C#

Basics

[Variable](https://dotnettutorials.net/lesson/variables-in-csharp/)[[1]](#endnote-1)

[Data Type](https://dotnettutorials.net/lesson/data-types-in-csharp/)[[2]](#endnote-2)

Value Type[[3]](#endnote-3)(STACK[[4]](#endnote-4))

References Type[[5]](#endnote-5)(HEAP[[6]](#endnote-6))

[Casting & Type conversions](https://dotnettutorials.net/lesson/type-casting-in-csharp/)

[Operators](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/operators/)[[7]](#endnote-7)

[Literals](https://dotnettutorials.net/lesson/literals-in-csharp/)[[8]](#endnote-8)

[Statements](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/statements)[[9]](#endnote-9)

Keywords

[Constant](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/const) vs. [read-only](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/readonly)[[10]](#endnote-10)

[Static](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/static)[[11]](#endnote-11)

[Checked](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/statements/checked-and-unchecked)[[12]](#endnote-12)

[Sealed](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/sealed)[[13]](#endnote-13)

[String](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/strings/)[[14]](#endnote-14)

[Method](https://learn.microsoft.com/en-us/dotnet/api/system.string.concat?view=net-9.0)

String Concatenation

[StringBuilder](https://learn.microsoft.com/en-us/dotnet/api/system.text.stringbuilder?view=net-9.0)

[String Interpolation](https://learn.microsoft.com/en-us/dotnet/csharp/tutorials/string-interpolation) ($””)

[Boxing and Unboxing](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/types/boxing-and-unboxing)[[15]](#endnote-15)

OOP

[Class, Struct, Records](https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/object-oriented/)[[16]](#endnote-16)

Fields

Properties

Contructors

Methods

Object[[17]](#endnote-17)

Access Modifiers[[18]](#endnote-18)

Principal

Inheritance[[19]](#endnote-19)

Virtual[[20]](#endnote-20)

Override[[21]](#endnote-21)

Abstract[[22]](#endnote-22)

Polymorphism[[23]](#endnote-23)

Overloading[[24]](#endnote-24)

Overriding[[25]](#endnote-25)

Abtraction[[26]](#endnote-26)

Abstract class[[27]](#endnote-27)

Interface[[28]](#endnote-28)

Encapsulation[[29]](#endnote-29)

Collections

Iterators[[30]](#endnote-30)

[Array](https://dotnettutorials.net/lesson/arrays-csharp/)

Methods

[2D Arrays](https://dotnettutorials.net/lesson/two-dimensional-array-in-csharp/)

List[[31]](#endnote-31)

Dictionary

Stack[[32]](#endnote-32)

Queue[[33]](#endnote-33)

[Collection generic](https://www.tutorialsteacher.com/csharp/csharp-collection)[[34]](#endnote-34)

[Collection no-genaric](https://www.tutorialsteacher.com/csharp/csharp-collection)[[35]](#endnote-35)

[Exceptions](https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/exceptions/exception-handling)

[Asynchorous](https://learn.microsoft.com/en-us/dotnet/csharp/asynchronous-programming/)

[Threads](https://learn.microsoft.com/en-us/dotnet/standard/threading/using-threads-and-threading)

[Tasks](https://learn.microsoft.com/en-us/dotnet/csharp/asynchronous-programming/task-asynchronous-programming-model)

[Await](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/operators/await)

[Async](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/async)

[Delegates & events](https://learn.microsoft.com/en-us/dotnet/csharp/delegates-overview)[[36]](#endnote-36)

[Generics](https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/generics)[[37]](#endnote-37)

[LinQ](https://learn.microsoft.com/en-us/dotnet/csharp/linq/)

GHI CHÚ

1. [Variables](https://dotnettutorials.net/lesson/variables-in-csharp/): <kiểu dữ liệu><tên biển>

   Giá trị dữ liệu có thể thay đổi

   Tên gọi tên chiếu đến vùng nhớ trong bộ nhớ [↑](#endnote-ref-1)
2. [Data type](https://dotnettutorials.net/lesson/data-types-in-csharp/):

   Là tập hợp các nhóm dữ liệu có cùng đặc tính, cách lưu trữ và thao tác xử lý.

