LẬP TRÌNH WINDOWS

BÀI 1: TỔNG QUAN VỀ .NET FRAMEWORK VÀ NGÔN NGỮ C#

Giảng viên: Lý Anh Tuấn

Email: tuanla@wru.vn

Tổng quan về .Net Framework

- Được phát triển bởi Microsoft
- Là một nền tảng lập trình và thực thi ứng dụng chủ yếu trên hệ điều hành Microsoft Windows
- Bao gồm môi trường Common Language Runtime (CLR) và tập các thư viện hỗ trợ lập trình .Net Framework Class Library

Common Language Runtime (CLR)

CLR là một máy ảo, cung cấp môi trường để chạy các chương trình và hỗ trợ các dịch vụ:

- An ninh phần mềm (*security*)
- Quản lý bộ nhớ (*memory management*)
- Xử lý ngoại lệ (*exception handling*)

.Net Framework Class Library (FCL)

FCL là những thư viện hỗ trợ việc xây dựng các chương trình phần mềm như:

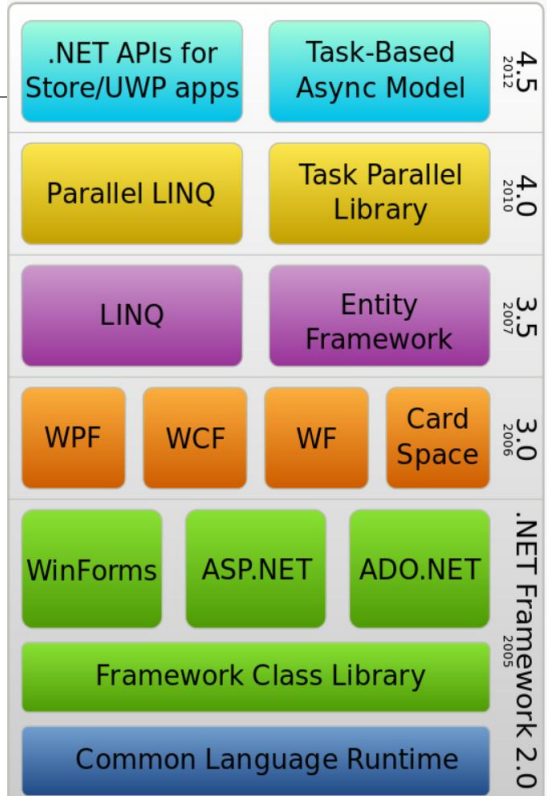
- Lập trình giao diện
- Truy cập, kết nối cơ sở dữ liệu
- Ứng dụng web
- Các giải thuật, cấu trúc dữ liệu
- Giao tiếp mạng
- …

Một số thư viện nền tảng

Namespace	Description
System	Chứa các lớp cơ bản
System.IO	Chứa các lớp cho thao tác Input và Output
System.Net	Chứa các lớp liên quan đến network protocol
System.Collections	Chứa các lớp liên quan đến xử lý tập hợp
System.Data	Chứa các lớp của ADO.NET
System.Drawing	Chứa các lớp thực thi chức năng GUI
System.Threading	Chứa các lớp lập trình MultiThread
System.Web	Chứa các lớp liên quan đến HTTP protocol
System.Xml	Chứa các lớp liên quan XML

