C# Object Oriented Programming

GV: Bùi Quang Đăng

Contents

1 Review

2 Class and Object, Memory

3 Method, Properties

4 Overload Method

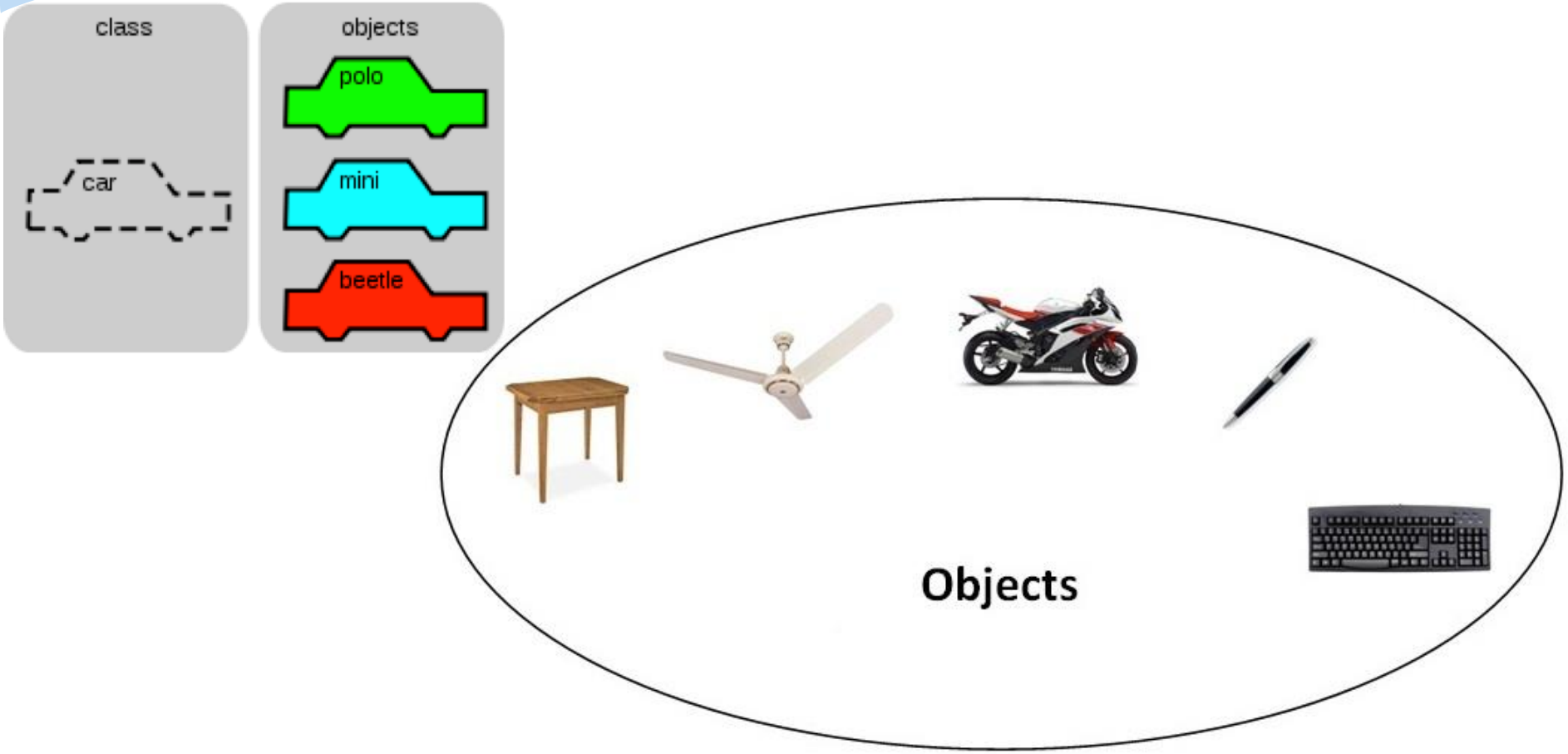
Review

- .NET Framework
  - Common Language Runtime
- Data type, Operators
- Statements
  - If…Else
  - Switch…Case
  - For, foreach, while, do…while
- Array, List
  - Array
  - List, ArrayList

Stanford
nâng tầm trí thức

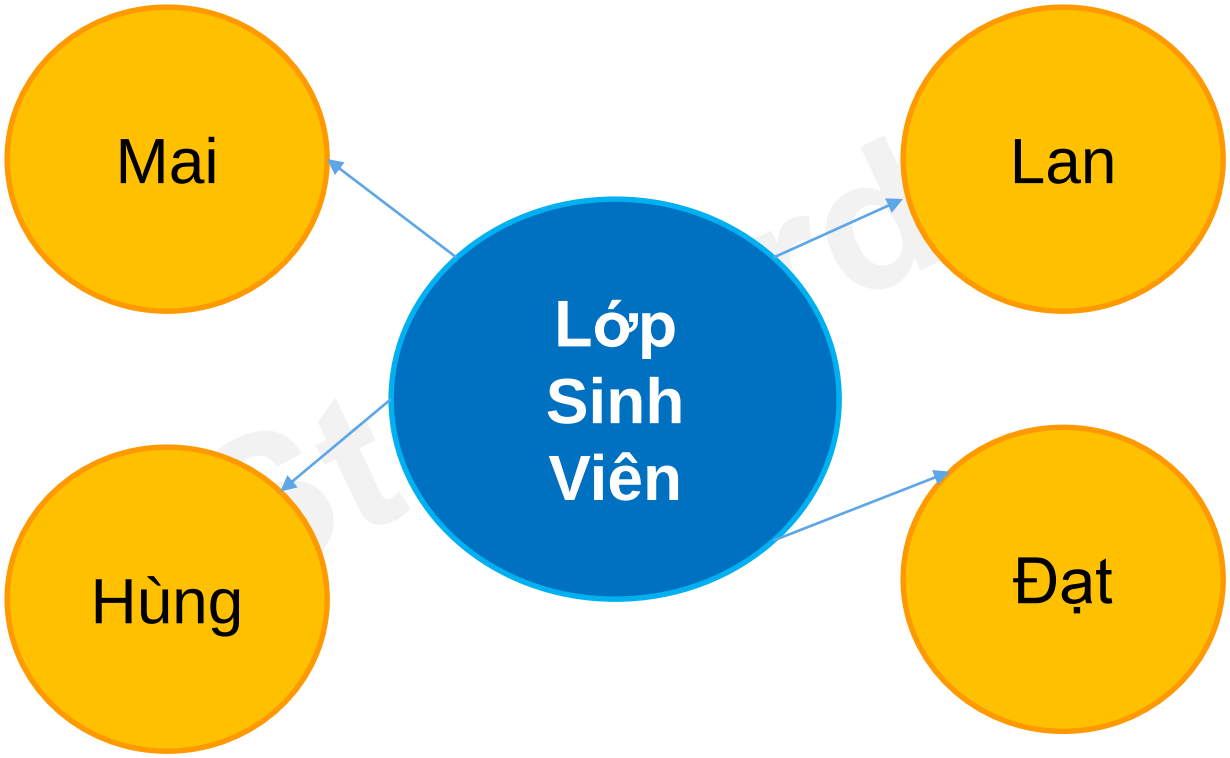
Class and Object

Class and Object



Class and Object

Class



Class and Object

1

Lớp là tập hợp các đối tượng có cùng thuộc tính

2

Các phương thức, thuộc tính, biến

3

Đối tượng là tập hợp các thuộc tính mô tả đối tượng đó.

