



PADI 2012/13

Aula 1

Introdução à Plataforma .NET

Sumário

1. Framework .NET

- Arquitetura

2. Linguagem C# 2.0

- Sintaxe
- C# vs. Java vs. C++

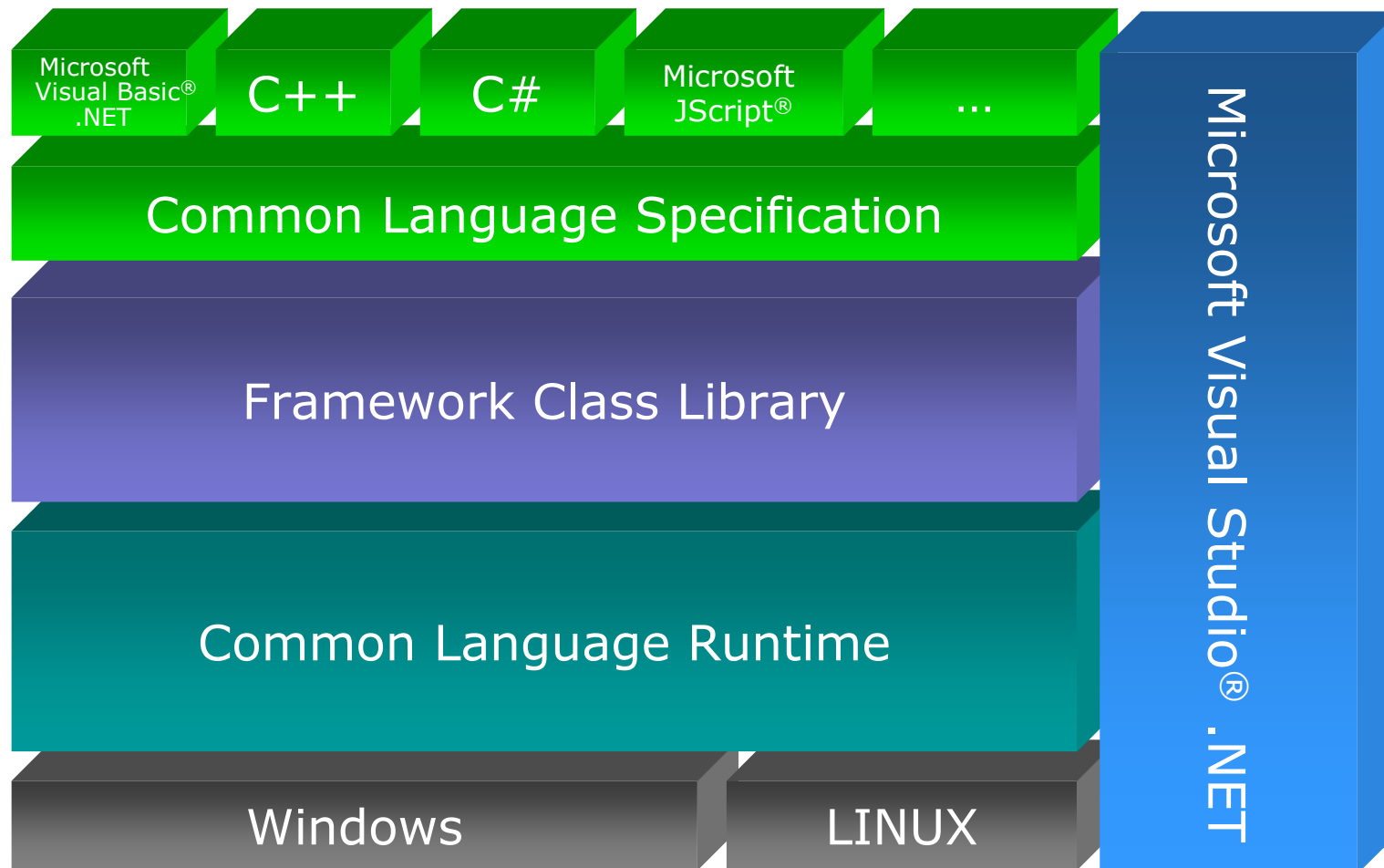
3. IDE: MS Visual Studio 2005 ou superior

