Промышленное программирование

Лекция 1

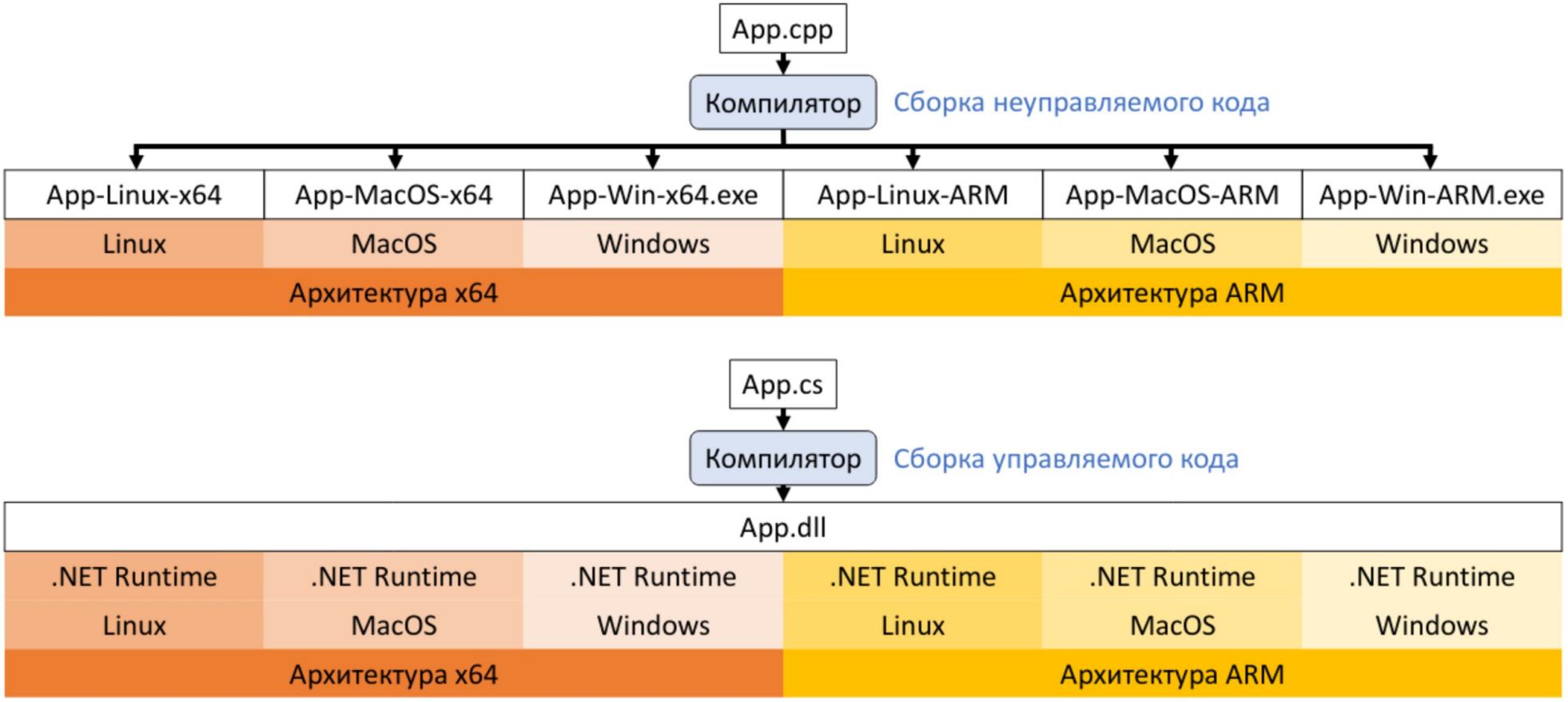
❶ Обзор .NET
❷ Типы значений и ссылочные типы
❸ Примитивные типы и операторы
❹ Ввод-вывод

1. **Обзор .NET**
2. Типы значений и ссылочные типы
3. Примитивные типы и операторы
4. Ввод-вывод

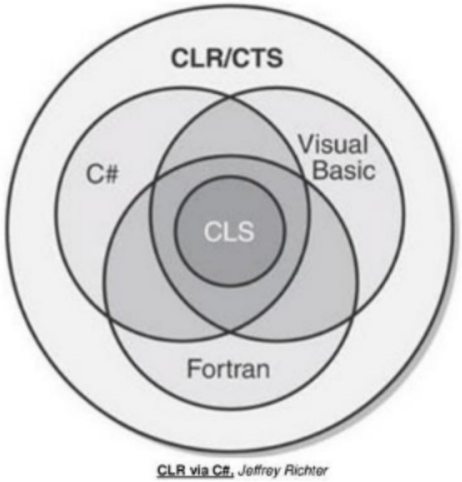
Зачем ещё один си-подобный язык?