Class and Object

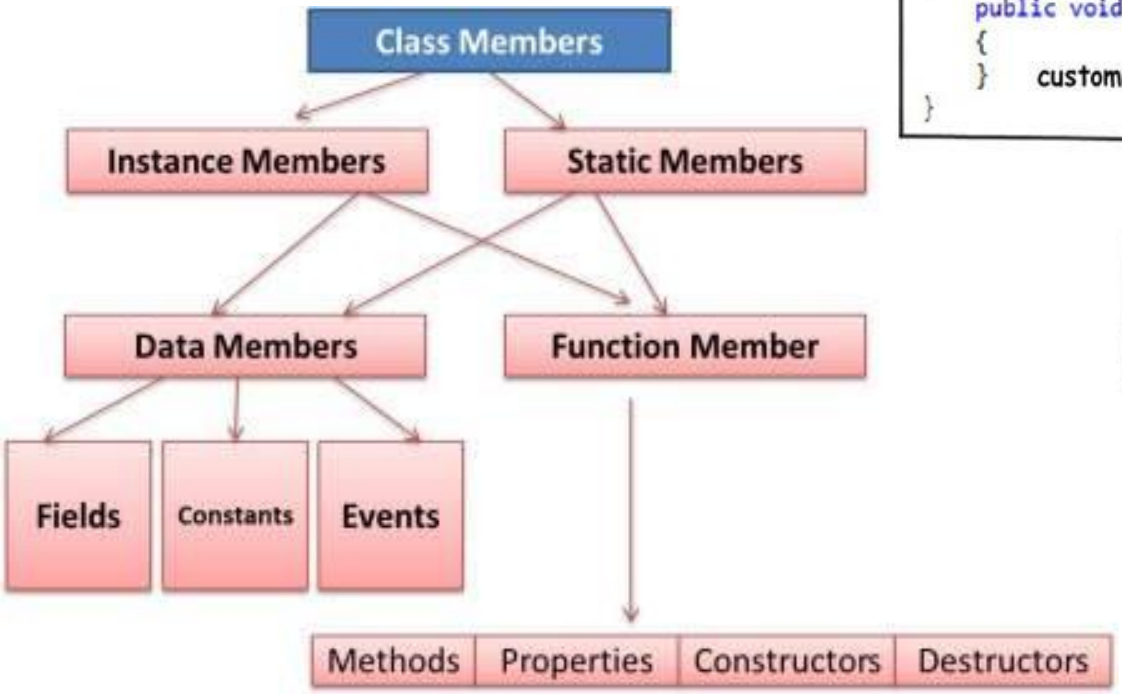
Class

- Lớp là một khuôn mẫu hay template để tạo ra các đối tượng
- **Cú pháp:**

```
[Phạm vi truy cập][Thuộc tính] class <Định danh lớp>[:Lớp cơ sở]
{
        // Biến
        // Thuộc tính
        // Phương thức
}
```

Class and Object

Class




```
public partial class Customer
{
    public void Add()
    {
    }
}
```
customer1.cs

```
public partial class Customer
{
    public void Delete()
    {
    }
}
```
customer2.cs

```
Customer obj = new Customer();
obj.
```
Add
Delete
void Custo

