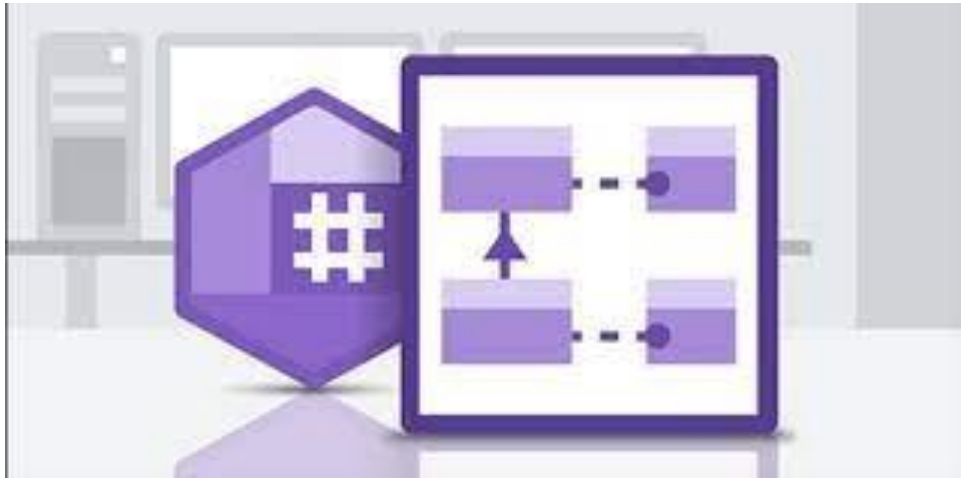
C# 10 – Không gian tên - Namespaces

Giảng viên: **ThS. Lê Thiện Nhật Quang**

Email: quangltn.dotnet.vn@gmail.com

Website: http://dotnet.edu.vn

Điện thoại: **0868.917.786**



MỤC TIÊU

- Tìm hiểu namespaces
- Giải thích namespaces lồng nhau

1.1. NAMESPACES

❑Namespace được sử dụng để chia các class và interface thành từng gói khác nhau.

❖Việc làm này tương tự quản lý file trên ổ đĩa trong đó class (như file) và namespace (như folder)

❑Mục đích sử dụng namespace là phân hoạch không gian các định danh, các kiểu dữ liệu thành những vùng dễ quản lý hơn, nhằm tránh sự xung đột giữa việc sử dụng các thư viện khác nhau từ các nhà cung cấp

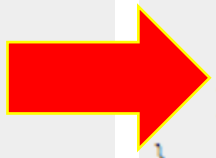
❑Ví dụ: lớp MyClass thuộc gói com.fpt

```
Namespace com.fpt;
public class MyClass{...}
```

1.1. NAMESPACES (2)

Ví dụ, các lớp bên đây mô tả hai dòng sản phẩm của hai hãng khác nhau:

```
class TiviSamsung
{
    ...
}
class SamsungWalkMan
{
    ...
}
class TiviSony
{
    ...
}
class SonyWalkMan
{
    ...
}
```



```
namespace Samsung
{
    class Television
    {
        ...
    }
    class WalkMan
    {
        ...
    }
}
namespace Sony
{
    class Television
    {
        ...
    }
    class Walkman
    {
        ...
    }
}
```

Việc đặt tên như ví dụ bên là khá dài và có thể gây khó khăn cho vấn đề bảo trì mã lệnh. Đoạn code bên đây có thể khắc phục được nhược điểm trên.

1.2. KHAI BÁO NAMESPACE

```
namespace MyLib
{
    namespace Demo1
    {
        class Example1
        {
            public static void Show1()
            {
                Console.WriteLine("Lop Example1");
            }
        }
    }
    namespace Demo2
    {
        public class Tester
        {
            public static void Main()
            {
                Demo1.Example1.Show1();
                Demo1.Example2.Show2();
            }
        }
    }
```

```
namespace NamespaceName
{
        // nơi chứa đựng tất cả các class
}
```