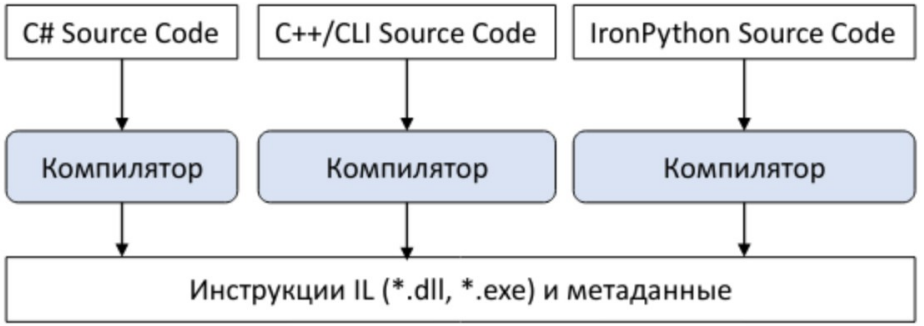
Язык	C	C++	C#
Парадигма	процедурная	ООП	ООП
Компиляция в ...	нативный код	нативный код	промежут. код
Доступ к hardware	полный	полный	ограниченный (?)
Управление памятью	ручное	автоматическое (?)	автоматическое
Ориентация на ...	производительность	производительность, надежность (?)	надежность, скорость разработки
Требования к квалификации	средние	высокие	средние