At the end of the day they are one class

Class and Object

Class

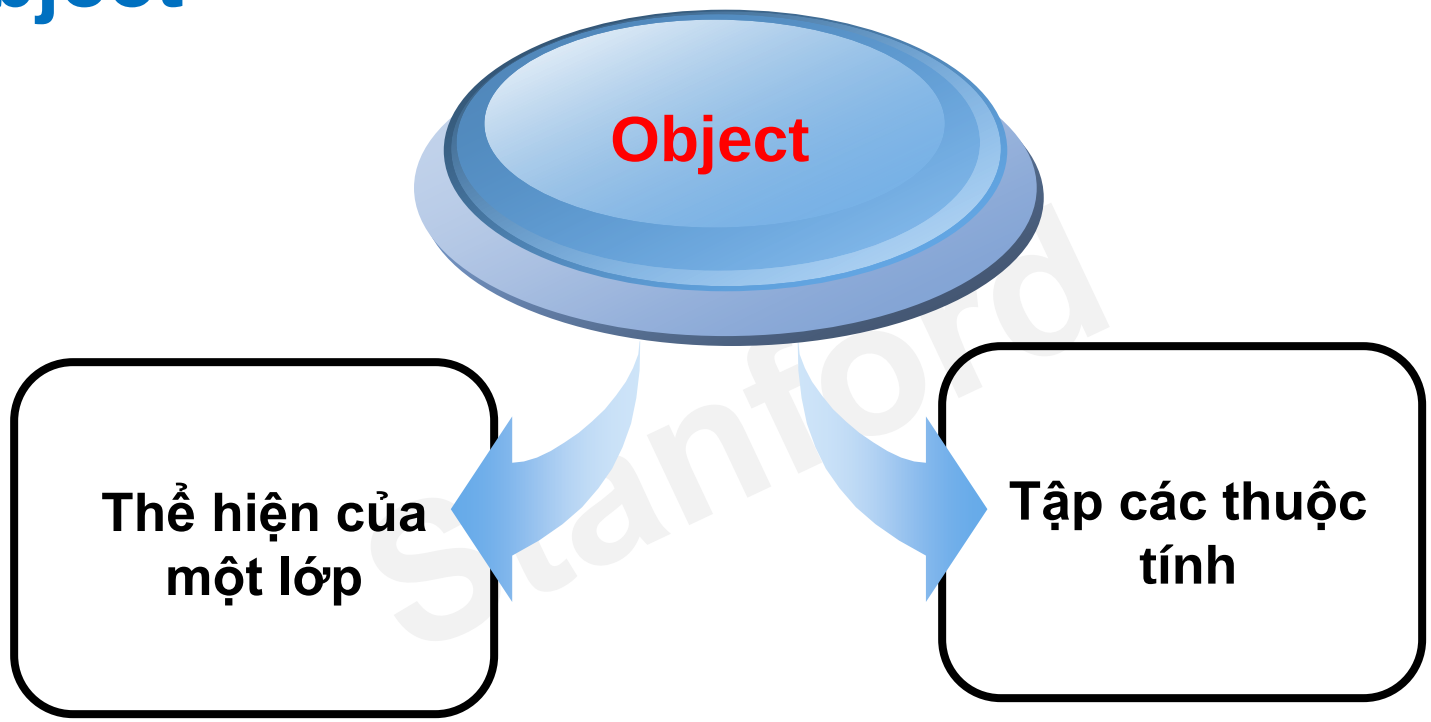
- **Ví dụ**: Tạo ra một lớp sinh viên bao gồm các thuộc tính sau:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace WinAppControlsAdvanced
{
    class SinhVien
    {
        public int Id { get; set; }
        public string HoTen { get; set; }
        public int GioiTinh { get; set; }
        public DateTime NgaySinh { get; set; }
        public string DienThoai { get; set; }
        public string Email { get; set; }
        public string QueQuan { get; set; }
    }
}
```

Class and Object

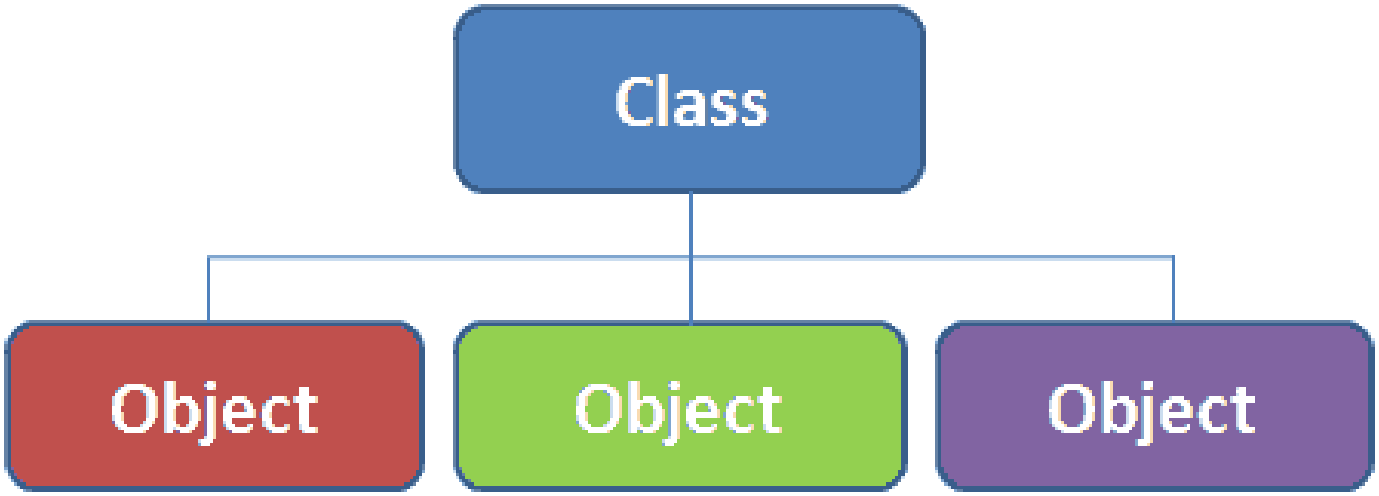
Object



Class and Object

Object

- Là một thể hiện cụ thể của lớp bao gồm tập hợp các thuộc tính mô tả đối tượng đó.



Class and Object

Object

- Khai báo một đối tượng
  - [Thuộc tính] [Bổ tử truy xuất] **[Tên lớp]** [Đặt tên đối tượng]
- Khởi tạo một đối tượng mới
  - [Đặt tên đối tượng] = **new** [Tên lớp]()
  - Sử dụng từ khóa new để khởi tạo một đối tượng mới.

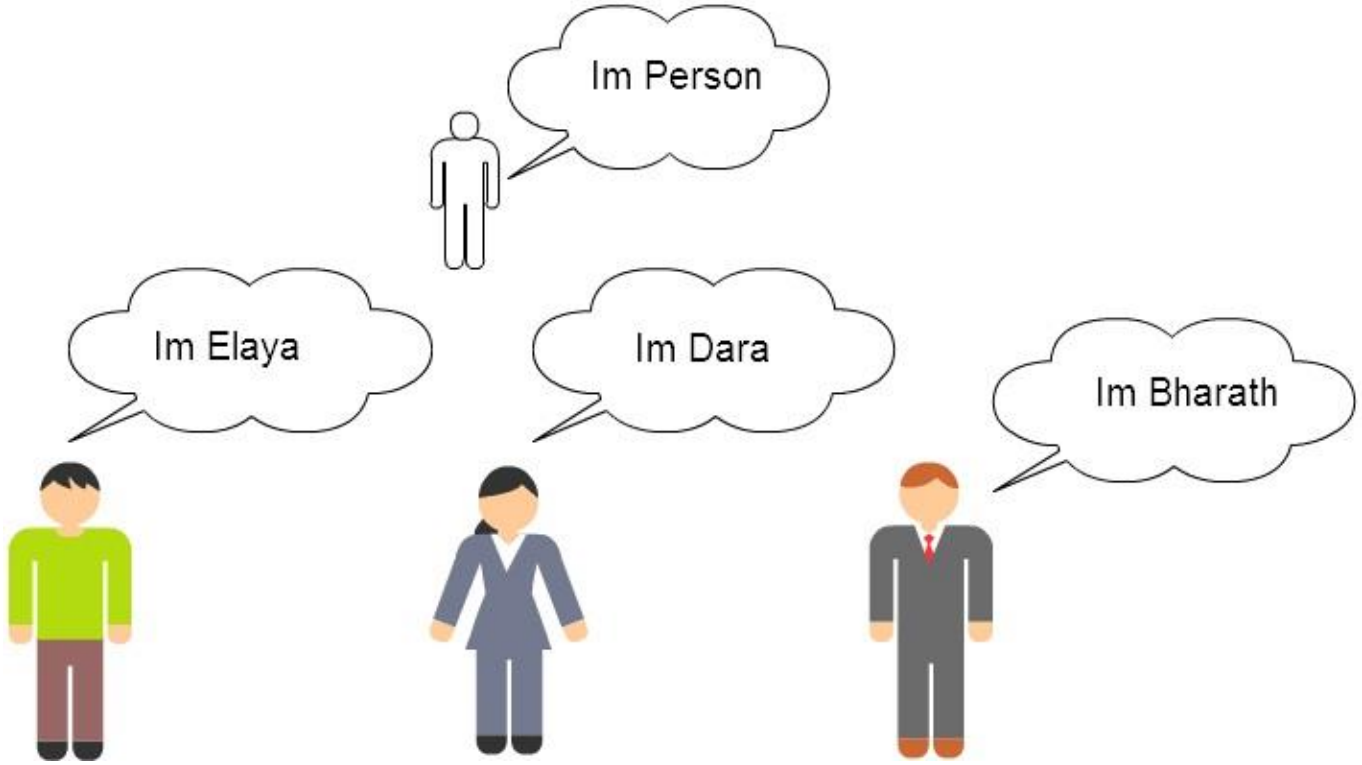
Ví dụ:

```
TinhToan Obj1 = new TinhToan();
SinhVien Obj2 = new SinhVien();
Person Obj3 = new Person();
```

Class and Object

Object

- Là một thể hiện cụ thể của lớp bao gồm tập hợp các thuộc tính mô tả đối tượng đó.



Class and Object

Object

- Ví dụ:

```
// Tạo sinh viên
SinhVien objKien = new SinhVien();
```

Tên đối tượng

Hàm khởi tạo không có đối số của lớp

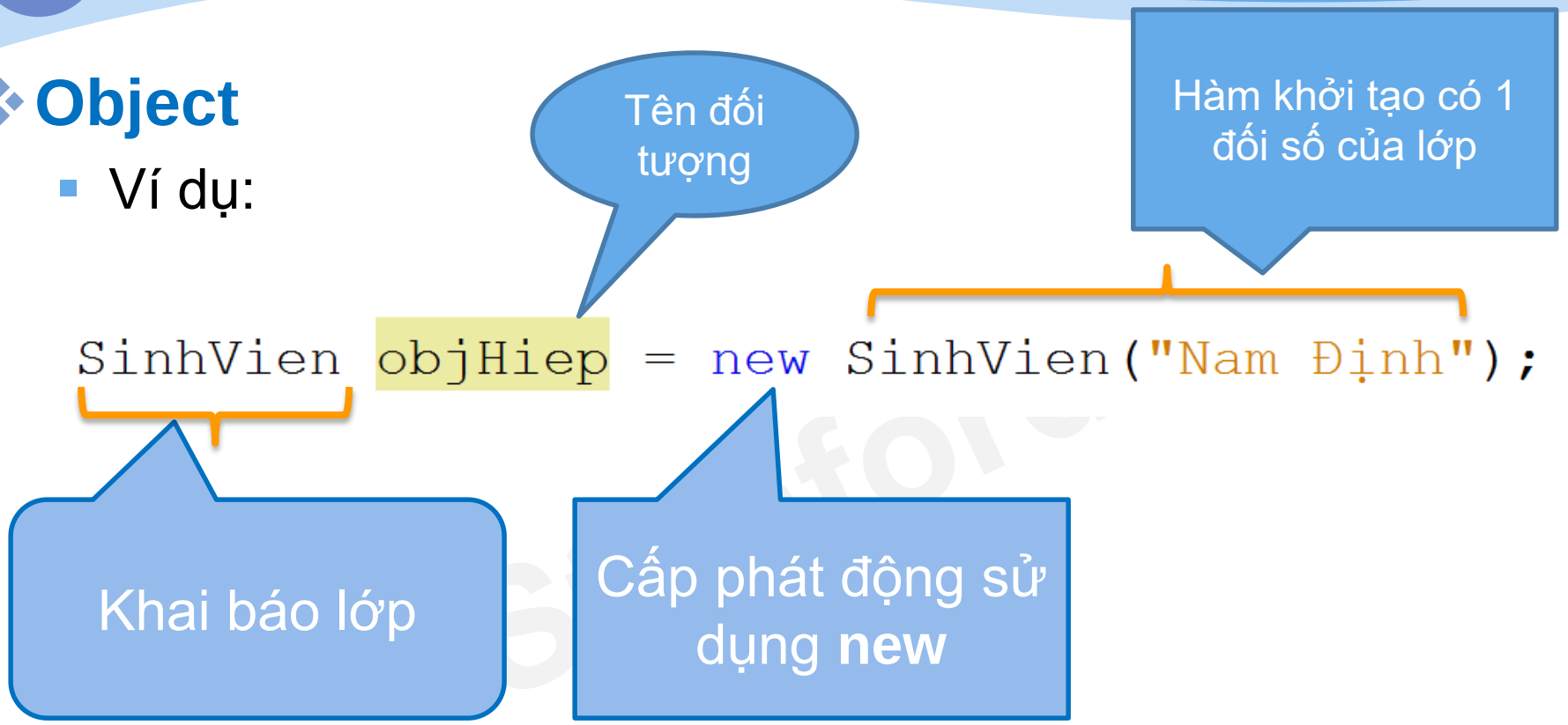
Khai báo lớp

Cấp phát động sử dụng **new**

Class and Object

Object

- Ví dụ:



Class and Object

Object

- Ví dụ:

```
public class SinhVien {

    private String maSV;