1.2. KHAI BÁO NAMESPACE (2)

❑C# đưa ra từ khóa using để khai báo không gian tên cho việc sử dụng các định danh, kiểu dữ liệu định nghĩa thuộc không gian tên trong chương trình

❑Ví dụ: using System;

❖ Cho phép ta sử dụng Console.WriteLine(); thay cho System.Console.WriteLine();

❑Ví dụ: using Demo1;

❖ Cho phép ta truy cập Example1.Show1(); thay cho Demo1.Example1.Show1();

1.3. USING NAMESPACES

- C# cho phép chỉ định mã định danh duy nhất cho mỗi namespace.
- Định danh này giúp truy cập các lớp trong không gian tên – namespace
- Ngoài các lớp, các cấu trúc dữ liệu sau có thể được khai báo trong namespace:

➢ **Interface:** Là kiểu tham chiếu chứa khai báo các event, indexers, method và properties. Các interface được kế thừa bởi các lớp, cấu trúc và các khai báo được triển khai trong các lớp và cấu trúc này.

➢ **Structure:** Là kiểu giá trị có thể chứa các giá trị của các kiểu dữ liệu khác nhau. Nó gồm trường, phương thức, hằng, hàm dựng, indexers, toán tử, và các cấu trúc khác

➢ **Enumeration:** Là kiểu giá trị gồm danh sách các hằng được đặt tên. Danh sách tên hằng được đặt tên là danh sách liệt kê

➢ **Delegate:** Là kiểu tham chiếu do người dùng định nghĩa đề cấp đến một hoặc nhiều phương thức. Nó có thể được sử dụng để truyền dữ liệu dưới dạng tham số cho các phương thức.

1.4. ĐẶC ĐIỂM VÀ LỢI ÍCH

Nhóm không gian tên gồm các lớp, cấu trúc hoặc giao diện, hỗ trợ các khái niệm đóng gói và trừu tượng OOP.

Không gian tên:

- Cung cấp cấu trúc phân cấp giúp xác định logic để nhóm các lớp

- Cho phép thêm nhiều class, structure, enumration, delegate và interface sau khi namespace được khai báo

- Bao gồm các lớp có tên duy nhất trong không gian tên

Một không gian tên cung cấp các lợi ích sau:

- Cho phép sử dụng nhiều lớp có cùng tên bằng cách tạo chúng trong các không gian tên khác nhau.
- Nó làm cho hệ thống chuyển thành mô-đun

1.5. CÁC KHÔNG GIAN TÊN CÓ SẴN

- .NET Framework bao gồm một số không gian tên có sẵn:

- Class
- Interface
- Structure
- Delegate
- Enumeration

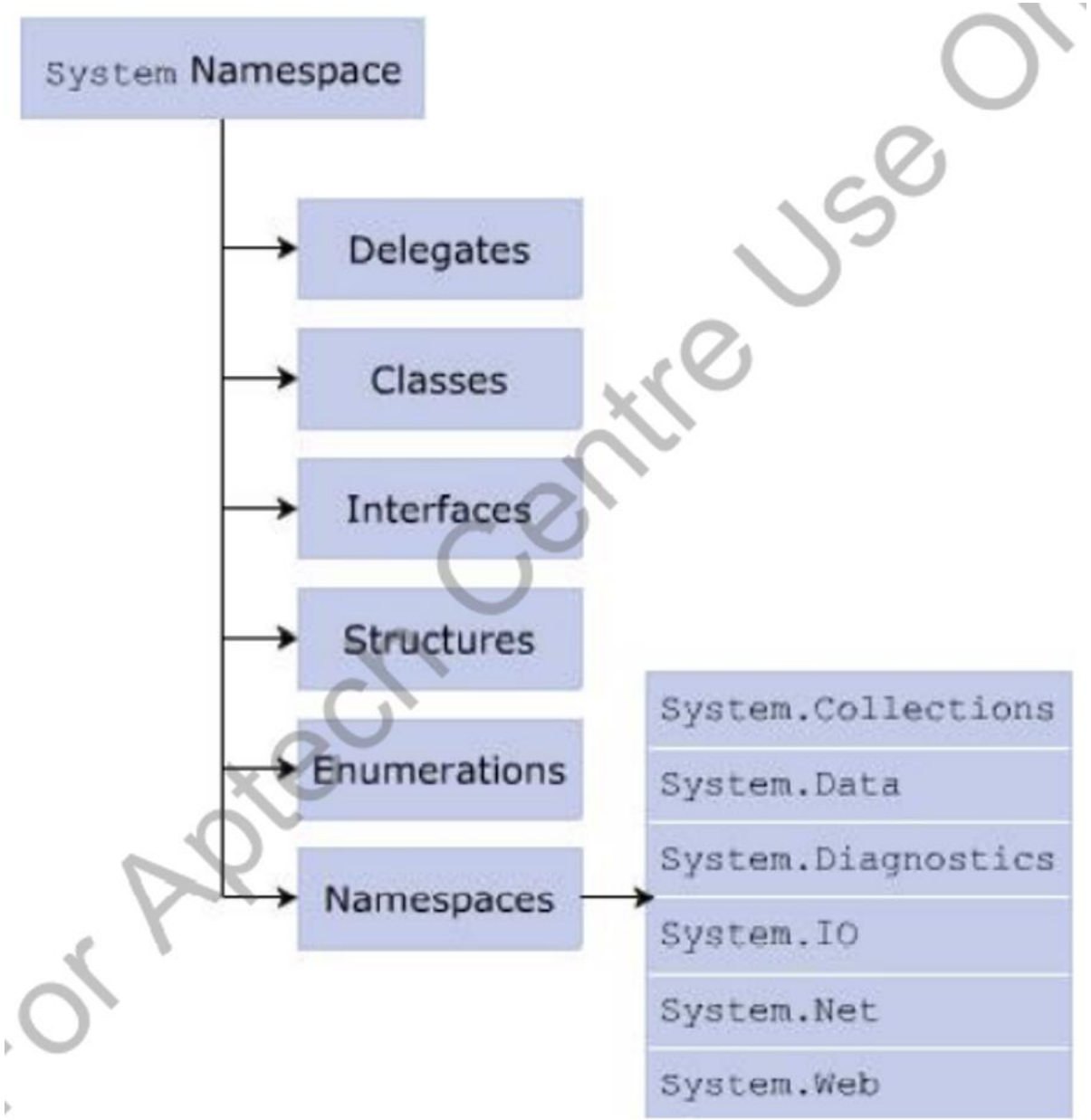
- Các không gian tên này được đề cập đến các không gian tên do hệ thống định nghĩa
- Không gian tên tích hợp được sử dụng rộng rãi nhất trong .NET là System
- Không gian tên System chứa các lớp:

- Xác định các kiểu dữ liệu giá trị và tham chiếu, interface và các không gian tên khác
- Cho phép tương tác với hệ thống, gồm các thiết bị đầu vào và đầu ra chuẩn.

1.5. CÁC KHÔNG GIAN TÊN CÓ SẴN (2)

- **System.Collections:** Chứa các class và interface định nghĩa cấu trúc dữ liệu phức tạp như danh sách, hàng đợi, bit array, hash table, và dictionaries
- **System.Data:** Chứa các class theo kiến trúc ADO.NET. Kiến trúc này cho phép xây dựng các thành phần thường dùng như insert, modify và delete dữ liệu từ nhiều nguồn dữ liệu
- **System.Diagnostics**: Chứa các class được sử dụng để tương tác với các quy trình hệ thống. Không gian tên này cũng cung cấp các class sử dụng để gỡ lỗi ứng dụng và theo dõi quá trình thực thi mã.
- **System.IO:** Chứa các class cho phép đọc và ghi vào luồng dữ liệu
- **System.Net:** Chứa các class cho phép các Web Application
- **System.Web:** Chứa các class và interface cho phép giao tiếp giữa trình duyệt và máy chủ