Неуправляемый и управляемый код



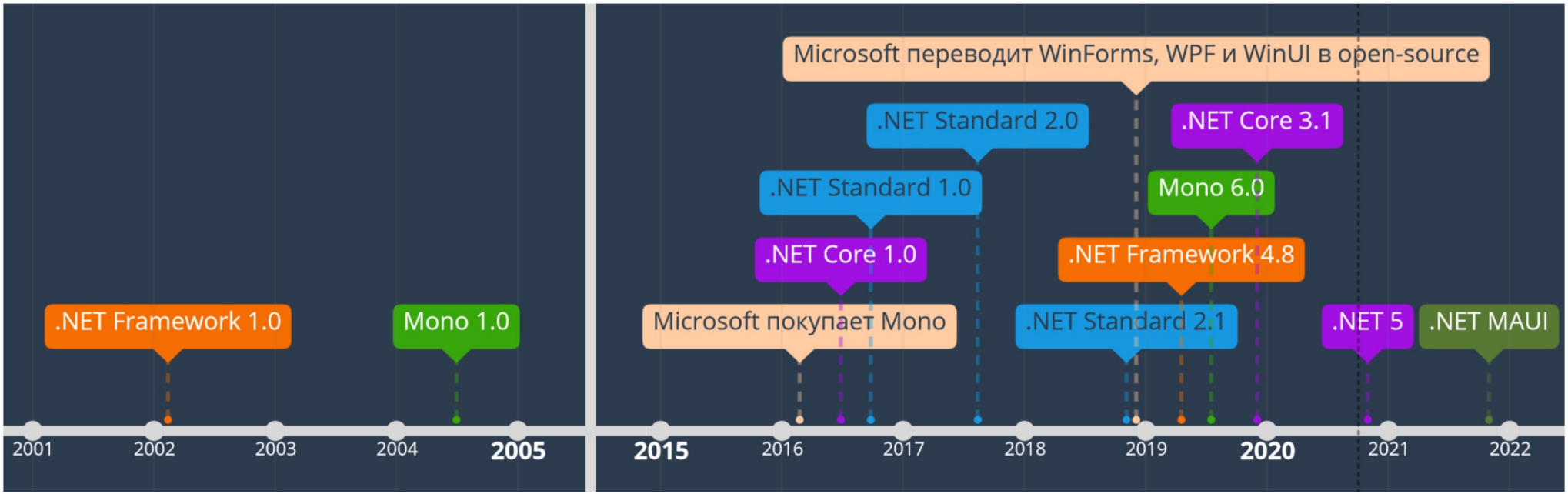
Основные понятия .NET





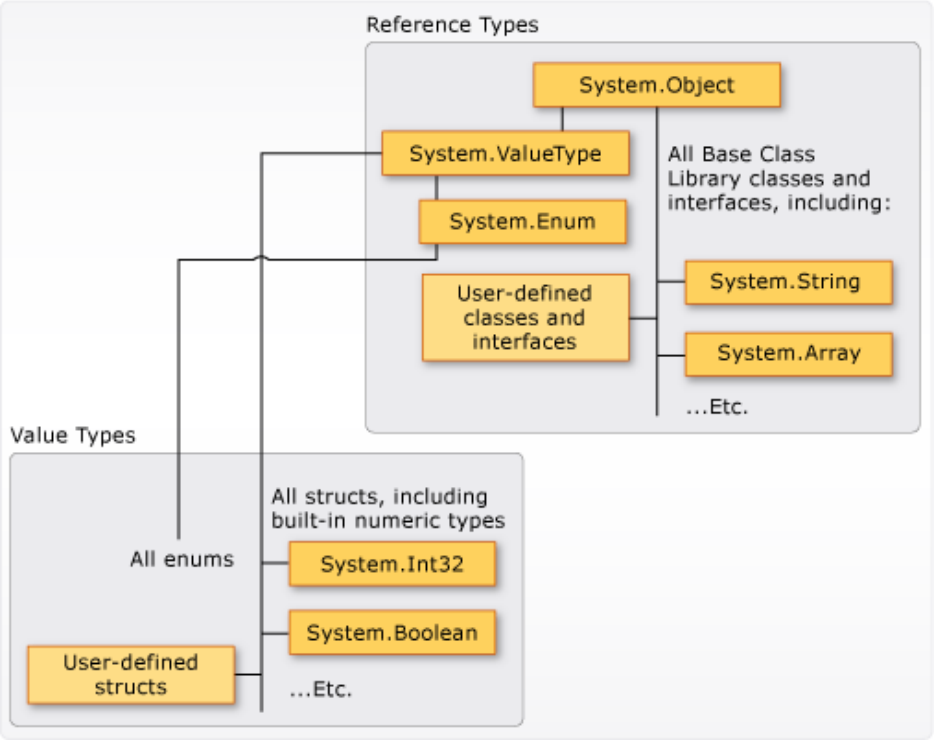
CLR	**Common Language Runtime (Общеязыковая исполняющая среда)** Управление объектами .NET (управление памятью, координирование потоков и т. п.).
CTS	**Common Type System (Общая система типов)** Описывает поддерживаемые CLR возможные типы данных и программные конструкции.
CLS	**Common Language Specification (Общеязыковая спецификация)** Подмножество общих типов и программных конструкций, которые должны поддерживать все языки программирования для .NET.
CIL **(IL, MSIL)**	**Common Intermediate Language (Общий промежуточный язык)** «Высокоуровневый ассемблер» виртуальной машины .NET.