    public SinhVien(){
        diaChi = "Hà Nội";
    }

    public SinhVien(String diaChi){
        this.diaChi = diaChi;
    }
```

Hàm khởi tạo của lớp

Scope Access

Scope Access

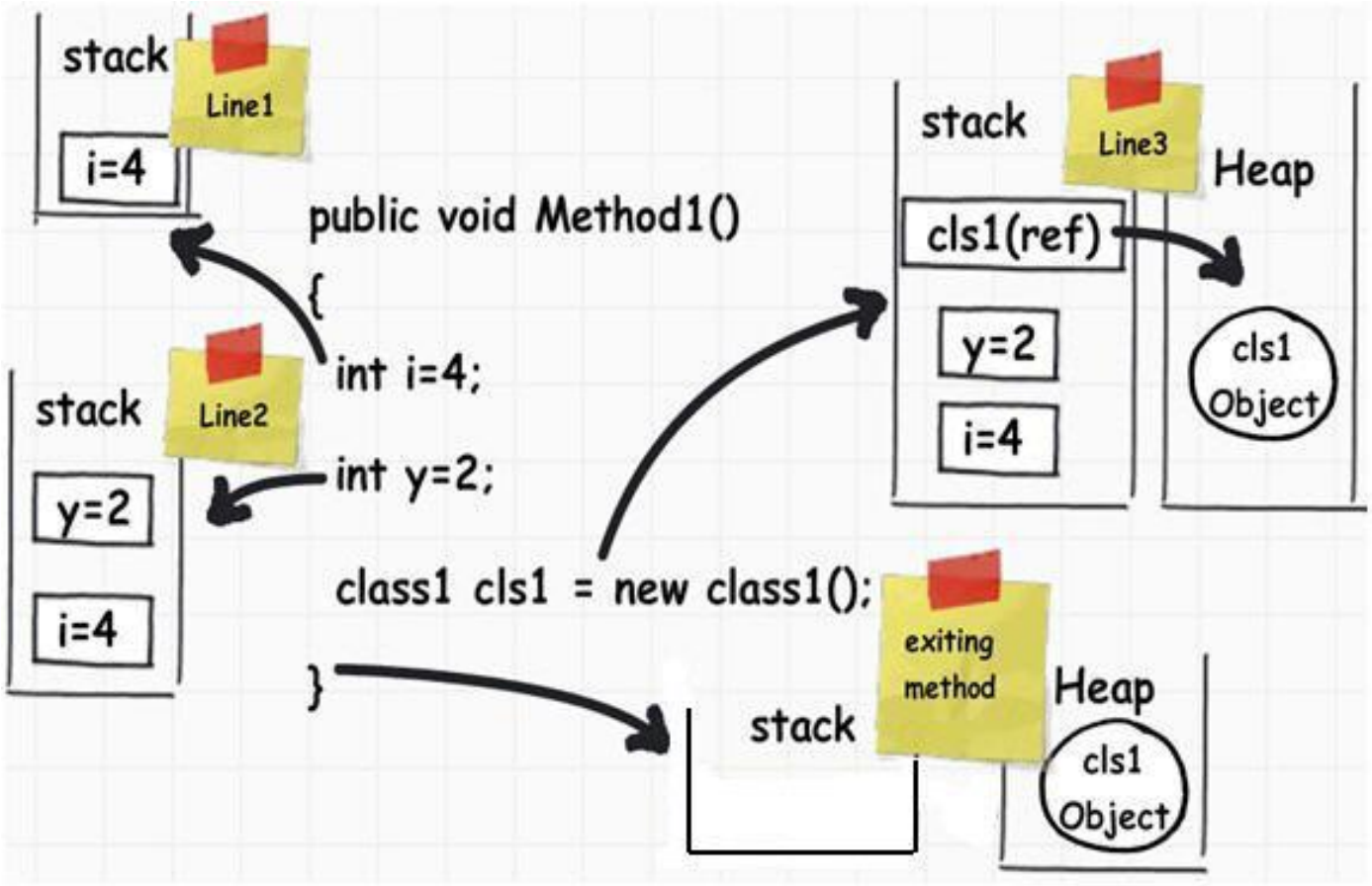
Properties	Access modifies
Public	+ Không hạn chế + Sử dụng bất kỳ phương thức của lớp và đối tượng khác
Private	+ Truy cập với các phương thức của chính lớp đó
Protected	+ Được truy cập bởi các phương thức của lớp đó. + Và những lớp đối tượng kế thừa từ nó mới sử dụng được.
Internal	+ Được truy cập bởi những phương thức của bất cứ lớp nào nhưng phải trong cùng một khối hợp ngữ (Assembly).
Protected Internal	+ Được truy cập bởi các phương thức của lớp đó. + Các lớp đối tượng kế thừa từ lớp đó. + Và bất cứ lớp nào trong cùng khối hợp ngữ.