1.5. CÁC KHÔNG GIAN TÊN CÓ SẴN (3)

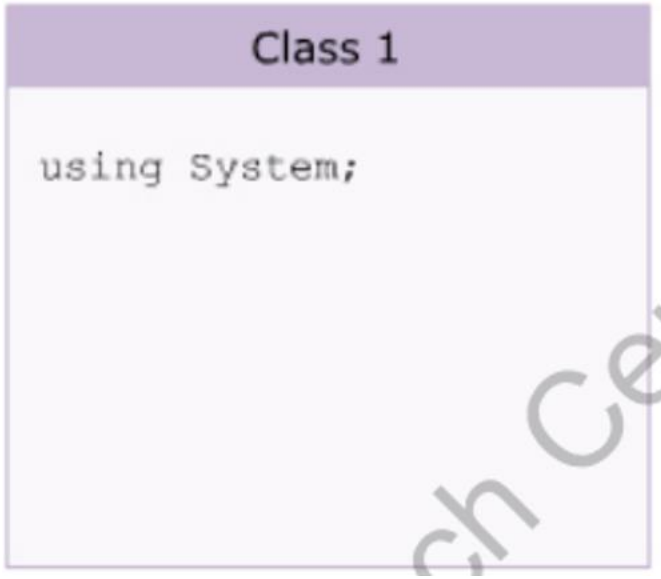


1.6. USING SYSTEM NAMESPACE

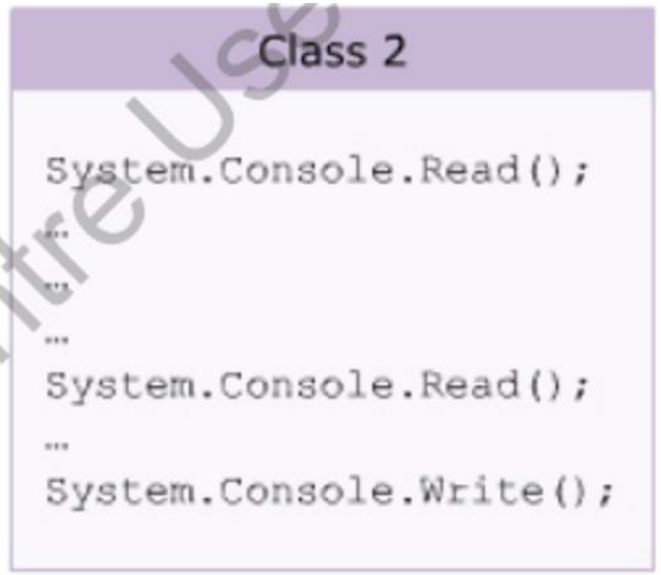
- System namespace được nhập mặc định trong .NET Framework
- Hiển thị ở dòng đầu tiên của chương trình với từ khóa using
- Để tham chiếu đến các lớp trong không gian tên dựng sẵn, cần tham chiếu rõ ràng đến các lớp bắt buộc
- Nó được thực hiện bằng cách chỉ định không gian tên và tên lớp được phân tách bằng dấu chấm (.) sau từ khóa using khi bắt đầu chương trình
- Có thể tham chiếu đến các lớp trong các không gian tên theo cách tương tự mà không cần từ khóa using
- Tuy nhiên, cách này là dư thừa bởi cần tham chiếu đến toàn bộ phần khai báo mỗi khi tham chiếu đến lớp trong mã.

1.6. USING SYSTEM NAMESPACE (2)

- Có 2 cách sử dụng



Hiệu quả



Không hiệu quả

2.1. NAMESPACE LỒNG NHAU

Các namespace cũng có thể khai báo lồng nhau, nhiều cấp sau đó dùng ký hiệu . để truy cập đến namepace mong muốn, ví dụ namepace B nằm trong namspace A

```
namespace A {
// Định nghĩa các lớp, cấu trúc ...
namespace B { // Định nghĩa các lớp, cấu trúc ... }
}
```

Khi đó muốn dùng các lớp, cấu trúc ... có trong namespace B thì khai báo

```
using A.B;
```

Thư viện .NET cung cấp rất nhiều namespace - mỗi namespace là tập hợp các lớp, cấu trúc về một vấn đề nào đó. Tạo ra namespace nhiều cấp còn có thể khai báo một cách riêng rẽ nhưng phải chỉ rõ tên đầy đủ của namespace (tên namespace gốc)