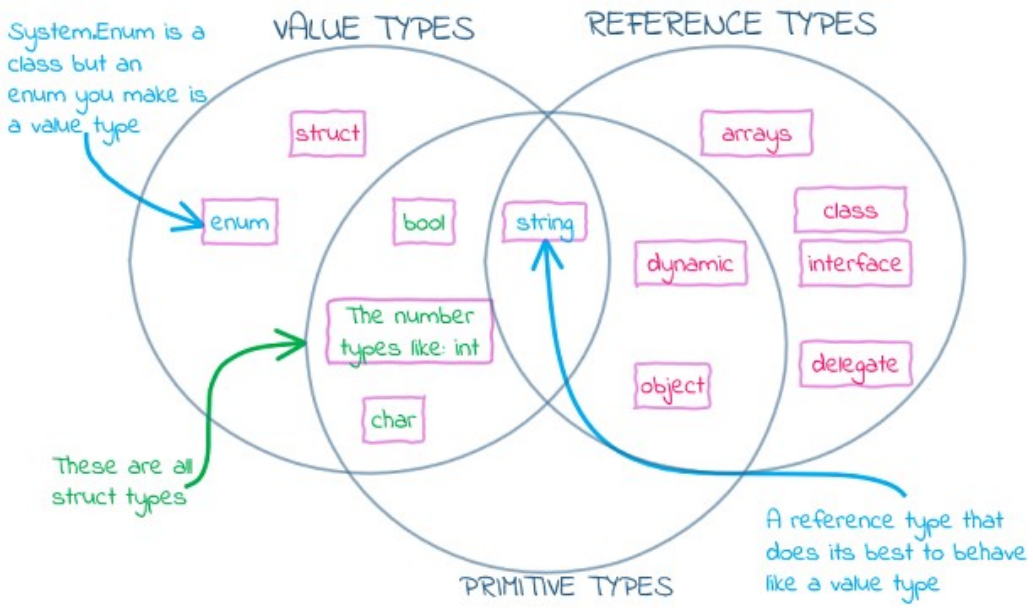
История .NET


.NET Framework 1.0
Mono 1.0
Microsoft переводит WinForms, WPF и WinUI в open-source
.NET Standard 2.0
.NET Core 3.1
.NET Standard 1.0
Mono 6.0
.NET Core 1.0
.NET Framework 4.8
Microsoft покупает Mono
.NET Standard 2.1
.NET 5
.NET MAUI
2001
2002
2003
2004
2005
2015
2016
2017
2018
2019
2020
2021
2022

1. Обзор .NET
2. **Типы значений и ссылочные типы**
3. Примитивные типы и операторы
4. Ввод-вывод

Иерархия типов



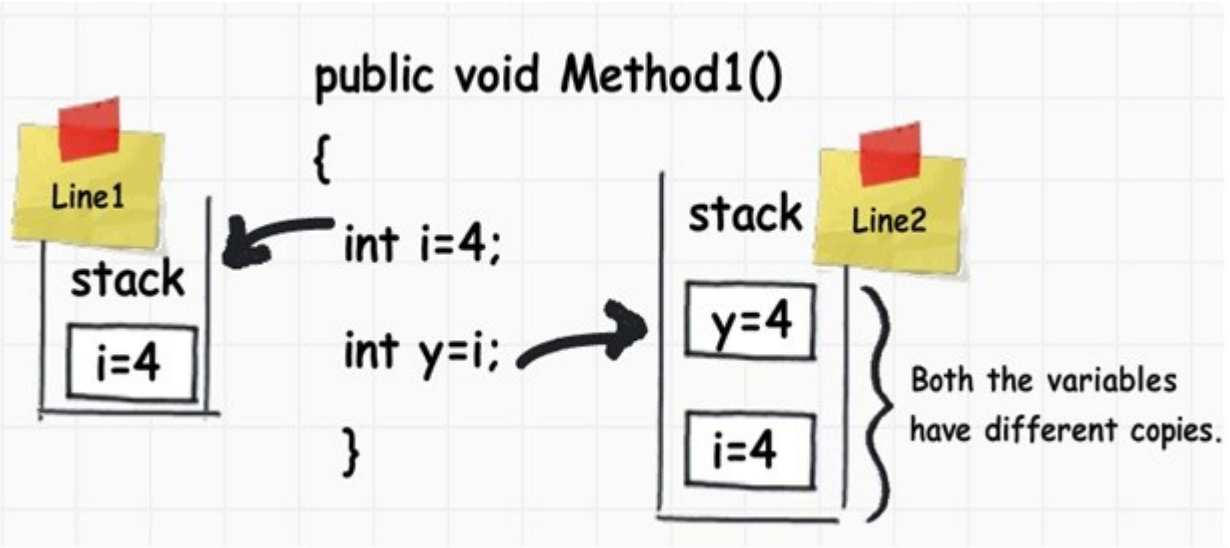


https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/types/
https://www.codeproject.com/Articles/1263638/A-Re-Introduction-to-Csharp-References-Post-Csharp