   Data type được chia thành 2 loại chính

   Built-in, được định nghĩa sẵn

   User-defined, do người dùng định nghĩa [↑](#endnote-ref-2)
3. Value type:

   Vùng nhớ của biến lưu giá trị dữ liệu và được lưu trữ trong bộ nhớ STACK

   Predefined Data Types: bool, byte, char, decimal, float, int, …

   User-defined Data Types: Structure(struct), Enumerations (enum), … [↑](#endnote-ref-3)
4. STACK:

   Vùng nhớ được cấp khi biên dịch

   Kích thước cố định

   Implement thread

   Stack overflow (tràn bộ nhớ) [↑](#endnote-ref-4)
5. Reference Type:

   Vùng nhớ của biến chứa địa chỉ của đối tượng dữ liệu và lưu trong bộ nhớ STACK

   Đối tượng dữ liệu được lưu trong bộ nhớ HEAP

   Predefined Reference Type: object, dynamic, string

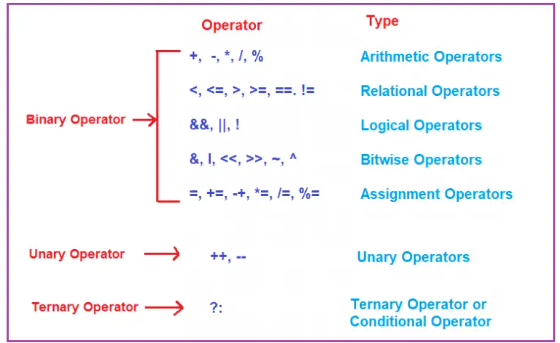
   User-defined Data Types: class, interface [↑](#endnote-ref-5)
6. HEAP:

   Vùng nhớ được cấp khi chạy chương trình

   Kích thước thay đổi

   Cấp phát bộ nhớ động (new)

   Garbage Collection để giải phóng vùng nhớ [↑](#endnote-ref-6)
7. [Operators](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/operators/):

    [↑](#endnote-ref-7)
8. [Literals](https://dotnettutorials.net/lesson/literals-in-csharp/): Giá trị hằng số, gán trực tiếp vào biến

   A diagram of a diagram

   AI-generated content may be incorrect. [↑](#endnote-ref-8)
9. [Statements](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/statements):

   A diagram of a control flow statement

   AI-generated content may be incorrect. [↑](#endnote-ref-9)
10. [Constant](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/const) vs. [read-only](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/readonly):

    Đều là hằng số (không thể thay đổi)

    |  |  |
    | --- | --- |
    | Const | Readonly |
    | Gán giá trị khi khai báo | Gán giá trị khi khai báo hoặc trong constructor |
    | static | Static, instance (gắn với object, tạo instance = new object) |

    [↑](#endnote-ref-10)
11. [Static](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/static):

    Khai báo thành phần (variable, method, class, contructor, …) phạm vi toàn cục

    Thành phần là static thì sẽ thuộc class thay vì instance của class

    Phân loại

    Static fields

    Biến chung của class

    Thuộc class

    Static methods

    Không thể truy cập các fields và method non-static của class

    Thuộc class

    Static contructor

    Gọi 1 lần khi khởi tạo lớp

    Có từ khóa static (static <name\_constructor>)

    Không có access modifier

    Static class

    Không khởi tạo (new)

    Chỉ chứa thành phần là static [↑](#endnote-ref-11)
12. [Checked](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/statements/checked-and-unchecked): Kiểm tra tràn số học (arithmetic overflow checking) vượt qua giới hạn của kiểu dữ liệu [↑](#endnote-ref-12)
13. [Selead](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/sealed): Ngăn không cho kế thừa từ class, method [↑](#endnote-ref-13)
14. [String](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/strings/): là class bất biến (immutable), các thao tác thay đổi chuỗi sẽ tạo ra chuỗi mới. [↑](#endnote-ref-14)
15. [Boxing and Unboxing](https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/types/boxing-and-unboxing):

    |  |  |
    | --- | --- |
    | **Boxing** | **Unboxing** |
    | Convert a value type into type object | Convert an object type into value type |
    | Implicit | Explicit |
    | Value stored on STACK, copy an object stored on HEAP | The object stored on HEAP, copy to value stored on STACK |