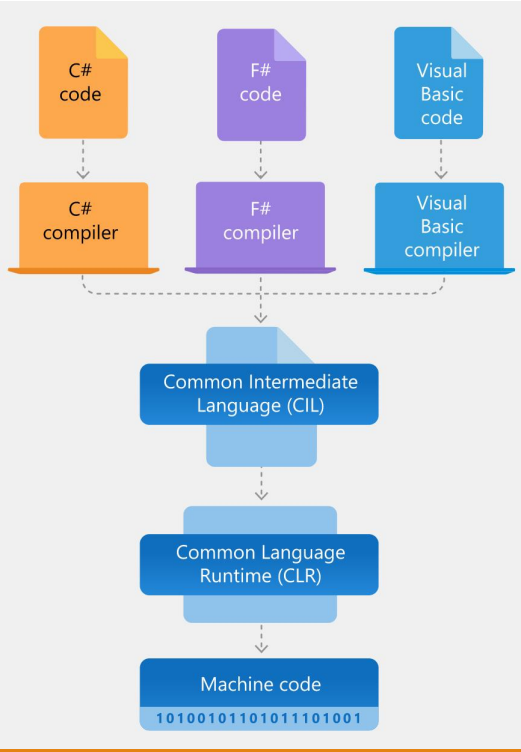
.Net Framework

- Qua nhiều giai đoạn phát triển, đến nay .Net Framework đã tích hợp rất nhiều thành phần thiết kế sẵn giúp cho việc lập trình nhanh và đơn giản hơn
- Hỗ trợ đa ngôn ngữ: C#, VB.Net, C++.Net, Jscrip.Net, F#



Biên dịch và chạy chương trình

- Mã nguồn được dịch theo trình biên dịch tương ứng với ngôn ngữ lập trình.
- Chuyển thành ngôn ngữ trung gian CIL, lưu thành file thực thi dưới dạng .exe hoặc .dll
- Khi chạy file thực thi, CLR sẽ gọi trình biên dịch JIT (just-in-time) chuyển thành mã máy để thực thi chương trình.



.Net Assembly

- Là các thành phần .Net khép kín
  - Được lưu trong các file .dll và .exe
- Bao chứa danh sách các lớp, các kiểu và các tài nguyên
- Là đơn vị triển khai bé nhất trong CLR
- Có một mã số phiên bản duy nhất

Giới thiệu về ngôn ngữ C#

- C# là ngôn ngữ lập trình đơn giản:
  - C# khá giống C / C++ về diện mạo, cú pháp, biểu thức, toán tử.
  - Các chức năng của C# được lấy trực tiếp từ ngôn ngữ C / C++ nhưng được cải tiến để làm cho ngôn ngữ đơn giản hơn.

Giới thiệu về ngôn ngữ C#

- C# là ngôn ngữ hiện đại, có những tính năng:
  - Xử lý ngoại lệ
  - Thu gom bộ nhớ tự động
  - Có những kiểu dữ liệu mở rộng
  - Bảo mật mã nguồn

Giới thiệu về ngôn ngữ C#

- C# là ngôn ngữ hướng đối tượng với những đặc tính:
  - Sự đóng gói (encapsulation)
  - Sự kế thừa (inheritance)
  - Tính đa hình (polymorphism)

Giới thiệu về ngôn ngữ C#

- C# là ngôn ngữ mạnh mẽ và mềm dẻo khi được dùng cho những dự án:
  - Xử lý văn bản
  - Xử lý đồ họa
  - Xử lý bảng tính
  - Thậm chí để tạo trình biên dịch cho các ngôn ngữ khác

Giới thiệu về ngôn ngữ C#

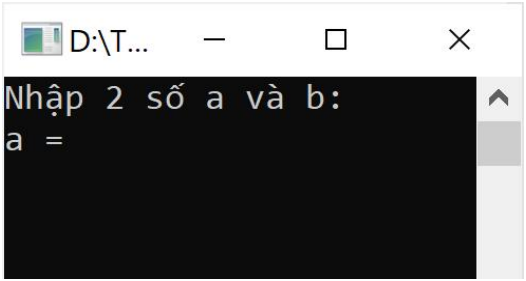
- C# là một trong những ngôn ngữ được phát triển ở nền tảng .Net
  - Để sử dụng được C# cần cài đặt .Net Framework
- C# là một ngôn ngữ lập trình trực quan
  - Để sử dụng được các tính năng lập trình trực quan cần cài đặt Microsoft Visual Studio

Một số dạng ứng dụng của C#

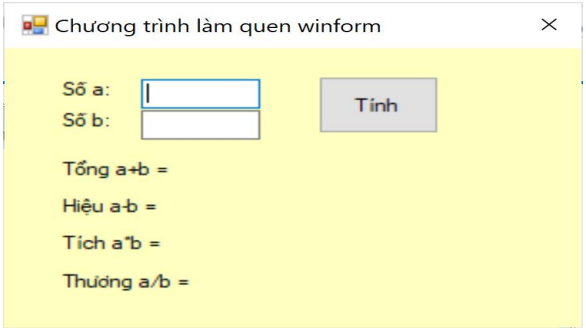
- Chương trình Console
  - Giao tiếp với người dùng bằng bàn phím
  - Chỉ sử dụng cửa sổ dòng lệnh, không có giao diện đồ họa
- Chương trình winform
  - Giao tiếp với người dùng bằng bàn phím và chuột
  - Có giao diện đồ họa và xử lý sự kiện
- Chương trình webform
  - Kết hợp với ASP.NET, C# đóng vai trò xử lý ngầm
  - Có giao diện đồ họa và xử lý sự kiện

