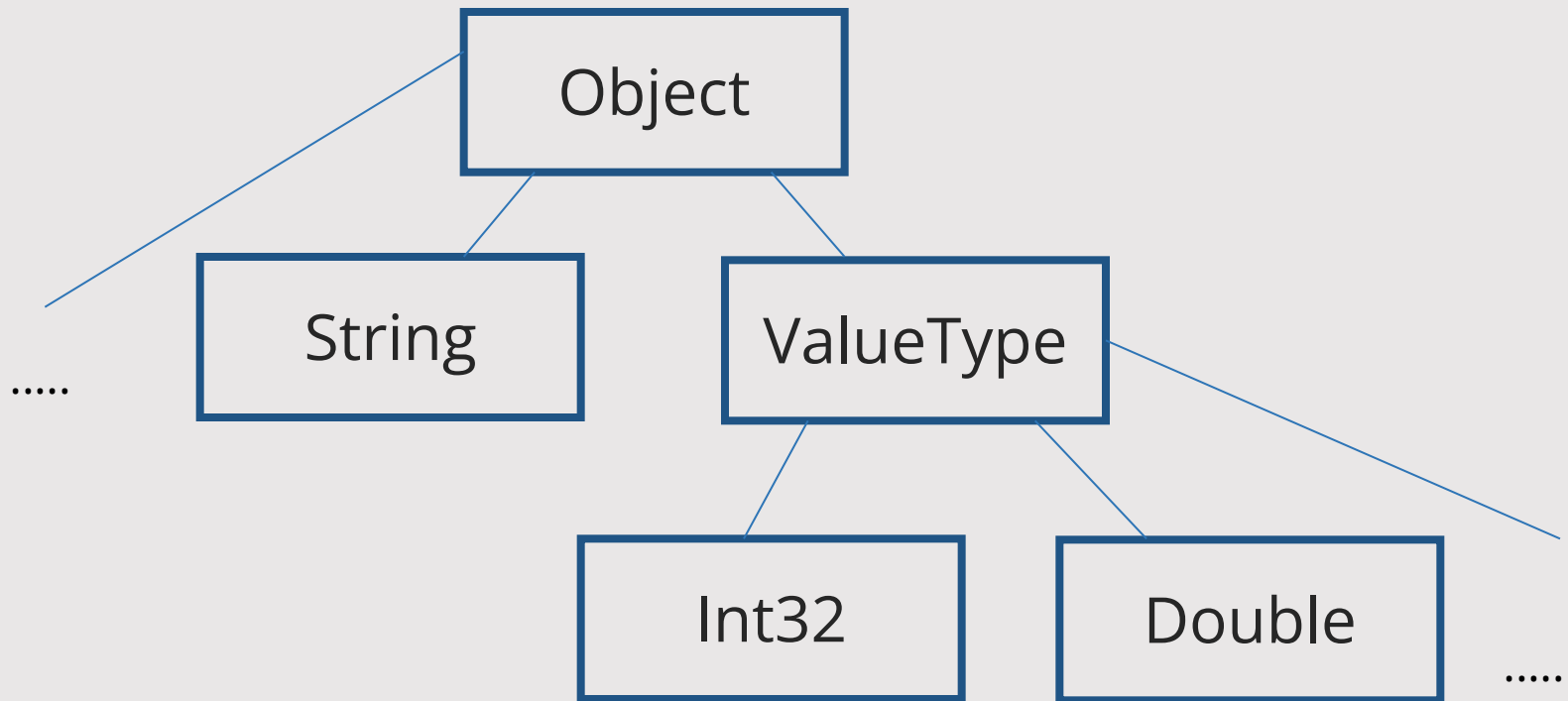
#line

#error и #warning

#region и #endregion

#pragma

Система типов C#



Вопрос. А есть в C# указатели?

Простые типы

Category	Bits	Type	Range/Precision
Signed integral	8	sbyte	-128...127
	16	short	-32,768...32,767
	32	int	-2,147,483,648...2,147,483,647
	64	long	-9,223,372,036,854,775,808...9,223,372,036,854,775,807
Unsigned integral	8	byte	0...255
	16	ushort	0...65,535
	32	uint	0...4,294,967,295
	64	ulong	0...18,446,744,073,709,551,615
Floating point	32	float	1.5×10^{-45} to 3.4×10^{38} , 7-digit precision
	64	double	5.0×10^{-324} to 1.7×10^{308} , 15-digit precision
Decimal	128	decimal	1.0×10^{-28} to 7.9×10^{28} , 28-digit precision

Пространства имён

```
namespace System
```

```
{
```

```
    class String
```

```
    {
```

```
        ...
```

```
    }
```

```
}
```

```
namespace K.A.I
```

```
{
```

```
    class String
```

```
    {
```

```
        ...
```

```
    }
```

```
}
```

Ошибки

?

Создание переменных

```
class Test
{
    static void Main() {
        short a=1, b;
        var c=0f;
        a=b/c;
        Console.WriteLine(a);
    }
}
```

Вопрос. Сколько ошибок в приведённом выше коде?

Неявные преобразования для простых типов

- sbyte -> short, int, long, float, double, or decimal.
- byte -> short, ushort, int, uint, long, ulong, float, double, or decimal.
- short -> int, long, float, double, or decimal.
- ushort -> int, uint, long, ulong, float, double, or decimal.
- int -> long, float, double, or decimal.
- uint -> long, ulong, float, double, or decimal.
- long -> float, double, or decimal.
- ulong -> float, double, or decimal.
- char -> ushort, int, uint, long, ulong, float, double, or decimal.
- float -> double.

Операции языка C#

Category	Operators
Primary	x.y f(x) a[x] x++ x-- new typeof default checked unchecked delegate
Unary	+ - ! ~ ++x --x (T)x
Multiplicative	* / %
Additive	+ -
Shift	<< >>
Relational and type testing	< > <= >= is as
Equality	== !=
Logical AND	&
Logical XOR	^
Logical OR	
Conditional AND	&&
Conditional OR	
Null coalescing	??
Conditional	?:
Assignment and lambda expression	= *= /= %= += -= <<= >>= &= ^= = =>

Nullable

- Только для ValueType

- Сравните:

```
int number; number = null;
```

```
int? number; number = null;
```

- Как получить значение?

```
number.HasValue true number.Value
```

или

```
int n = number ?? 0;
```

Enum – список именованных констант

```
enum Color: long  // по умолчанию int
{
    Red,
    Green=5, // по умолчанию 1
    Blue
}
```

```
int num=Color.Red;  // ok?
Color color=5;  // ok?
```

Struct – объединение данных под одним именем

```
struct Point
{
    public int x, y;
    public int z=1;    //error!
}

Point p;                //чему равны x, y?
Point []points=new Point[10];
```

Важно !!

- нет наследования.
- а какой порядок следования элементов структуры?

Практика

- Создайте новое консольное приложение.
- Создайте перечисление с именами(типами) 3-5 плоских геометрических фигур.
- Определитесь с набором характеристик, общих для всех плоских геометрических фигур.
- Создайте структуру с этими данными. Добавьте в структуру тип геометрической фигуры.
- Создайте, инициализируйте и распечатайте одну из фигур в коде.