- Ferramentas
- Console/Win Form Applications

1. Framework .NET

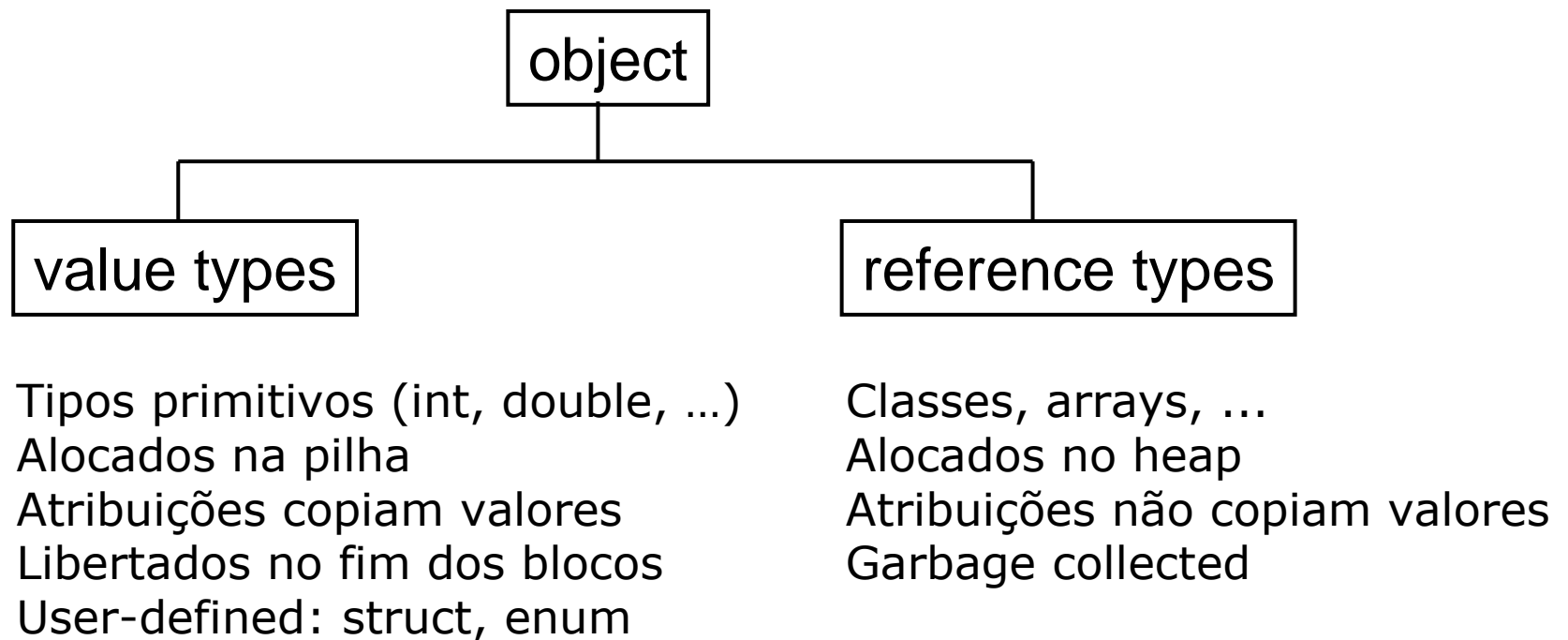
Introdução
Arquitectura

Arquitectura da Framework .NET



Common Language Runtime

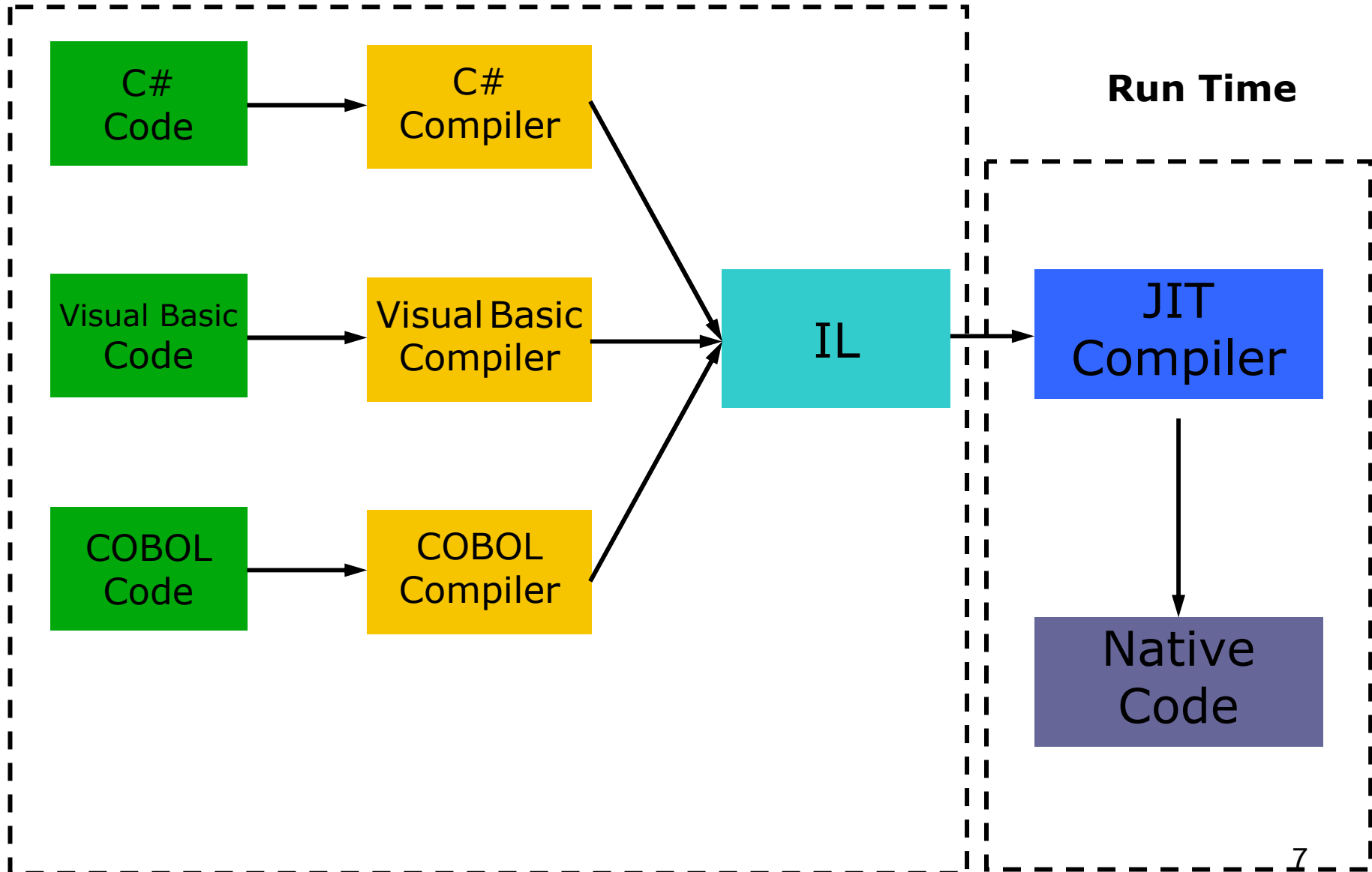
- Ambiente de execução
- Gestão de memória
- Garbage collection
- Comon type system



Framework Class Library

- System
- System.Collections
- System.Drawing
- System.IO
- System.Data
- System.Windows.Forms
- System.Web.UI
- System.Web.Services
- . . .