Giao diện các loại ứng dụng trong C#

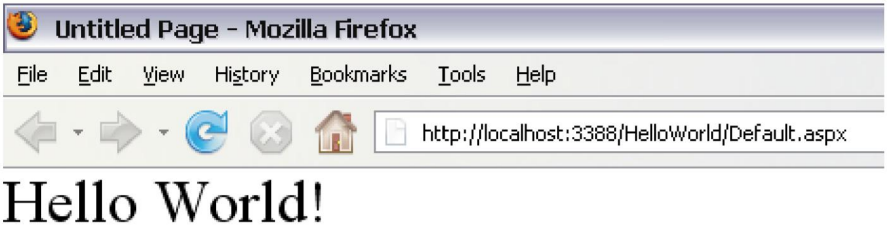
- Chương trình Console:



- Chương trình winform:



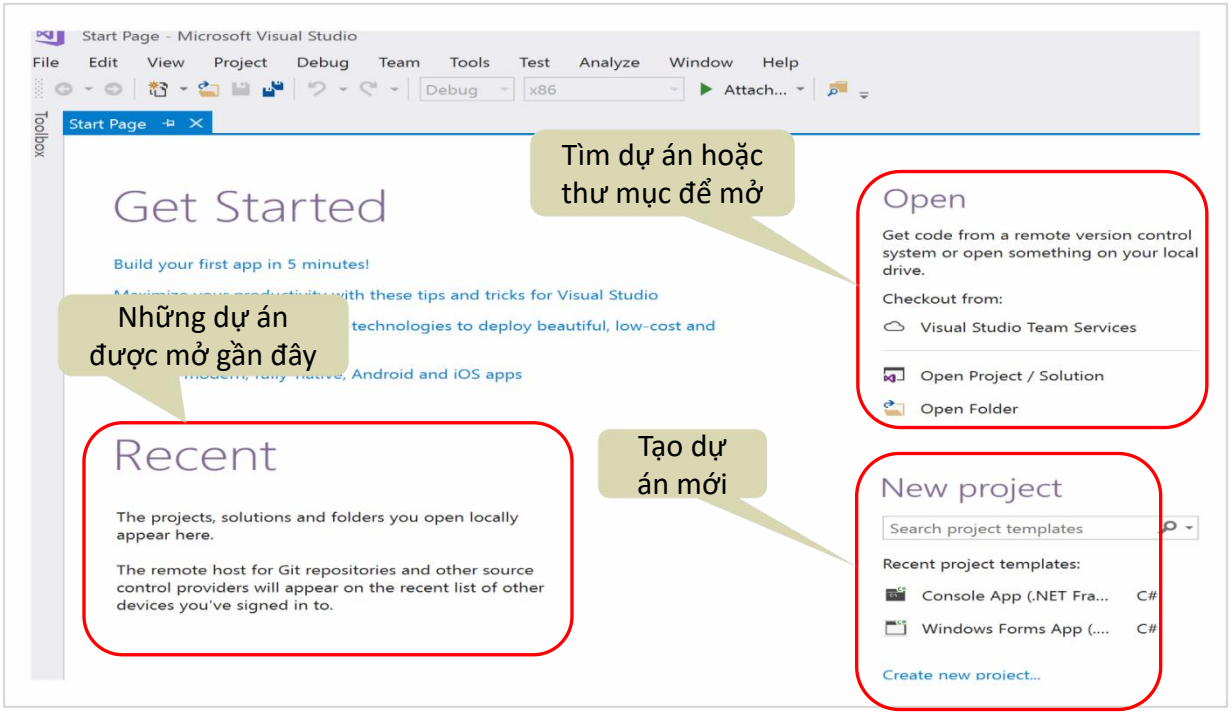
- Chương trình webform:



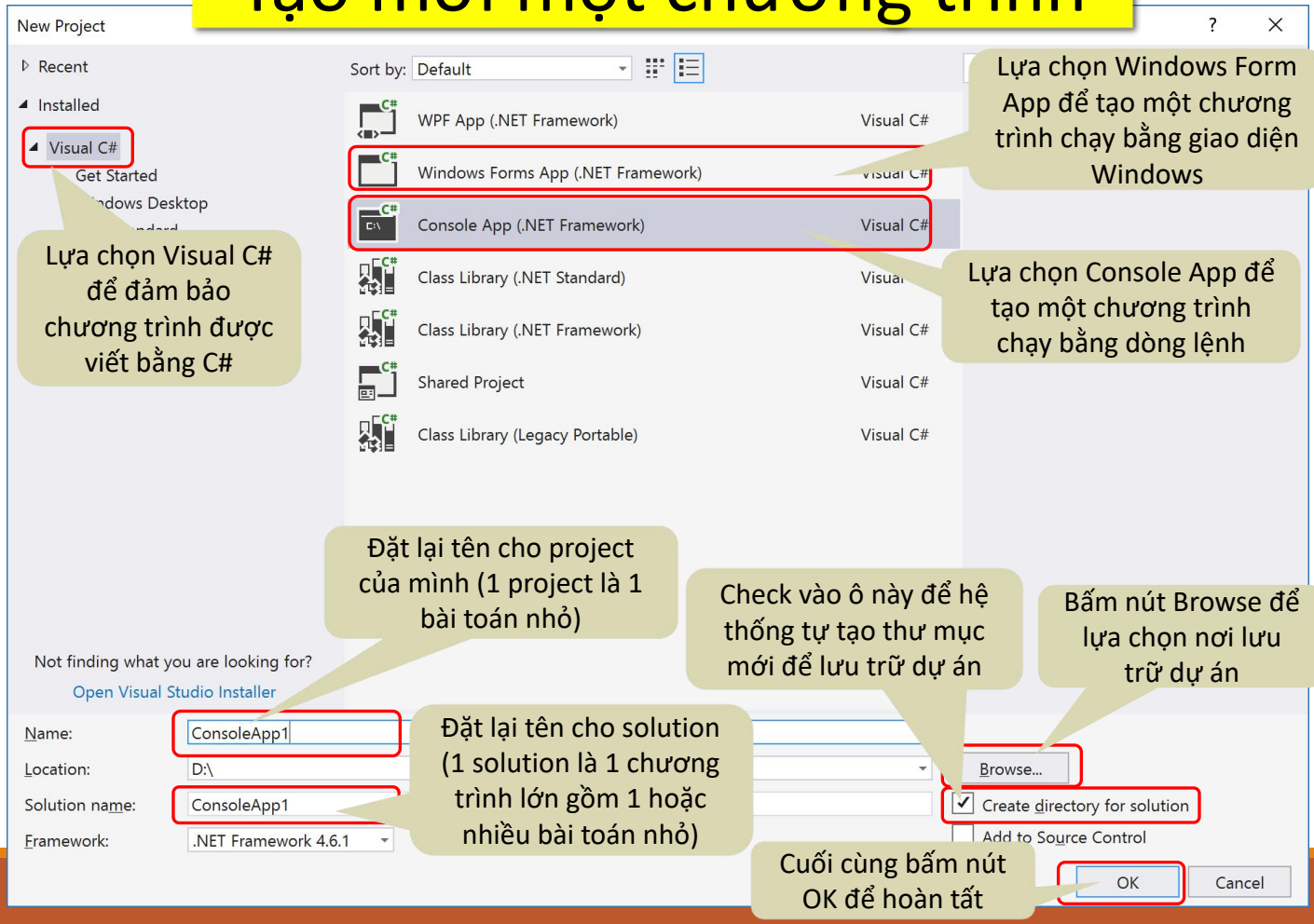
Cách tạo một chương trình Console trong C#