Scope Access

visibility / keyword	Containing Classes	Derived Classes	Containing Assembly	Anywhere outside the containing assembly
public	yes	yes	yes	yes
protected internal	yes	yes	yes	no
protected	yes	yes	no	no
private	yes	no	no	no
internal	yes	no	yes	no

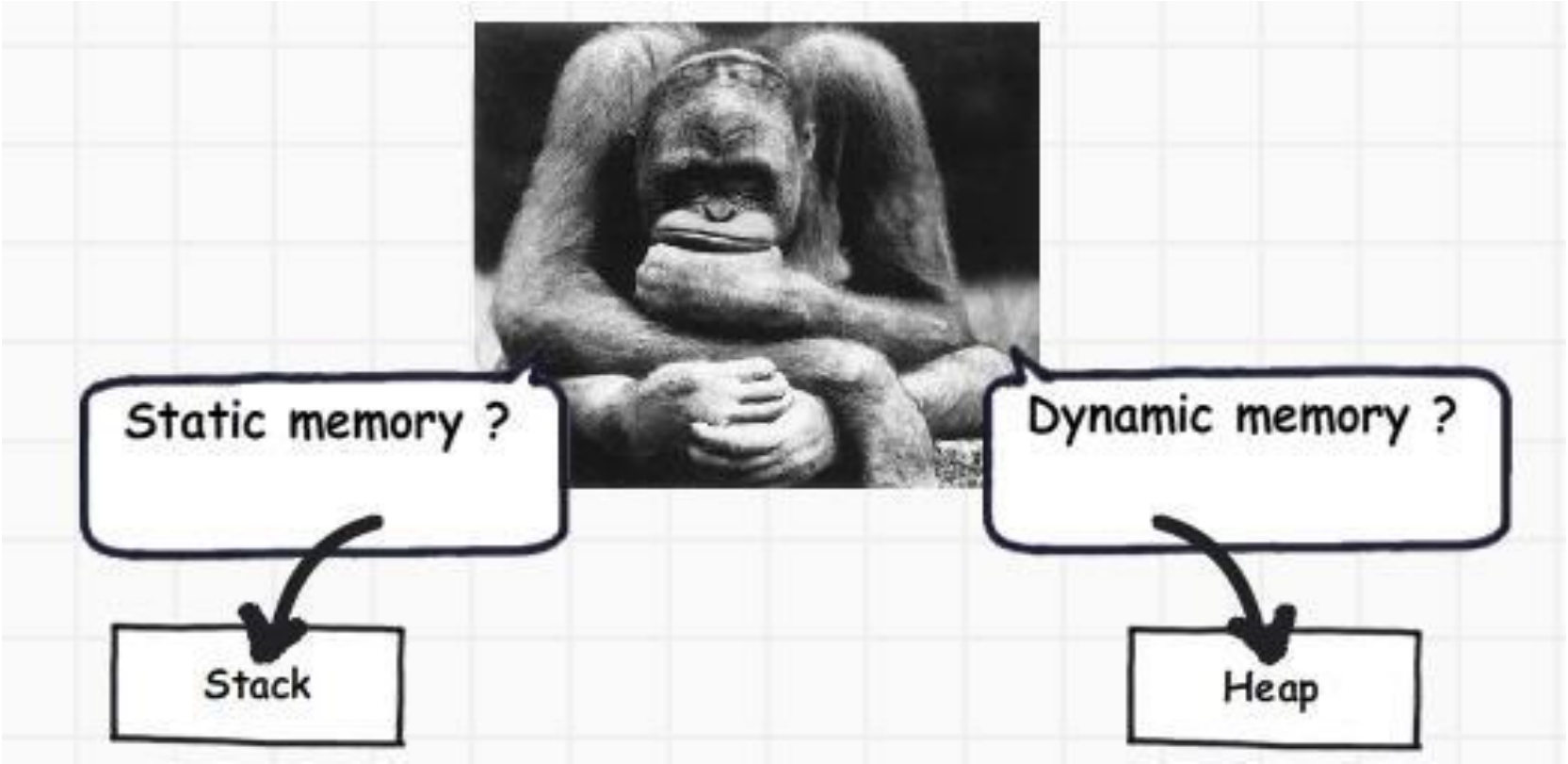
Class and Object

Stack and heap



Class and Object

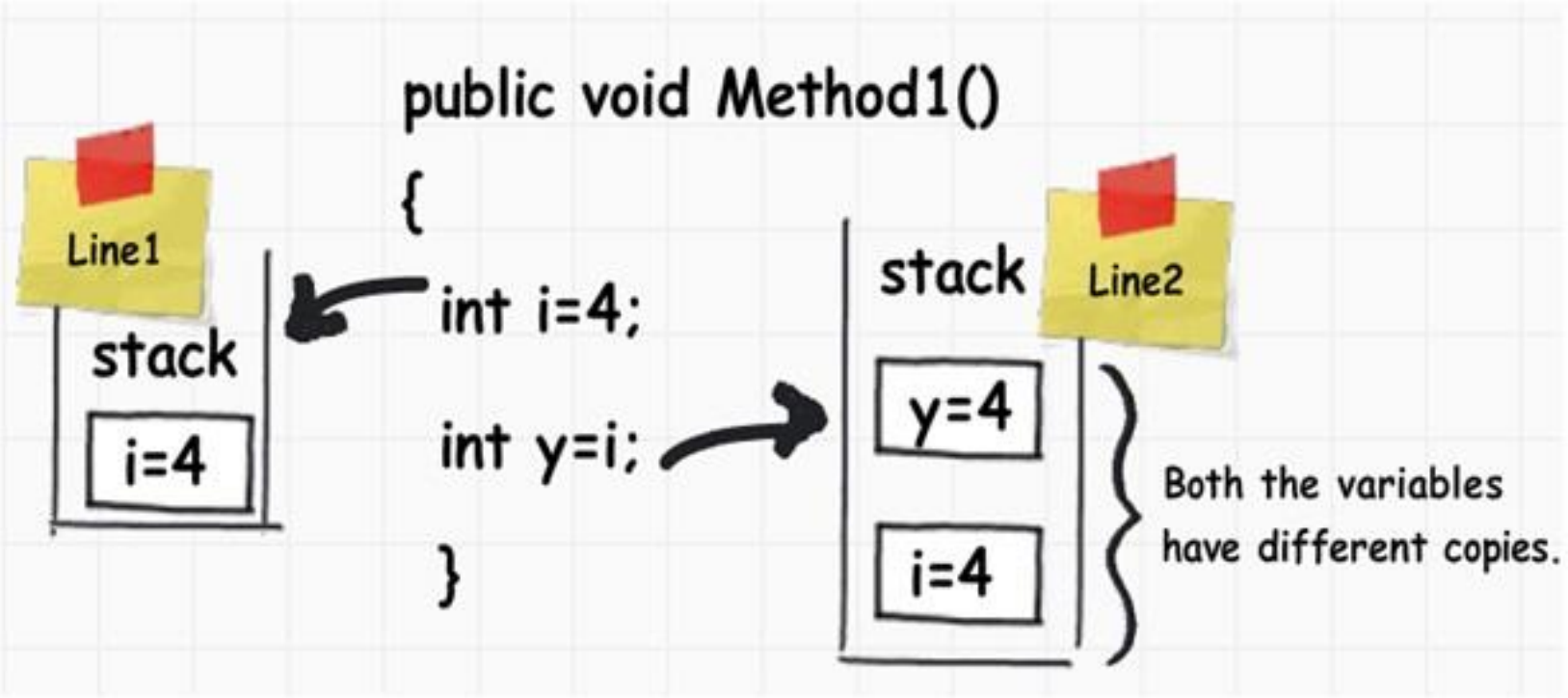
❖ Stack and heap



Class and Object

Stack and heap

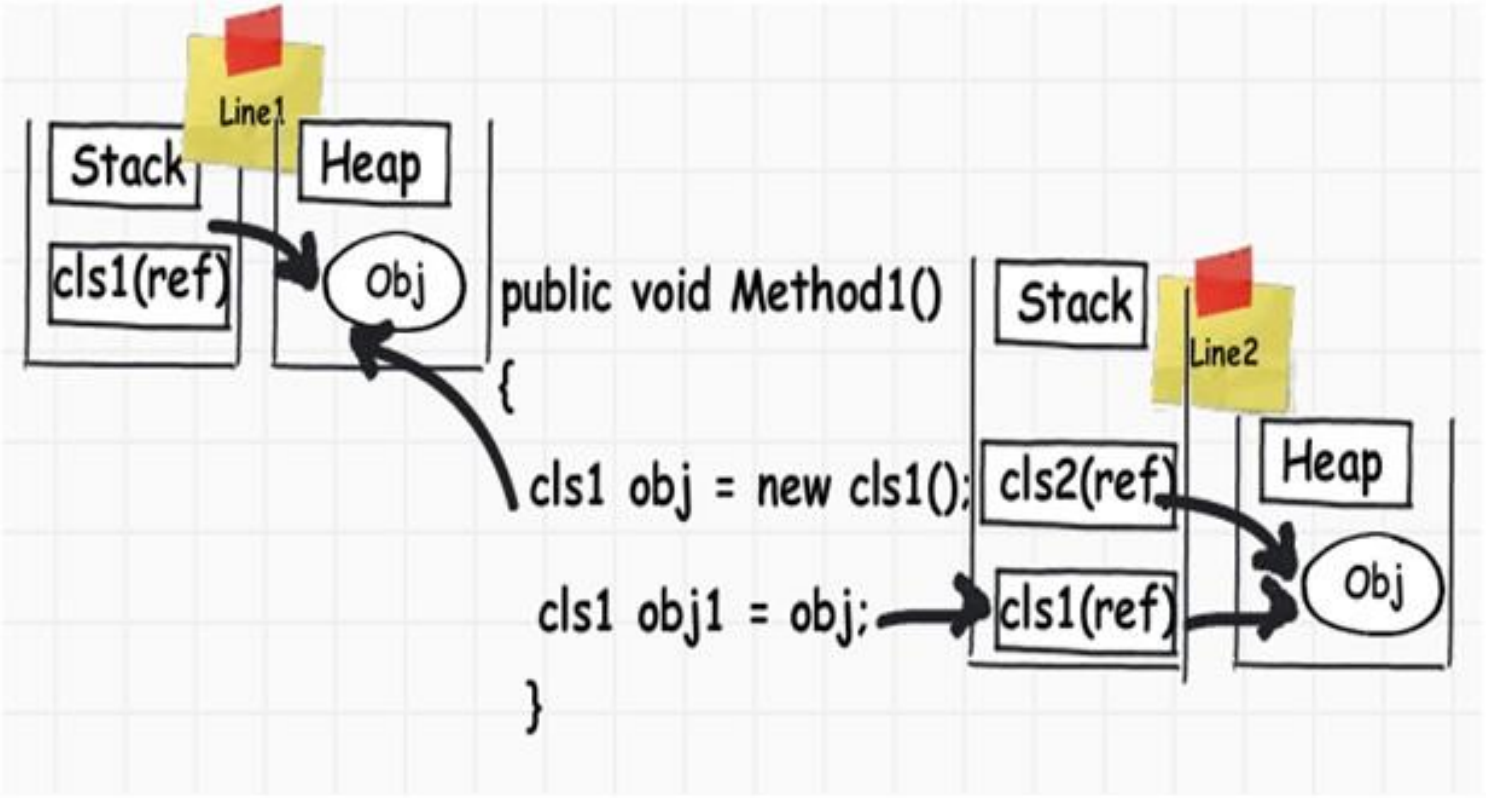
- **Value Types** chứa cả dữ liệu và bộ nhớ trên cùng một vị trí



Class and Object

Stack and heap

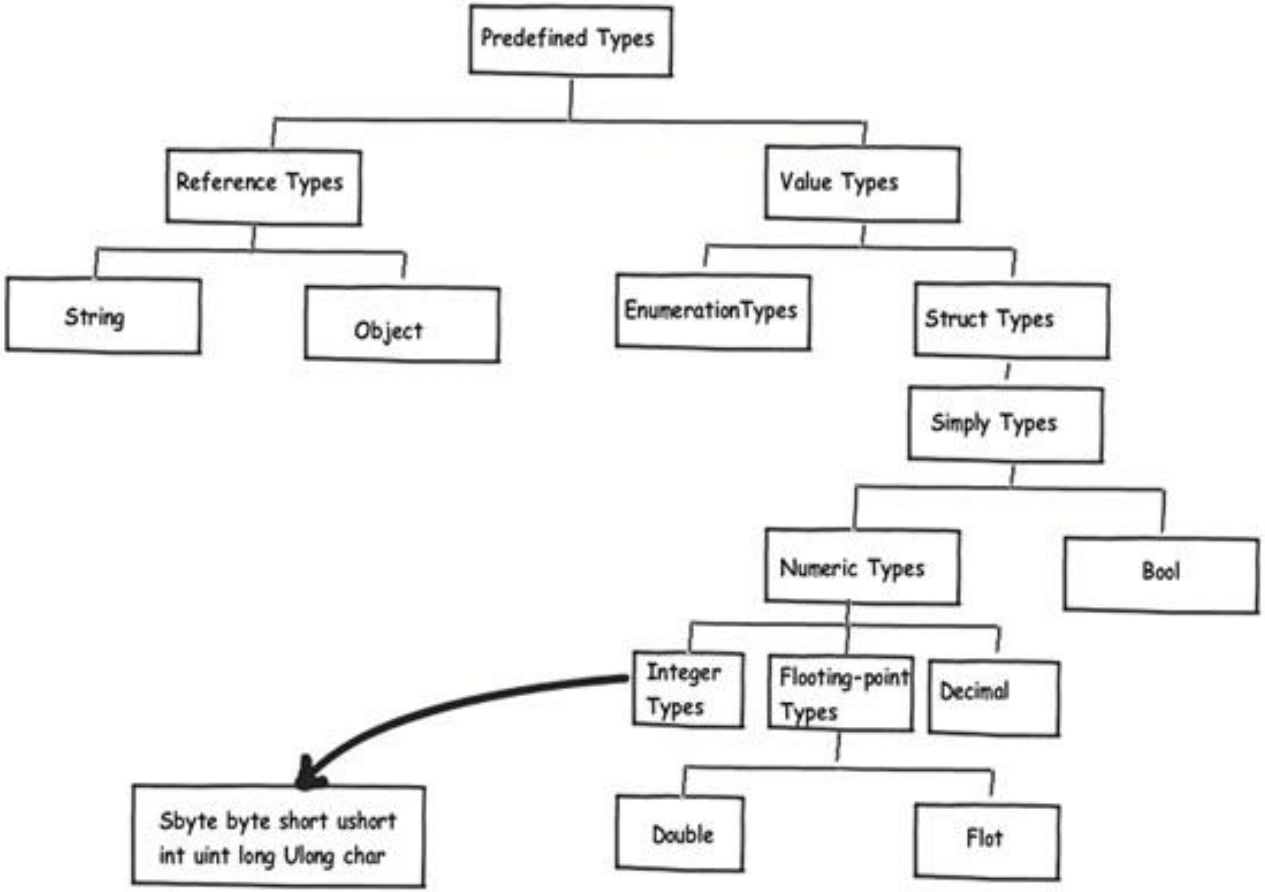
- **Reference Types** có một con trỏ trỏ đến vị trí bộ nhớ.



Stanford
nâng tầm trí thức