    [↑](#endnote-ref-15)
16. Class, Struct, Records:

    **Class**:

    Tập hợp các object có cùng điểm chung (attribute, method)

    Ví dụ: class smartphone

    Attribute: color, size, …

    Method: call, message, …

    → các object có thể là: Iphone, Samsung,..

    **Struct**:

    Value type

    Đóng gói các fields khác nhau nhưng liên quan

    Khi sử dụng phải cấp phát vùng nhớ (new), Không được kế thừa

    Trong sruct còn các method và struct khác.

    |  |  |
    | --- | --- |
    | **Class** | **Struct** |
    | Inheritance | Non-inheriatance |
    | Reference types (HEAP) | Value types (STACK) |
    | Create instance with new keyword | Create instance with/ without new keyword |
    | Null | Not null (unless Nullable<T>) |
    | Destructor ( ~className() ) | Not Destructor |
    | Large program | Small program |

    Class member

    |  |  |
    | --- | --- |
    | **fields** | a variable of any type that is declared directly in a class or struct |
    | **constants** | are fields whose value is set at compile time and cannot be changed |
    | **properties** | provide protection for a class field to keep it from being changed from outside |
    | **methods** | define the actions that a class can perform |
    | **constructors** | are methods that are called when the object is first created |

    [↑](#endnote-ref-16)
17. Object

    Là một reference type

    Là instance của class

    Gồm attribute và method

    Ví dụ: object Iphone

    Attribute: color, size, …

    Method: call, message, …

    |  |  |
    | --- | --- |
    | **Class** | **Object** |
    | Blueprint (mẫu thiết kế) | Instance của class |
    | Giá trị cố định | Giá trị thay đổi |
    | Logical entities | Physical entity |
    | Không cấp phát bộ nhớ đến khi tạo object | Cấp phát bộ nhớ khi khởi tạo |
    | Khởi tạo bẳng class keyword | Khởi tạo bằng new keyword |
    | Tạo một lần | Tạo nhiều lần |

    [↑](#endnote-ref-17)
18. Access Modifiers:

    Giới hạn phạm vi truy cập fields, methods, contructors, classes, …

    5 loại access modifers:

    Public

    Private

    Protected

    Internal

    Protected internal [↑](#endnote-ref-18)
19. Inheritance:

    Kế thừa, lớp con kế thừa các attribute, method của lớp cha

    Ví dụ:

    Lớp cha là vehicles, có attribute: color, size, …

    Lớp con Honda cũng có các attribute: color, size, … [↑](#endnote-ref-19)
20. Virtual: Đánh dấu phương thức có thể ghi đè ở lớp con [↑](#endnote-ref-20)
21. Override: Ghi đè phương thức của lớp cha [↑](#endnote-ref-21)
22. Abstract:

    Abstract class: Lớp chứa phương thức thuần ảo

    Abstract method: Phương thức ảo, không có định nghĩa bên trong [↑](#endnote-ref-22)
23. Polymorphism:

    Đa hình, cho phép cùng một phương thức có nhiều hình thức

    Ví dụ:

    Các smartphone cùng kế thừa attribute của parent class nhưng samsung chạy hệ điều hành android, Iphone chạy hệ điều hành Ios.