Khởi động phần mềm Visual Studio

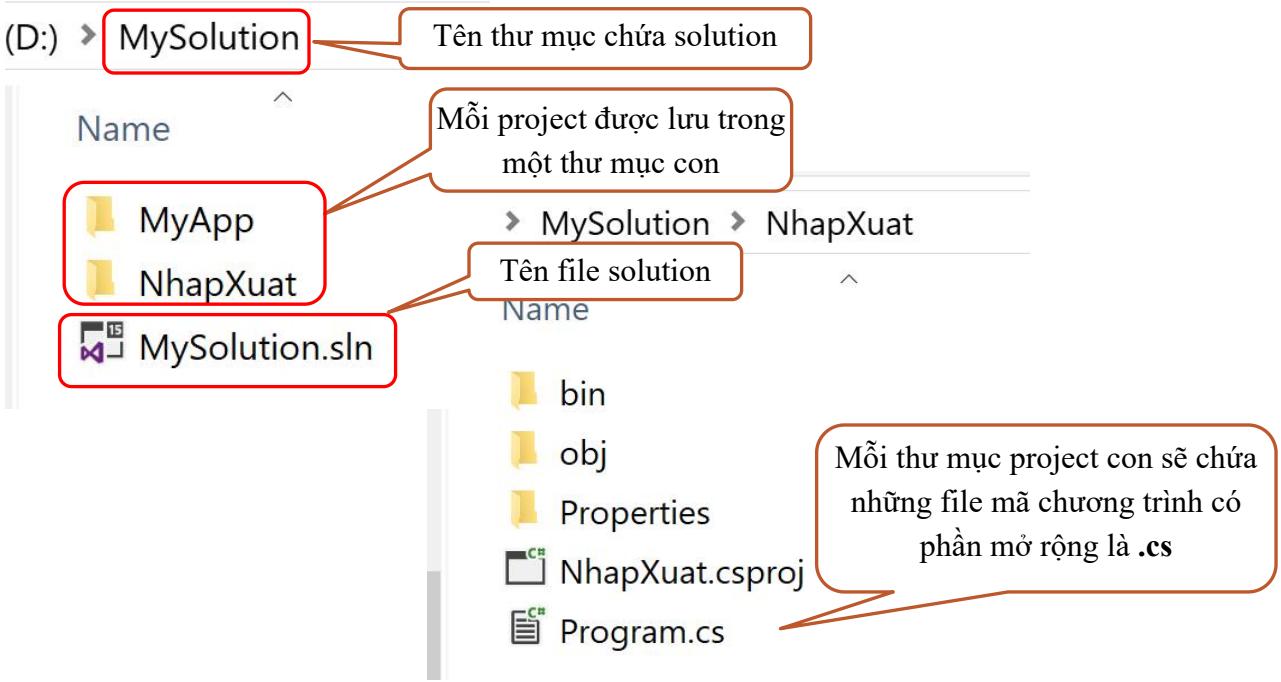
Khởi động phần mềm Visual Studio



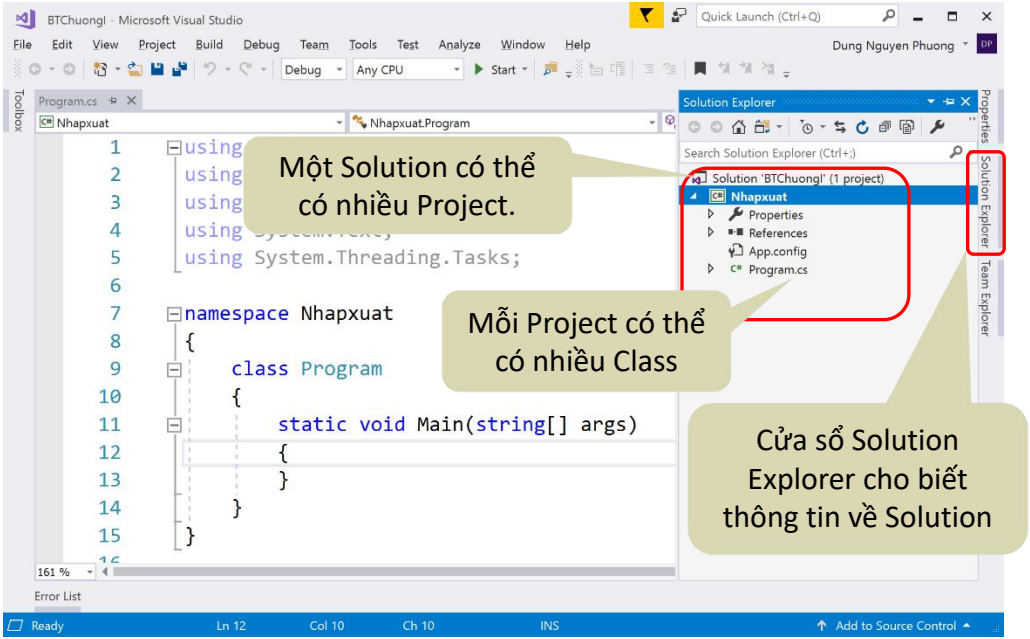
Tạo mới một chương trình



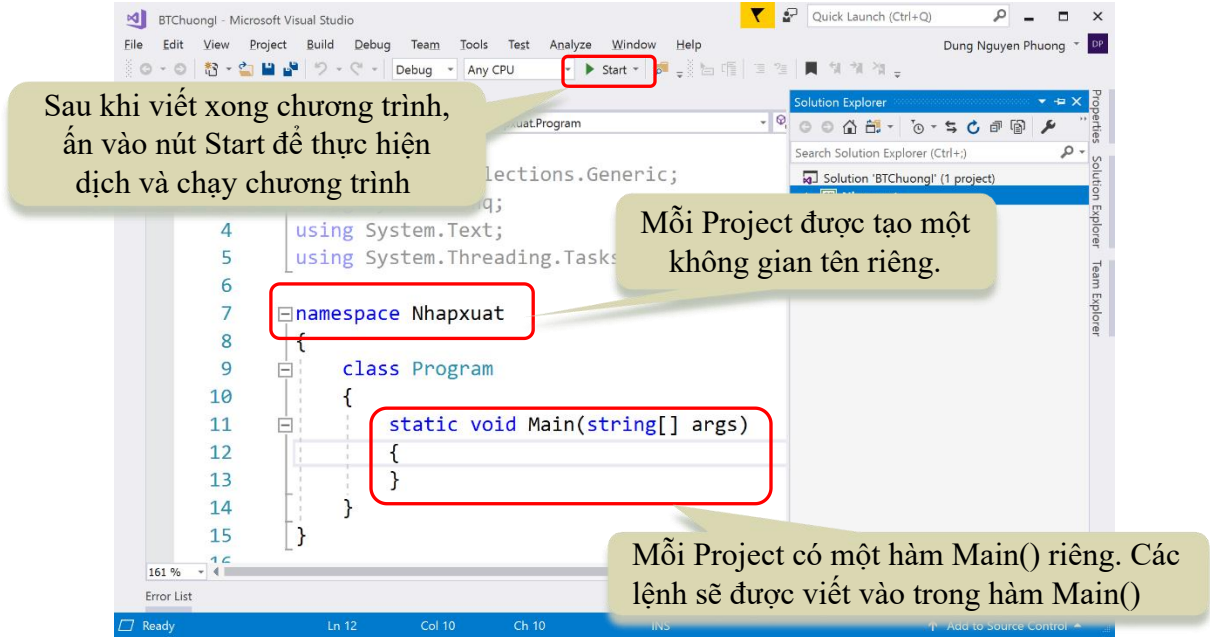
Cấu trúc thư mục solution



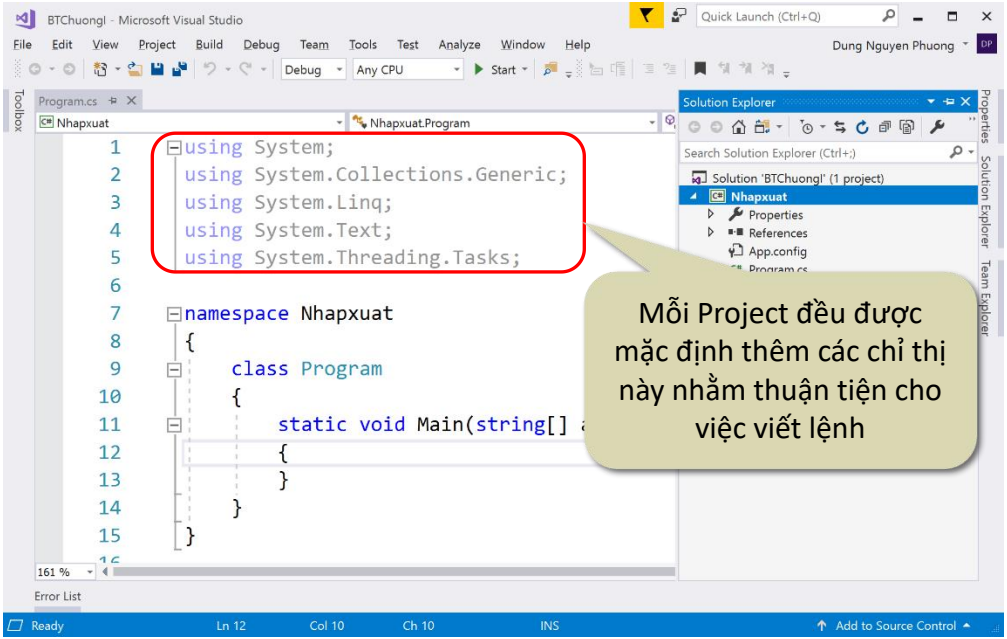
Làm việc với một chương trình Console trong C#



Làm việc với một chương trình Console trong C#



Làm việc với một chương trình Console trong C#



Lệnh nhập/xuất trong C#

- Việc nhập, xuất dữ liệu ra màn hình Console trong C# sử dụng lệnh ReadLine(), WriteLine()
- Hai lệnh này thuộc lớp Console trong namespace System
- Do đó ở đầu chương trình sử dụng chỉ thị using System thì trong chương trình chỉ cần viết Console.WriteLine() mà không cần viết System.Console.WriteLine();

Lệnh nhập/xuất trong C#

VD:

```
namespace Nhapxuat
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, what your name?");