2.1. NAMESPACE LỒNG NHAU (2)

Ví dụ, namespace C là con của namespace B, B lại là con của namespace A thì bạn có thể khai báo lồng nhau như trên hoặc:

```
namespace A
{
    public struct StructInA {};
}
namespace A.B
{
    public struct StructInB {};
}
namespace A.B.C
{
    public struct StructInC {};
}
```

Cách khai báo này hoàn toàn giống:

```
namespace A
{
    public struct StructInA { };

    namespace B
    {
        public struct StructInB { };

        namespace C
        {
            public struct StructInC { };
        }
    }
}
```

2.2. TRIỂN KHAI CÁC NAMESPACE LỒNG NHAU

- C# cho phép tạo ra hệ thống phân cấp không gian tên tạo các không gian tên trong các không gian tên
- Việc lồng các không gian tên như vậy được thực hiện bằng cách đặt khai báo không gian tên trong khai báo của không gian khác
- Có thể sử dụng cú pháp sau để tạo các không gian tên lồng nhau:

```
namespace <NamespaceName>
{
   namespace <NamespaceName>
   {
   }
   namespace <NamespaceName>
   {
   }
}
```

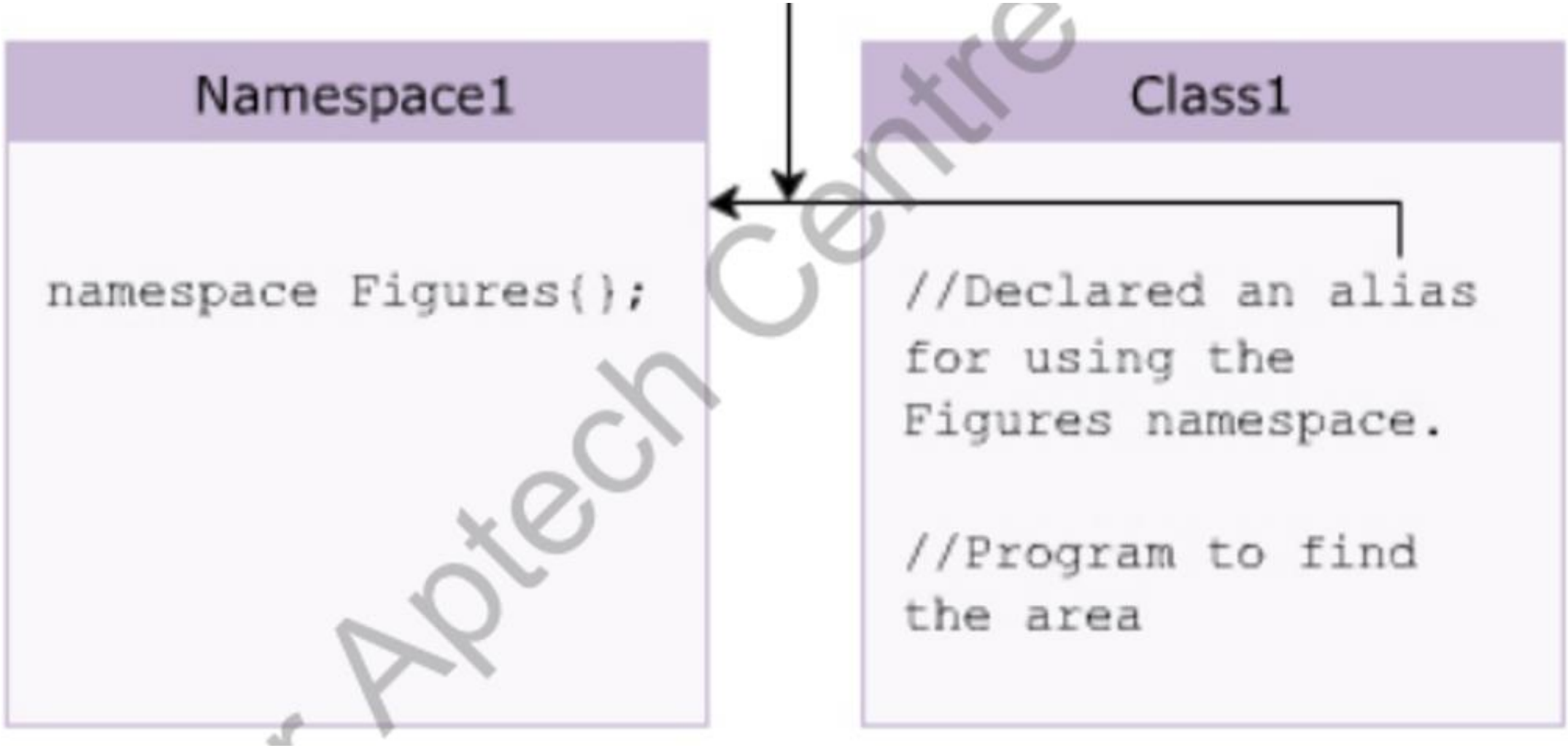
```
namespace Contact
{
    public class Employees
    {
       public int EmpID;
    }
    namespace Salary
    {
       public class SalaryDetails
       {
          public double EmpSalary;
       }
    }
}
```

```
using System;
class EmployeeDetails
{
     static void Main(string [] args)
     {
          Contact.Salary.SalaryDetails objSal = new
          Contact.Salary.SalaryDetails();
          objSal.EmpSalary = 1000.50;
          Console.WriteLine("Salary: " + objSal.EmpSalary);
     }
}
```