The mother of all objects

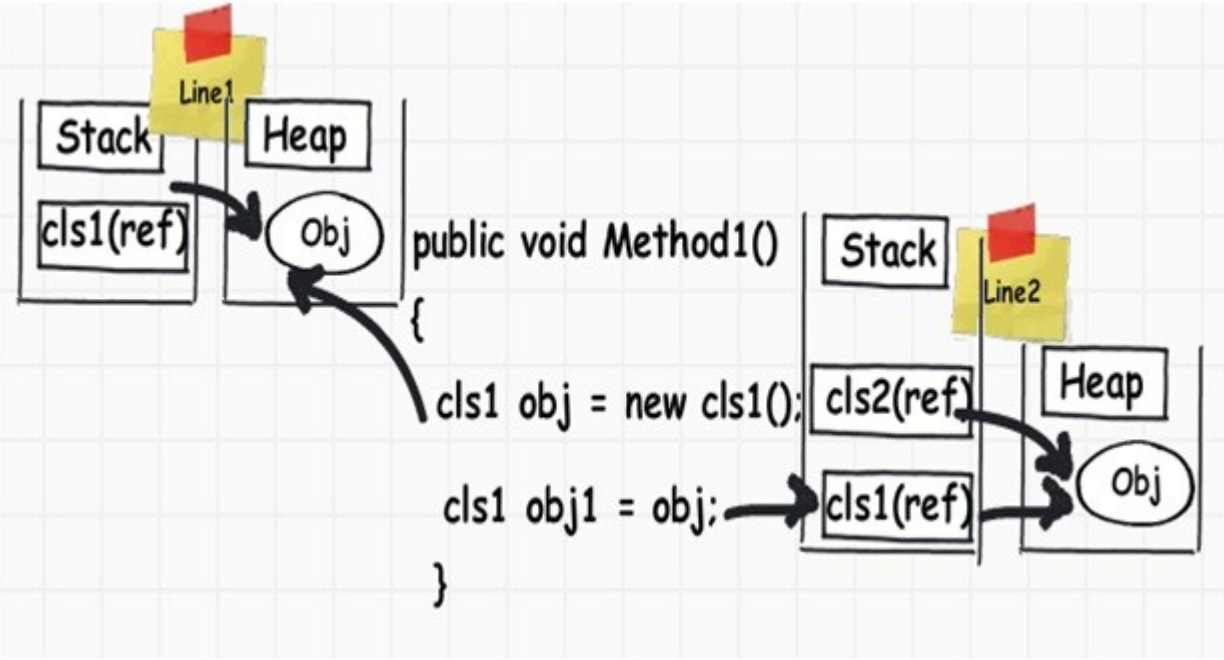
```
namespace System
{
    public class Object
    {
        public static bool Equals(Object? objA, Object? objB);
        public static bool ReferenceEquals(Object? objA, Object? ObjB);

        public virtual bool Equals(Object? obj);
        public virtual int GetHashCode();
        public Type GetType();
        public virtual string? ToString();

        protected Object MemberwiseClone();
    }
}
```

Типы значений (struct)



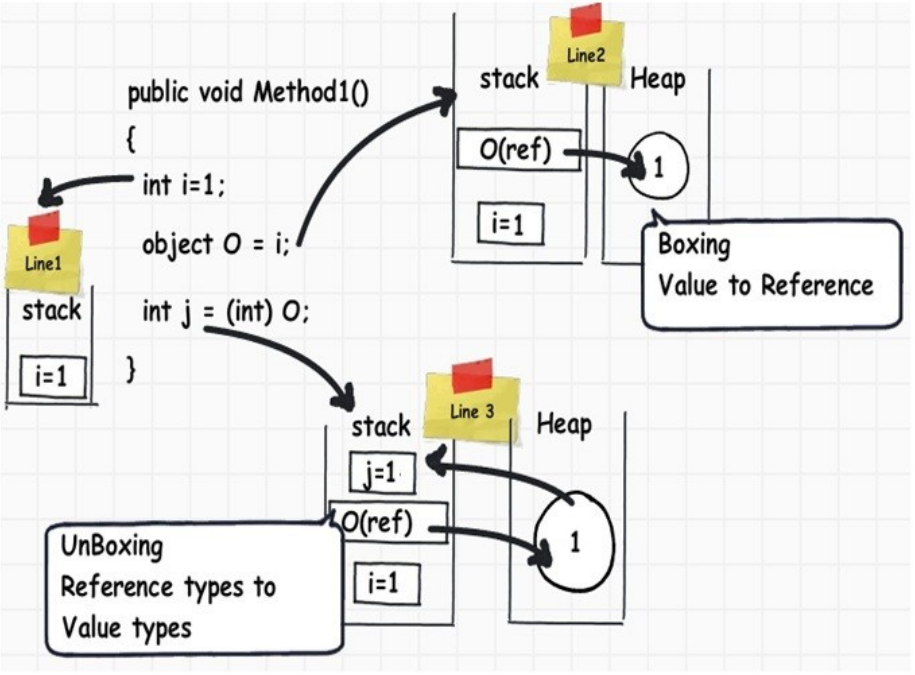
https://gurunguns.wordpress.com/2012/10/14/stack-heap-value-types-reference-types-boxing-and-unboxing/