            string name = Console.ReadLine();
            Console.WriteLine("Hello " + name);
            Console.ReadLine();
        }
    }
}
```

Chuỗi in ra được cộng dồn

In ra giá trị của biến

- Cú pháp chung:
  - **Console.Write("{0} {1} {2} {...}", <giá trị 0>, <giá trị 1>, <giá trị 2>, ...);**
  - trong đó: <giá trị 0> sẽ được điền tương ứng vào vị trí số 0, tương tự như vậy cho các giá trị còn lại
- Ví dụ:

```
int a = 5, b = 10;
Console.Write("a = {0}, b = {1}", a, b); // In ra: a = 5, b = 10
```

Nhập dữ liệu vào một biến

Chú ý:

- Lệnh ReadLine() dùng để nhập một dòng dữ liệu
- Muốn nhập dữ liệu kiểu số **cần chuyển đổi kiểu** dữ liệu cho dòng dữ liệu nhập vào
- Lệnh chuyển kiểu: là các lệnh thuộc lớp Convert trong namespace System
- VD: int a;

```
a = Convert.ToInt32(Console.Readline());
```

Công cụ Debug của VS

- Đặt breakpoint trong C#:

```
namespace ChaoMung
{
    class Program
    {
        static void Main(string[] args)
        {
            int a;
            int b;
            a = b = 10;
            int c = 90;
            String s = "Chao Mung";
            Console.WriteLine(" a + b + c = {0}", a + b + c);
            Console.WriteLine(s);
        }
    }
}
```

Công cụ Debug của VS

- Sử dụng các nút lệnh sau trên thanh công cụ Debug để tiến hành chạy từng bước:

– Step into (F11): Công cụ Debug sẽ thiết lập con chạy thực thi ở lệnh đầu tiên của phương thức đầu tiên được gọi từ dòng hiện thời

– Step out (Shift + F11): Công cụ Debug sẽ thực thi các dòng lệnh còn lại của phương thức và thiết lập con chạy thực thi ở dòng lệnh ngay sau lời gọi phương thức

– Step over (F10): Bỏ qua việc thực thi chi tiết một phương thức cụ thể

Các kiến thức đã học trong bài

- Tổng quan về .Net Framework
- Tổng quan về ngôn ngữ C#
- Cách tạo chương trình Console trong C#
- Công cụ Debug của Visual Studio