2.3. NAMESPACE BÍ DANH

- Bí danh:
  - Là tên thay thế tạm thời tham chiếu đến cùng một thực thể
  - Cũng hữu ích khi chương trình chứa nhiều khai báo lồng namespace và muốn phân biệt lớp nào thuộc vùng namespace nào.
  - Giúp mã dễ đọc hơn đối với lập trình viên và dễ dàng bảo trì hơn.
- Namespace được tham chiếu với từ khóa using tham chiếu đến tất cả lớp bên trong namespace.
- Tuy nhiên, thỉnh thoảng có thể truy cập class từ một namespace cụ thể.
- Có thể sử dụng tên bí danh tham chiếu đến lớp bắt buộc và ngăn việc sử dụng tên đủ điều kiện

2.3. NAMESPACE BÍ DANH (2)



2.3. NAMESPACE BÍ DANH (2)

Cú pháp:

```
using<aliasName> = <NamespaceName>;
```

```
namespace Bank.Accounts.EmployeeDetails
{
     public class Employees
     {
        public string EmpName;
     }
}
```

```
using IO = System.Console;
using Emp = Bank.Accounts.EmployeeDetails;
class AliasExample
{
    static void Main (string[] args)
    {
         Emp.Employees objEmp = new Emp.Employees();
         objEmp.EmpName = "Peter";
         IO.WriteLine("Employee Name: " + objEmp.EmpName);
    }
}
```

2.4. BÍ DANH BÊN NGOÀI

Ví dụ:

- Tổ chức lớn trên nhiều project
- Lập trình viên làm việc trên 2 project khác nhau có thể sử dụng tên lớp giống nhau bên trong namespace giống nhau và lưu trữ trong assembly khác nhau.
- Assembly được tạo ra có thể chứa các phương thức tương tự
- Khi assembly được sử dụng trong chương trình đơn lẻ và phương thức chung từ các assembly được gọi, lỗi biên dịch được sinh ra.
- Lỗi biên dịch xảy ra do cùng một phương thức nằm trong cả hai assembly
- Đây được gọi là tham chiếu mơ hồ và nó có thể được giải quyết bằng cách triển khai bí danh bên ngoài.

2.4. BÍ DANH BÊN NGOÀI (2)

```
extern alias LibraryOne;
extern alias LibraryTwo;
using System;
class Companies
{
   static void Main (string [] args)
   {
      LibraryOne::Employees.Display();
      LibraryTwo::Employees.Display();
   }
}
```

```
/reference: LibraryOne =One.dll
/reference: LibraryTwo =two.dll
```