Class and Object

Stack and heap



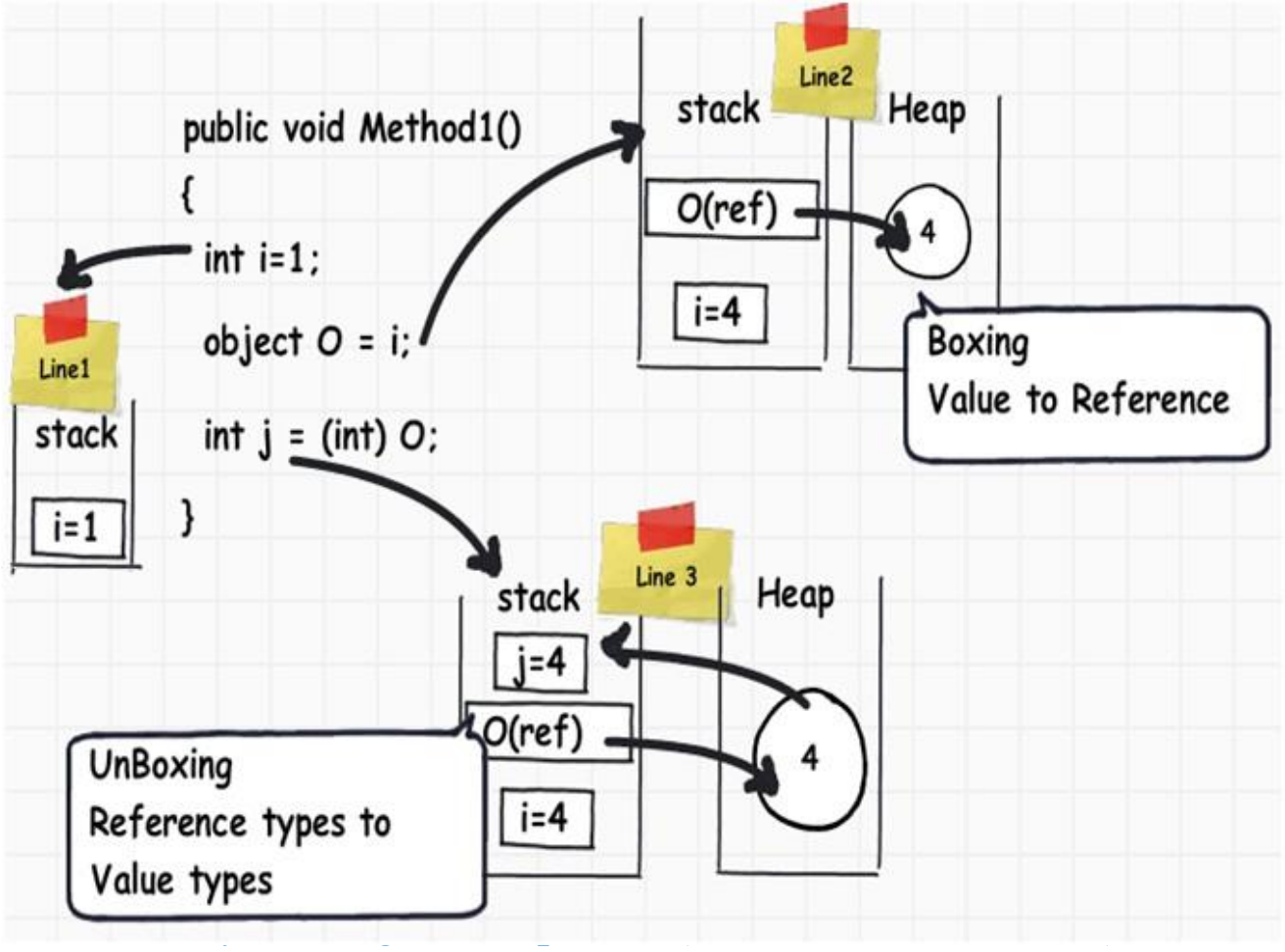
Class and Object

Boxing and unboxing

- **Boxing** là cơ chế chuyển dữ liệu từ value types sang Reference types.
- **Unboxing** là cơ chế chuyển dữ liệu từ Reference types sang Value types.

Class and Object

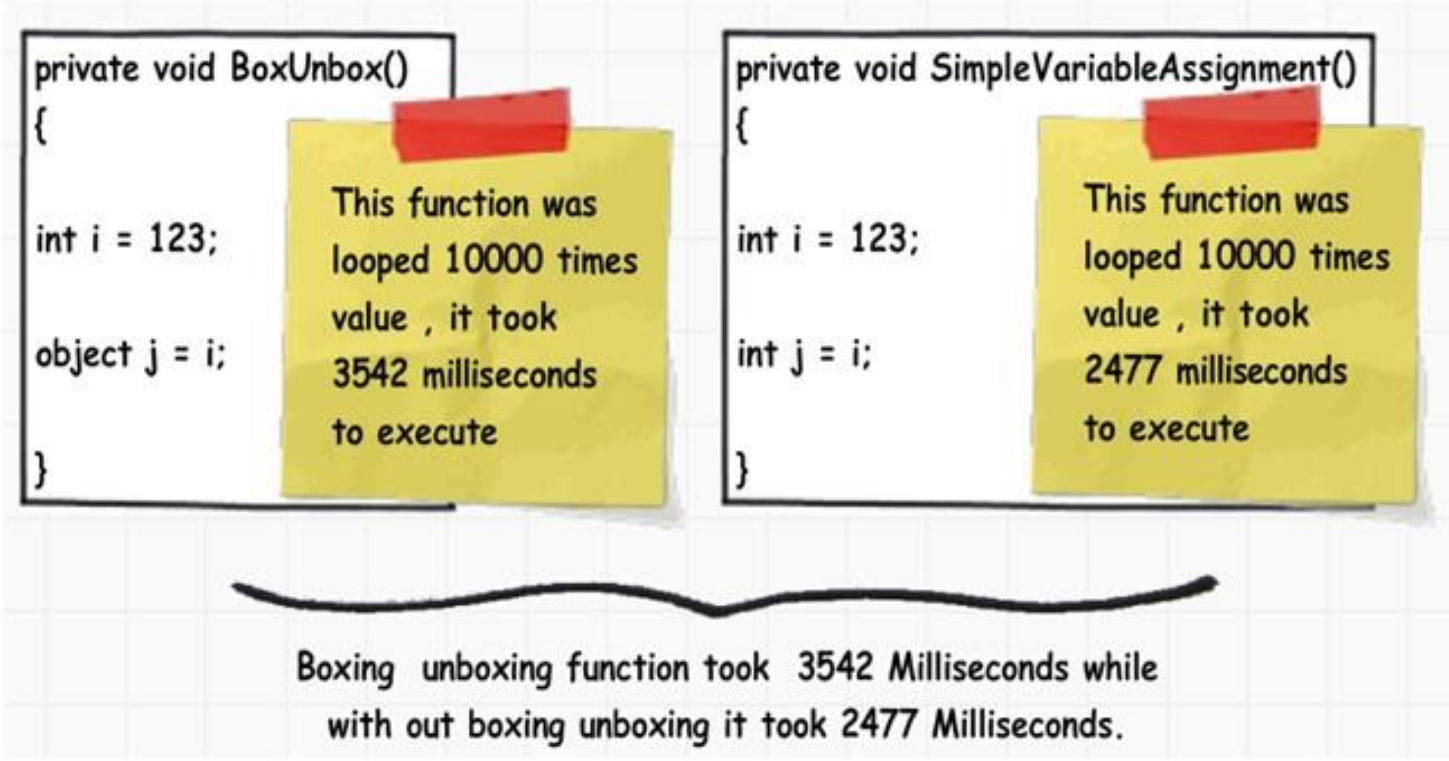
Boxing and unboxing



Class and Object

Boxing and unboxing

- **Performance implication of boxing and unboxing**



Stanford
nâng tầm tri thức

Methods and Properties

Methods

- Là một đơn vị chương trình độc lập dùng để thực hiện một phần việc nào đó như: Nhập số liệu, in kết quả hay thực hiện một số phép tính toán.
- **Cú pháp**
  - [Phạm vi truy cập] [Thuộc tính] **Tên phương thức**([Tham số])