Ссылочные типы (class, interface)



https://gurunguns.wordpress.com/2012/10/14/stack-heap-value-types-reference-types-boxing-and-unboxing/

Упаковка/распаковка (boxing/unboxing)



https://gurunguns.wordpress.com/2012/10/14/stack-heap-value-types-reference-types-boxing-and-unboxing/

1. Обзор .NET
2. Типы значений и ссылочные типы
3. **Примитивные типы и операторы**
4. Ввод-вывод

Примитивные типы данных

CLS	C# keyword
System.Object	object
System.Boolean	bool
System.Byte	byte
System.Int16	short
System.Int32	int
System.Int64	long
System.Single	float
System.Double	double
System.Decimal	decimal
System.Char	char
System.String	string

```
// Логические литералы
var t = true;
var f = false;

// Целочисленные литералы
var dec = 1_048_576;
var bin = 0b_0010_0001;
var hex = 0xDEAD;

// Вещественные литералы
var f32 = 3.14f;
var f64 = 3.14;
var f128 = 3.14m;
```

Операторы

Category	Operators
arithmetic	-, +, *, /, %, ++, --
logical	&&, ||, !, ^
binary	&, |, ^, ~, <<, >>
comparison	==,!=, >, <, >=, <=
assignment	=, +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=
string concatenation	+
type conversion	(type), as, is, typeof, sizeof
other	., new, (), [], ?:, ??

https://www.csharp-console-examples.com/general/operators-in-c/

1. Обзор .NET
2. Типы значений и ссылочные типы
3. Примитивные типы и операторы
4. **Ввод-вывод**

Вывод данных

```
var n = 50.123;
Console.WriteLine(n);                                          // 50.123
Console.WriteLine(n.ToString(CultureInfo.InvariantCulture));   // 50.123
Console.WriteLine(n.ToString("F2", CultureInfo.InvariantCulture)); // 50.12
Console.WriteLine("{0,0:F1} {1,0:F2}", n, n + 1);              // 50.1 50.12
Console.WriteLine("{0,5:F1} {0,5:F1}", n);                     //  50.1  50.1

var o = new object();
Console.WriteLine(o);                                          // System.Object
```

https://docs.microsoft.com/en-us/dotnet/standard/base-types/formatting-types
https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-numeric-format-strings
https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-numeric-format-strings

Ввод данных

```
int ReadInt(string prompt, int min = int.MinValue, int max = int.MaxValue)
{
    while (true)
    {
        Console.Write(prompt);
        var line = Console.ReadLine();
        if (line == null)
            throw new EndOfStreamException();
        if (!int.TryParse(line, NumberStyles.Integer, CultureInfo.InvariantCulture,
                          out var result))
        {
            Console.WriteLine("Failed to parse int. Try again.");
            continue;
        }
        if (min <= result && result <= max)
            return result;
        Console.WriteLine($"Integer must be in range [{min}, {max}]. Try again.");
    }
}
```

Заключение

1. object – базовый ссылочный тип для любой сущности
2. Типы значений:
    1. Передаются сами объекты целиком по значению
    2. Equals сравнивает поля по значению, == не определён
3. Ссылочные типы:
    1. Копируются ссылки, но не сами объекты
    2. По умолчанию Equals сравнивает ссылки (адреса объектов)
    3. При необходимости рекомендуется перегружать Equals
    4. По умолчанию == сравнивает ссылки (исключение: string)
    5. Перегружать == не рекомендуется
4. При приведении типа значения к object и обратно используется упаковка/распаковка (boxing/unboxing)