Compile Time



.NET: Principais vantagens

- Ambiente virtual de execução
- Muitas bibliotecas
- APIs para desenvolvimento de software para Internet
- Interoperabilidade entre linguagens
- Novo standard: C#

2. Linguagem C# 2.0

Sintaxe básica
C# vs. Java, C# vs. C++

Exemplo 1: Hello World

```
using System;

public class HelloWorld
{
    public static void Main(string[] args)
    {
        Console.WriteLine("Hello world!");
    }
}
```

Exemplo 2

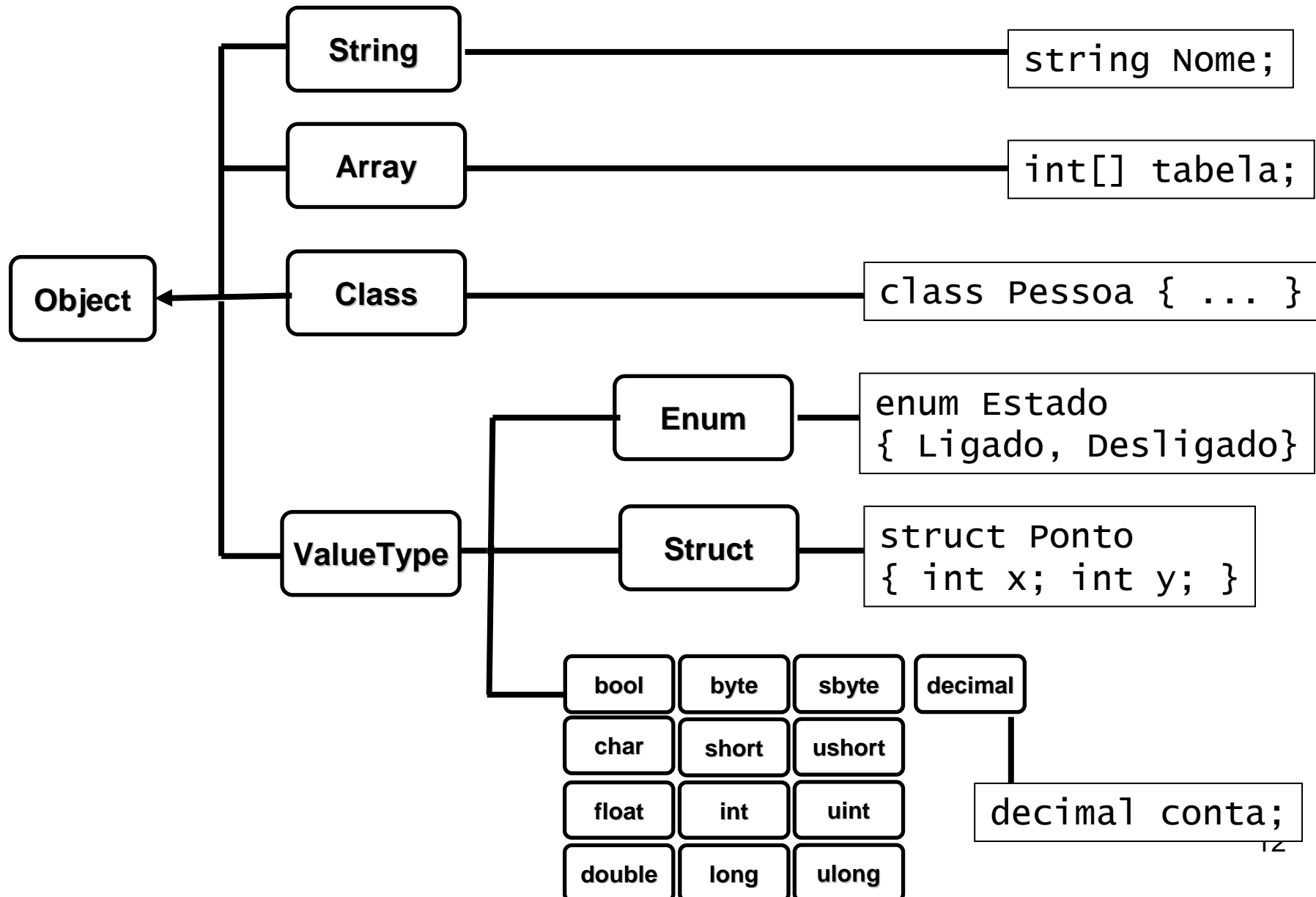
```
public class Pessoa
{
    private string    nome;
    private int       idade;

    public Pessoa(string nome, int idade) {
        this.nome = nome;
        this.idade = idade;
    }

    public void MostraInfo() {
        Console.WriteLine("{0}, tem {1} anos",
                           nome, idade);
    }
}

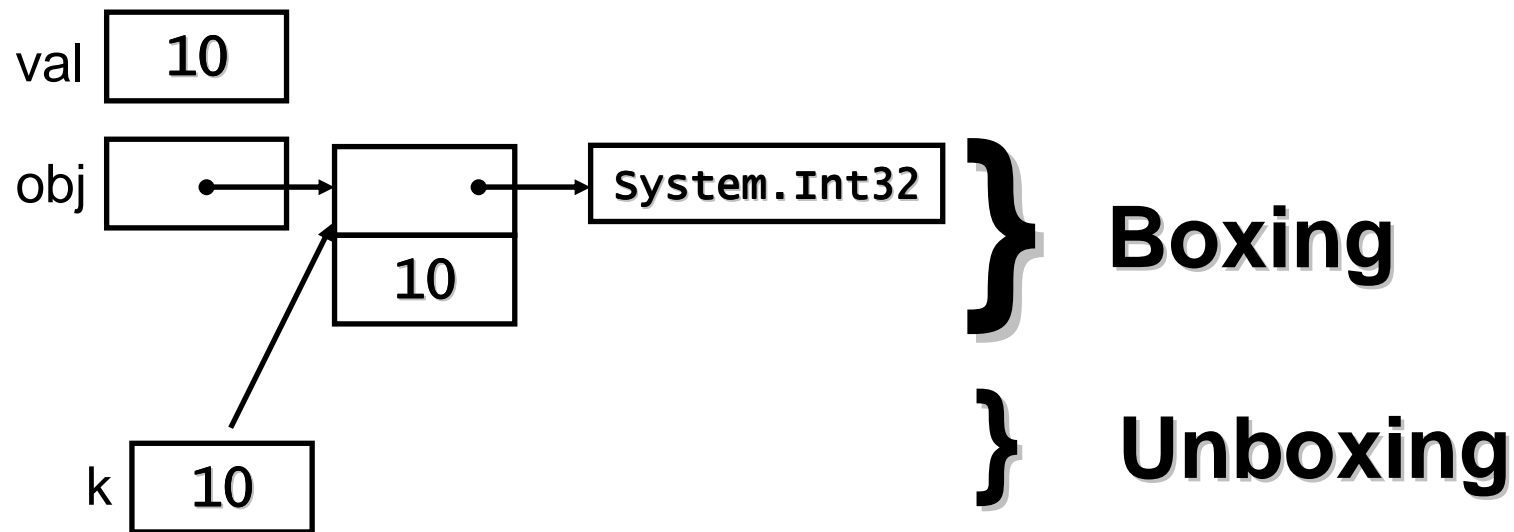
Pessoa cliente = new Pessoa("Carlos", 25);
cliente.MostraInfo();
```