```
{
  // Nội dung xử lý
}
```

Methods

- **Ví dụ**:

```
public float TinhTong(float a, float b)
{
    float tongso=0;
    Tongso = a + b;
    return tongso;
}
```

Tên hàm

Danh sách các tham số của hàm

Kiểu dữ liệu trả về

Phạm vi truy cập

Nội dung xử lý của hàm

Methods

Ví dụ:

```
/// <summary>
/// Hàm để tạo dữ liệu mẫu ban đầu trong hệ thống
/// </summary>
1 reference
public void taoDuLieuBanDau()
{
    //Khai báo 1 đối tượng sinh viên
    SinhVien objNgoThinh = new SinhVien(); //Gọi đến hàm khởi tạo của lớp không có tham số,
       contructor

    //Gán các giá trị cho thuộc tính
    objNgoThinh.MaSV = "SF001";
    objNgoThinh.HoTen = "Ngô Văn Thịnh";
    objNgoThinh.DienThoai = "0988234568";
    objNgoThinh.Email = "vanthinh@gmail.com";

    //Thêm vào danh sách
    lstSinhVien.Add(objNgoThinh);
```

Hàm không có đối số

Gán giá trị cho các thuộc tính của lớp

Properties

- Là các thuộc tính mô tả các đặc điểm của lớp
- Thuộc tính cung cấp khả năng đọc ghi dữ liệu
  - **Cú pháp**:

```
[Từ khóa] [Kiểu dữ liệu] [Tên thuộc tính]
{
    get { return _bienthuoctinh; };
    set  { _bienthuoctinh=value; };
}
```

  - **get**: Lấy giá trị biến thành viên
  - **set**: Thiết lập giá trị cho biến thành viên

Properties

- Ví dụ:

```
private string _MaSV;

8 references
public string MaSV
{
    get { return _MaSV; }

    set {

        _MaSV = value;

    }
}

private string _HoTen;

6 references
public string HoTen
{
    get { return _HoTen; }
    set { _HoTen = value; }
}
```

Lấy giá trị từ thuộc tính

Thiết lập giá trị cho thuộc tính

Methods

- Từ khóa **this** đại diện cho 1 class hay 1 object mà mình đang làm việc trên nó.

Ví dụ:

```
private int Nam = 0;
public void SetYear(int Nam)
{
  this.Nam = Nam;
}
```

Methods

Từ khóa static

Biến tĩnh:

- Là biến có từ khóa static trước kiểu dữ liệu của biến.
- Biến tĩnh chỉ được khởi tạo một lần trong bộ nhớ cho tất cả các thể hiện của đối tượng đó.

  Ví dụ: static string KeyName = "";

Methods

Từ khóa static

- **Phương thức tĩnh**:
  - Là phương thức có từ khóa static trước kiểu dữ liệu trả về của phương thức đó khi khai báo phương thức.
  - Trong thân của phương thức chỉ được phép truy xuất những thành viên tĩnh (static member).

Ví dụ:

```
public static TinhTong(int a, int b){
  //Nội dung xử lý
};
```

Methods

Từ khóa static

- **Lớp tĩnh:** Là lớp có khai báo từ khóa static trước tên lớp
  - Chỉ chứa các tình viên tĩnh (They only contain static members).
  - Không có thể hiện của lớp (They cannot be instantiated). Tức là không thể dử dụng từ khóa new để tạo đối tượng của lớp tĩnh.
  - Là lớp niêm phong (They are sealed). Tức là không thể kế thừa từ lớp tĩnh.
  - Không có phương thức khởi tạo thể hiện. Tức là phương thức khởi tạo cũng phải là static.

C#.NET for Base

Overload Method

C#.NET for Base

❖ Overload Method

C# hỗ trợ phương thức nạp chồng với một vài dạng phương thức khác nhau về những đặc tính sau: tên, số lượng thông số, và kiểu thông số.

- Không chấp nhận hai phương thức chỉ khác nhau về kiểu trả về.
- Không chấp nhận hai phương thức chỉ khác nhau về đặc tính của một thông số đang được khai báo như *ref* hay *out.*

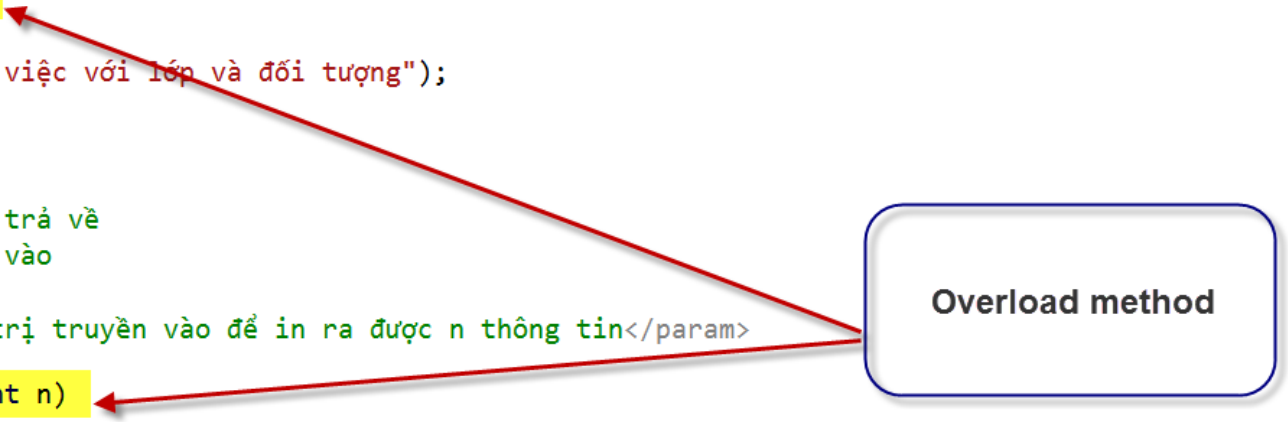
C#.NET for Base

Overload Method

Ví dụ:

```
/// <summary>
/// Hàm không có giá trị trả về và không có tham số
/// truyền vào
/// </summary>
0 references
public void inThongTin()
{
    Debug.WriteLine("Làm việc với lớp và đối tượng");
}

/// <summary>
/// Hàm không có giá trị trả về
/// và có tham số truyền vào
/// </summary>
/// <param name="n">Giá trị truyền vào để in ra được n thông tin</param>
0 references
public void inThongTin(int n)
{
    for (int i = 1; i <= n; i++)
    {
        Debug.WriteLine("I LOVE YOU " + i);
    }
}
```



Exercises