    Lớp Animals, các lớp con cùng thực hiện method Speak, nhưng chó sẽ kêu “gâu gâu”, mèo sẽ kêu “meo meo” [↑](#endnote-ref-23)
24. Overloading: Các methods, functions có thể cùng tên nhưng thứ tự, loại, số lượng tham số được truyền khác nhau [↑](#endnote-ref-24)
25. Overriding: Lớp con có thể override các method của lớp cha theo cách khác. [↑](#endnote-ref-25)
26. Abtraction:

    Ẩn những chi tiết không cần thiết

    Tăng tính bảo mật

    Ví dụ: Call API nhưng không biết chi tiết hoạt động như thế nào. [↑](#endnote-ref-26)
27. Abstract class:

    Là một class nhưng không tạo entites (không khởi tạo)

    Có thể chứa hoặc không các abtract method

    Tạo template cho class con kế thừa thực hiện (Task, type, rule)

    Lớp con kế thừa một lần, abtract class có thể kế thừa nhiều interface [↑](#endnote-ref-27)
28. Interface:

    Xác định hành vi của đối tượng (type, events, …)

    Chứa các method không có phần thân

    Không chứa các fields

    Được kế thừa nhiều lần

    Mặc định là public và abstract

    [[link](https://yinyangit.wordpress.com/2012/01/15/oop-interface-vs-abstract-class/)]

    |  |  |
    | --- | --- |
    | **Abstract class** | **Interface** |
    | Cho phép khai báo constructor, field | Không cho phép |
    | Không hỗ trợ đa kế thừa | Hỗ trợ đa kế thừa |
    | Các method sử dụng các access modifiers (private, protected, …) | public |
    | Class con chỉ kế thừa 1 abstract class | Class con kế thừa nhiều interface |
    | Có khai báo, định nghĩa (declaration, defination) | Không có định nghĩa |

    [↑](#endnote-ref-28)
29. Encapsulation:

    Đóng gói dữ liệu, ẩn thông tin

    Tăng tính bảo mật [↑](#endnote-ref-29)
30. [Iterators](https://www.geeksforgeeks.org/iterators-in-c-sharp/): Là phương thức để truy xuất từng phần tử của array hoặc collection [↑](#endnote-ref-30)
31. List:

    Tập hợp nhóm đối tượng

    Method: Contain, Find, Add, … [↑](#endnote-ref-31)
32. [Stack](https://learn.microsoft.com/en-us/dotnet/api/system.collections.stack?view=net-6.0):

    Là mô hình LIFO (last in first out)

    Method: Push, Pop, Peek, … [↑](#endnote-ref-32)
33. [Queue](https://learn.microsoft.com/en-us/dotnet/api/system.collections.queue?view=net-6.0):

    Là mô hình FIFO (first in first out)

    Method: Peek, Clone, Clear, … [↑](#endnote-ref-33)
34. Collection generic:

    Dùng trên nhiều kiểu dữ liệu khác nhau

    List<T>, Dictionary<Tkey,Tvalue>, SortedList<Tkey,Tvalue>, Queue<T>, Stack<T>, HashSet<T> [↑](#endnote-ref-34)
35. Collection non-generic:

    ArrayList, SortedList, Stack, Queue, HashTable, BitArrray [↑](#endnote-ref-35)
36. [Delegates & events](https://learn.microsoft.com/en-us/dotnet/csharp/delegates-overview):

    [Delegate](https://howkteam.vn/course/khoa-hoc-lap-trinh-c-nang-cao/delegate-trong-c-4040):

    Tương tự con trỏ trong C, C++

    Biến kiểu tham chiếu, chứa tham chiếu đến method

    Dùng triển khai method, callback event

    delegate <kiểu trả về> <tên delegate> (<tham số nếu có>);

    [Events](https://howkteam.vn/course/khoa-hoc-lap-trinh-c-nang-cao/event-voi-delegate-trong-c-4041):

    Là delegate

    Mục đích thông báo lên cho đối tượng cha biết để xử lý

    event <Kiểu delegate> <tên event>; [↑](#endnote-ref-36)
37. [Generics](https://dotnettutorials.net/lesson/generics-csharp/):

    Cho phép tạo class, method, interface, delegate hoạt động với nhiều kiểu dữ liệu

    Generic class, generic method [↑](#endnote-ref-37)