C# : Sistema de tipos



C#: Tudo é um objecto (2)

```
int val = 10;  
object obj = val;  
int k = (int) obj;           // k fica c/ valor 10
```



C# : Controlo de execução

- Controlo de execução
 - if, for, do, while, switch, foreach...
 - switch sem fall-through:

```
switch a {  
case 2:  
    x = 4;  
    goto case 3  
// fall-through explícito  
case 3:  
    ...  
}
```

- Iteração (foreach):

```
int[] tabela = {1, 2, 3};  
  
foreach (int i in tabela)  
    Console.WriteLine("{0}", i);
```

C#: Types (a.k.a. Classes)

- Hierarquia de nomes – namespaces
- Herança simples
- Podem implementar múltiplas interfaces
- Membros
 - campos, métodos (incluindo construtores), propriedades, indexadores, eventos
 - Controlo de acesso: *public*, *protected*, *internal*, *private*
 - *internal*: visível apenas dentro da assembly em que foi definido
 - Membros static e instance
 - Abstract (para polimorfismo)

C# : Herança

```
public class Pessoa
{
    private string nome;

    public Pessoa(string nome) {
        this.nome = nome;
    }

    public virtual void MostraInfo()
    {
        Console.WriteLine("Nome:{0}",
            nome);
    }
}
```

```
public class Empregado : Pessoa
{
    private string empresa;

    public Empregado(string nome,
        int empresa)
        : base(nome)
    {
        this.empresa = empresa;
    }

    public override void MostraInfo() {
        base.MostraInfo();
        Console.WriteLine("Empresa: {0}",
            empresa);
    }
}
```

- Por omissão, os métodos não são virtuais!

