2

Introduction to C#

What is C#

- C# (pronounced as 'see sharp') is a general-purpose programming language by Microsoft introduced in 2002, for mainly used for development of Console Apps, Windows GUI Apps and Windows Services.

Features of C#

- Object Oriented Programming (OOP) Language.
- Case Sensitive Language.
- Strongly-Typed Language.
- Compiler-based Language.
- Compiled based on CLI; Executed by CLR.
- Developed by Anders Hejlsberg.

Objects

- Object is a small unit (entity) in the program that represents a real-world person or thing.

  Ex: You, Your laptop
- Any physical thing can be considered as object.
- Object is instance (example) of "class".
- Object stores a set of fields (details about object).

regNo: MH123
carModel: Honda City
carYear: 2020

regNo: TS456
carModel: Duster
carYear: 2021

Classes

- Class is a model of objects.
- Class (a.k.a "type") represents structure of data that you want to store in objects.
- Class isn't collection of objects.

```
class Car
{
    string regNo;
    string carModel;
    int    carYear;
}
```

Methods

- Method is a collection of statements to perform certain operation (process or work), such as performing some calculation, displaying some output, checking some conditions etc.
- Method should be a member (part) of class.
- The code statements are not allowed outside the class; they are allowed inside the method only.

```
class  Car
{
      int  calculateEmi( int  carPrice, int noOfMonths, int interestRate )
      {
         //do calculation here
         return (emi);
      }
}
```

Namespace

- Namespace is a collection of classes.
- The goal of namespace is to group-up the classes that are meant for specific purpose.
- You can access the class of a namespace, by using:

  namespace.class

```
namespace  Garage
{
    class  Car
    {
    }
}
```

Syntax of C# Program

```
class ClassName
{
    static void Main( )
    {
    }
}
```

Starting Point of Program Execution

- Every C# program should have a class with "Main" method. "M" is uppercase.
- Main method should be "static" method. A static method can be executed without creating any object for the class.
- Main method should return "void". "void" is a keyword that specifies that the method doesn't return any value to the caller.

C# Language Tokens

- **Keywords:**
  - abstract, as, base, bool, break, byte, case, catch, char, class, const, continue, decimal, default, delegate, do, double, else, enum, event, false, finally, float, for, foreach, goto, if, in, int, interface, internal, is, long, namespace, new, null, object, out, override, private, protected, public, readonly, ref, return, sbyte, sealed, short, sizeof, static, string, struct, switch, this, throw, true, try, typeof, uint, ulong, ushort, using, virtual, void, while, async, await, from, join, let, orderby, partial, set, get, value, var, where
- **Operators:**
  - +, -, *, /, %, <, >, =, ==, != etc.
- **Literals:**
  - Integer Literals : Numbers without decimal part
  - Floating-point literals : Numbers with decimal part
  - Character literals : ' '
  - String literals : " "
  - Boolean literals : true, false

- **Identifiers:**
    - All types of names (variable names, class names, field names, property names, method names etc.)

C# Versions

C#	.NET Framework	Year of Release	Visual Studio
C# 1.0	1.0	2002	Visual Studio 2002
C# 1.2	1.1	2003	Visual Studio 2003
C# 2.0	2.0 & 3.0	2005	Visual Studio 2005 Visual Studio 2008
C# 3.0	2.0, 3.0 & 3.5	2007	Visual Studio 2008
C# 4.0	4.0	2010	Visual Studio 2010
C# 5.0	4.5	2012	Visual Studio 2012 Visual Studio 2013
C# 6.0	4.6	2015	Visual Studio 2015
C# 7.0	4.7	2017	Visual Studio 2017
C# 7.1	4.7.1	2017	Visual Studio 2017
C# 7.2	4.7.2	2017	Visual Studio 2017
C# 7.3	4.7.3	2018	Visual Studio 2017
C# 8.0	4.8	2019	Visual Studio 2019

C# 1.0 New Features

- Classes & Objects
- Structs
- Interfaces
- Events
- Properties
- Delegates

- Attributes
- Reflection
- Threading

C# 2.0 New Features

- Generics
- Partial Classes / Partial Structures
- Anonymous Methods
- Nullable Value Types
- Iterators
- Getter/Setter separate accessibility
- Static Classes
- Null coalescing operator

C# 3.0 New Features

- Auto-Implemented Properties
- Anonymous Types
- Lambda Expressions
- Query Expressions (LINQ)
- Expression Trees
- Extension Methods
- Implicitly Typed Local Variables / Type Inference
- Partial Methods
- Object Initializer
- Collection Initializer

C# 4.0 New Features

- Dynamically Typed Variables
- Named Arguments
- Optional Arguments
- Covariance and Contravariance

C# 5.0 New Features

- Async & Await
- Task Parallel Library

C# 6.0 New Features

- Static Imports (using static)
- Exception Filters (catch when)
- Auto-Implemented Property Initializers
- Null Propagator
- String Interpolation
- nameof operator

C# 7.0 New Features

- Out Variable Declaration
- Tuples
- Discards
- Pattern Matching

- Local Functions
- Expression Bodied Members

C# 7.1 New Features

- Async Main method
- Default literals
- Inferred Tuple Element Names

C# 7.2 New Features

- 'private protected' access modifier
- 'in' parameter modifier

C# 7.3 New Features

- Ref returns
- == operator on tuples

C# 8.0 New Features

- readonly structs
- Switch Expressions
- Using Declarations
- Static Local Functions

C# Naming Conversions

camelCase

- For all local variables, parameters
- Ex: customerName

PascalCase

- For all class names, structure names, namespace names, field names, method names, property names.
- Ex: CustomerName

IPascalCase

- For all interface names.
- Ex: ICustomerName

_camelCase

- For all private fields.
- Ex: _customerName