<https://odetocode.com>

<https://weblogs.asp.net/scottgu>

<http://www.hanselman.com/blog/>

C# vs .NET

C# is a programming language

.NET is a framework for building applications on Windows

+ .Net framework consists of two components is called CLR ( Common Language Runtime ) and Class Library

Namespace is container for related classes

Assembly is a container for related Namespace

Application is container for more Assembly

DLL: Dynamic Link Library

**static**

static is instance mothod of class, this will be call by class

**Array**

Array is a data structure to store a collection of variables of the same type

**String**

String is a sequence of characters.

**Enum**

Enum is a set of name/value pairs (constants)

with **struct** the memory value save on stack and no change, primitive is a struct, struct is value type (tham tri)

with **class** the memory value save on heap, on stack only save local, class is reference type (tham chieu)

// <https://daynhauhoc.com/t/su-khac-nhau-cua-struct-va-class-trong-c/19226/2>

Trong C#, những cái gì là **reference type** thì variable đều trỏ lên **heap**.  
Còn **value type** thì variable **thường** nằm trên **stack**.

Khi bạn sử dụng **value type** làm **tham số** cho hàm, khi hàm đó chạy là **cả 1 cục dữ liệu value type** sẽ được bỏ vào hàm.  
Còn nếu là **reference type** làm **tham số** thì chỉ có cái **reference** (có thể coi là 1 dạng **pointer cao cấp**) được bỏ vào hàm thôi, rồi từ **reference** đó truy cập đến dữ liệu thật.

Theo mình biết thì thường khi gọi hàm, các **tham số** sẽ được **push vào stack**, dùng kiểu **reference** thì **tiết kiệm** được **stack**, nhưng tốc độ chạy thì chưa chắc vì làm thế này là dùng **heap** vốn không nhanh hơn **stack**, thường thì những dữ liệu loại **nhỏ** mà được **sử dụng liên tục** thì có thể dùng **stack**, [quy tắc của Microsoft để lựa chọn class và struct47](https://msdn.microsoft.com/en-us/library/ms229017.aspx).

À mà bởi vì **reference type** nó kiểu như **con trỏ**, nên có thể gán **null**, **value type** **không** như vậy được.

//

Random rnd = new Random();

int month = rnd.Next(1, 13); // creates a number between 1 and 12

int dice = rnd.Next(1, 7); // creates a number between 1 and 6

int card = rnd.Next(52); // creates a number between 0 and 51

* **Using System.Convert class:** This class provides useful methods to convert any built-in data type to another built-in data type.

* **Using ToString() method:** This method belongs to the Object class and converts any data type value into string.

* Boxing is a process for converting a value type, like integers, to its reference type, like objects that is useful to reduce the overhead on the system during execution because all value types are implicitly of object type.

* Unboxing refers to converting a reference type to a value type.

Array method

<https://msdn.microsoft.com/en-us/library/system.array(v=vs.110).aspx>

**Array vs Lists**

Array: fixed size

List: dynamic size

var numbers = new List<int>( );

var numbers = new List<int>( ) {1, 2, 3, 4} ;

Some method of List:

Add( ) ; AddRange( ); Remove( ); RemoveAt( ); IndexOf( ); Contains( ); Count

<https://msdn.microsoft.com/en-us/library/6sh2ey19(v=vs.110).aspx>

|  |  |
| --- | --- |
| Edit.LineDelete | Ctrl+Shift+L |
| Edit.UncommentSelection | Ctrl+K, Ctrl+U |
| Edit.CommentSelection | Ctrl+K, Ctrl+C |
| Edit.FormatDocument | Ctrl+K, Ctrl+D |

**Method Syntax In LINQ**

<https://medium.com/omarelgabrys-blog/method-syntax-in-linq-60bbefcf4b42>

<https://msdn.microsoft.com/en-us/library/system.linq.enumerable_methods(v=vs.100).aspx>

<https://toidicodedao.com/2015/03/26/series-c-hay-ho-linq/>

LINQ:

Stands for: Language Integrated Query

Gives you the capability to query objects

You can query:

* + Objects in memory, eg collections (LINQ to Objects)
  + Databases (LINQ to Entities)
  + XML (LINQ to XML)
  + ADO>NET data Sets (LINQ to Data Sets)

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/query-syntax-and-method-syntax-in-linq>

<https://msdn.microsoft.com/en-us/library/gg509017.aspx?cs-save-lang=1&cs-lang=csharp#code-snippet-1>

**DateTime Format**

<https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-date-and-time-format-strings>

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

**how to get license key for resharper 2018**

<http://jetbrains.license.laucyun.com/>

key for visual studio

<https://www.quykhanhit.com/2017/11/phan-mem-visual-studio-2017-full-key-huong-dan-cai-dat.html>

use ReSharpes to cover all the code of the method use: crl + w

fn + return = insert.