C#: Passagem de parâmetros

- **ref** – passagem de tipos-valor por referência

```
{...
```

```
    char c='c';
```

```
    g(ref c);
```

```
}
```

```
void g(ref char pc) {  
    pc ='x'; }
```

- **out** – passagem de tipos-valor não inicializados por referência:

```
{...
```

```
    int x;
```

```
    f(out x);
```

```
}
```

```
void f(out int x) {  
    x=2; }
```

C# : Passagem de parâmetros

- **params** – passagem de n° variável de parâmetros

```
public static void Main()
{
    UseParams(1, 'a', "test");
    int[] myarray = new int[3] {10,11,12};
    UseParams(myarray);
}
```

```
public static void UseParams(params object[] list)
{
    for ( int i = 0 ; i < list.Length ; i++ )
        Console.WriteLine(list[i]);
    Console.WriteLine();
}
```

C# : Redefinição de operadores

- É possível redefinir os operadores existentes.

```
Lista A = new Lista();  
Lista B = new Lista();  
  
A.Add(1);  
A.Add(2);  
  
B.Add(3);  
B.Add(4);  
  
Lista C = A + B;           // Junta ambas as listas
```

C#: Redefinição de operadores (2)

```
public class Lista
{
    private object[] Elementos;

    ...
}
```

```
    public static Lista operator+(Lista a, Lista b)
    {
        Lista resultado = new Lista();

        // Copia os elementos de <a> e <b> para
        // a lista <resultado>

        return resultado;
    }
}
```

```
Lista A = new Lista();
Lista B = new Lista();

...

Lista C = A + B;
```

C# : Código *unsafe*

- Suporte de elementos avançados, como a utilização de ponteiros
- Sempre que são utilizados estes elementos, o código tem de ser colocado dentro de um contexto *unsafe*

```
int total = 0;

unsafe
{
    int* ptr = &total;

    *ptr = 10;
}
```

```
public unsafe
void FastCopy(byte* dst, byte* src,
              int count)
{
    for (int i=0; i<count; i++)
        *dst++ = *src++;
}
```

C#: Documentação em XML

```
/// <summary>
/// Este método calcula o ordenado de uma pessoa,
/// baseado nos seus dias de trabalho.
/// </summary>
///
/// <param name="diasTrabalho">
/// O número de dias que trabalhou.
/// </param>
///
/// <returns> O salário da pessoa. </returns>
public int CalculaOrdenado(int diasTrabalho)
{
    ...
}
```

C# vs. C++

- GC destrói objectos inacessíveis
- Tipos de referência e tipos-valor
- *Boxing, unboxing*
- Redefinição de métodos tem de ser explícita
- boolean não são inteiros
- switch sem *fall-through*
- Não se podem usar variáveis sem atribuição (*out*)
- Não há métodos globais

C# vs. Java

- Várias classes num ficheiro
- namespaces em vez de packages
- goto
- foreach
 - Java suporta iterações (com *for*) apenas a partir da versão 1.5
- Redefinição de operadores
- Enumerações
 - Java suporta enumerações apenas a partir da versão 1.5
- Tipos-valor e *boxing*
 - Java lida automaticamente com *boxing* e *unboxing* apenas a partir da versão 1.5
- Código inseguro (*unsafe*)
- ref serve para passar por referência
- Geração de um executável (.exe) ou de uma biblioteca (.dll)

3. IDE: MS Visual Studio 2005/2008

Ferramentas
Console / Win Form Applications

IDE

- Ambiente de desenvolvimento
 - .Net Framework 2.0
 - Visual Studio .NET 2005/2008
- Ferramentas
 - Editor
 - Compilador
 - Debugger
- Projectos
 - Console application
 - Windows application
 - Class library
 - ASP .Net web service
 - ASP .Net web application
 - ...