Generate = Alt + Insert

**FileInfo:** provides instance methods

**File:** provides static methods

**They also have method**

Create( ), Copy( ), Delete( ), Exists( ), GetAtributess( ), Move ( ),ReadAllText( ), WriteAllText( ), ReadAllLines( ), OpenRead( )

**Directory:** provides static methods

**DirectoryInfo:** provides instance methods

**They also have method**

CreateDirectory( ), Delete( ), Exists( ), GetCurrentDirectory( ), GetFiles( ), Move( ), GetLogicalDrives( ),

**Path Class**

GetDirectoryName( ), GetFileName( ), GetExtention( ), GetTempPath( )

**Delegates**

<https://toidicodedao.com/2015/02/10/series-c-hay-ho-callback-trong-c-delegate-action-predicate-func/>

<http://code5s.com/windows/visual-c-sharp/delegate-va-event-trong-csharp.html>

<http://csharpindepth.com/Articles/Chapter2/Events.aspx>

Delegates is an object that know to call a method (or agroup of methods)

Delegates is a reference to a object

**Why do we need delegates?**

For designing extensible and flexible applications (like frameworks)

* **Action**: Action<T in1, T in2, …>. Action tương đương 1 delegate với kiểu trả về là void, với in1, in2 là các params nhận vào.
* **Predicate**: Predicate<T in>. Predicate tương đương 1 delegate với kiểu trả về là bool, với in là các param nhận vào. Predicate chỉ có thể nhận vào **1 param duy nhất**.
* **Func**: Func<T in1, T in2, … , T result>. Function tương đương 1 delegate với kiểu trả về do ta khai báo (result), in1, in2 là các params nhận vào. Func bắt buộc phải trả ra giá trị, không thể trả void.

**Events**

Events is: - A mechanism for communication between objects

- Use in building Loosely Coupled Applications

- Helps extending applications

3 step to create event

1. Define a delegate
2. Define an event based on that delegate
3. Raise the event

**What are Extension Method?**

Allow us to add methods to an existing class without:

* + Changing its source code, or
  + Creating a new class that inherits from it

**BinaryFormatter**

<https://www.mastercode.vn/blog/web-development/bai-9-serialization-nen-tang-lap-trinh-c.57>

**File Class**

<https://msdn.microsoft.com/en-us/library/system.io.file(v=vs.110).aspx>

[hoangdinhdng@gmail.com](mailto:hoangdinhdng@gmail.com)

[hoangdnm@softech.vn](mailto:hoangdnm@softech.vn)

**What is a Lambda Expression?**

An anonymous method :

* No access modifier
* No name
* No return statement

**Why do we use them?**

For convenience

**Socket – TcpListener**

<https://www.c-sharpcorner.com/UploadFile/201fc1/creating-a-serversharp47client-application-using-only-tcp-prot/>

<https://www.c-sharpcorner.com/UploadFile/ajyadav123/applied-C-Sharp-net-socket-programming/>

**Dynamic**

Programming Languages

* + Statically-typed language: C#, Java
  + Dynamically-typed language: Ruby, Javascript, Python

Type resolution

Static languages: at compile-time

Dynamic languages: at run-time

Benefits

Static languages: early feedback (compile-time)

Dynamic languages: easier and faster to code

C# history

* + Started as a static language
  + .NET 4 added the dynamic capability, to improve interoperability with
    - COM (eg writing office applications)
    - Dynamic languages (IronPython)

**Asynchronous and Synchronous**

Synchronous Program Execution

Program is executed line by line, one at a time.

When a function is called, program execution has to wait until the function returns

Asynchronous Program Execution

When a function is called, program execution continues to the next line, without waiting for the function to complete

When to use asynchronous?

Accessing the web

Working with files and databases

Working with images

How?

Tradition approaches:

Multi-threading

Callbacks

New approach since NET 4.5

Async / Await