C# 10.0 Programming

Start learning how to program applications using the C# programming language and become a professional from beginner/novice user.

About the Course & Importance of C#

This is an introduction & programming course using Visual C#.NET. It is purely aimed at complete beginners and assumes that you do not have any programming experience whatsoever. No special software is required to buy for the course. You can use the free Visual Studio community edition from Microsoft. You will be guided about which version is required and how to install the same.

The course is all about learning how to develop applications starting from fundamental stage to real world using the C# programming language. Why use C# instead of C++, Java, ActionScript, or some other programming language you may have heard of? First, using C# lets us use the Microsoft Open Source .NET Framework and many 3rd party frameworks, which help us quickly develop applications for mainly Windows and other platforms too. Second, it is the main pillar for building many other applications in .NET for various platforms like building web apps using ASP.NET Web Forms and MVC, Mobile applications using windows phone sdk, Game applications using XNA & XBOX SDK Framework, Cloud applications using Windows Azure and many more And finally, C# is a really good language for learning how to program.

That learning how to program comment is important because our course doesn't assume you have any previous programming experience. Don't worry if you've never written code before; we'll start at the very beginning and work our way up to building industry oriented applications along with project by the end of the course. Throughout the course you'll learn core programming concepts that apply to lots of programming languages, including C#, and you'll also learn how to apply those concepts when you develop different types of applications like Console Applications, classical windows forms applications, modern Windows Presentation Foundation & Distributed programming by combining all of them.

Computer programming is really fun in general, and programming with C# Language using Visual Studio is even better & simpler!

Learning C# 10.0:

There are 4 phases to learn C# 10.0.

1. Introduction to .NET Core & Language Fundamentals
2. Application Development and C# 10.0 new features
3. Files, Database’s, XML & Cloud Storage
4. Distributed Programming

Pre-Requisites Duration : 45 days
Course Materials & Textbooks : 1. Running Notes in class, 2. Pdfs, 3. HandOuts, 4. Certification Material in Pdfs will be provided for this course.
Reference Websites:www.dotnetgurukul.com
Instructor: Praveen Kumar M Learning from Praveen Sir, is always different and you will experience it within few sessions – you attend and work regularly as per given guidance then you will be best in any team that you work in any company – assured.

C# Course Syllabus by Praveen

Phase I – Introduction to .NET Core & Language Fundamentals

.NET Core & Languages

- Introduction & Evolution of .NET Core
- .NET Core, Platform and Technology
- Pros & Cons of .NET
- Comparisons .Net Framework Vs .Net Core Vs .Net Standard
- .NET Languages – C#, VB & others
- Expressing C# models in UML
- Building Project with .NET Core

C# & its Role in .NET Development

- Comparing different versions of C#
- as Basic Language
- Unique Features – Current Trend
- Importance in .NET Development

Visual Studio .NET, TFS, GitHub & others

- IDE software's & their role
- Team Development for complete SDLC
- TFS, GitHub as most essential
- Installing, configuring lively with local & remote development
- Understanding complete project environment

Programming Basics using C# & VB.NET

- Console Applications
- Current & Latest Versions
- Program structure
- Solution – Project – files
- Type System in .NET – CTS & CLS , NET Core CLI , .NET Standards
- CSC,VBC compilers etc
- Programming Structures

Data types and control constructs

- Declaring Implicit and Explicit variables
- Value and Reference types
- Unicode characters and strings

Defining and calling methods

- The Main method specification
- Passing arguments and returning values
- The scope and lifetime of variables
- Named and symbolic methods
- Handling exceptions
- Recovering resources

Object Oriented Programming – I

- Need for Object Oriented & current programming Model
- .NET as Component Oriented Development
- Creating Classes & Objects according to Component Specification
- Developing C# Classes
- Defining classes
- Encapsulating attributes with methods and properties
- Providing consistent initialization using constructors
- Overloading methods and constructors
- Achieving reuse through inheritance and polymorphism

Creating and using objects

- Allocating object memory with new
- Passing initial values to constructors

- Choosing value or reference allocation
- Invoking methods and accessing properties

Object Oriented Programming – II

Exposing interfaces

- Defining an interface specification
- Interface polymorphism
- Events and delegates
- Classes and Methods, Classes and Inheritance
- Namespaces and Referencing Libraries
- Understanding Scope and Utilizing Access Modifiers

Component Development of .NET

- Component Features of .NET
- Manifests and Assemblies
- .NET Assembly and MetaModel
- Creating and calling custom components
- Extending System.ComponentModel.Component
- Interfacing legacy components
- Accessing COM/DCOM
- Tools for forward and backward compatibility
- Calling existing components
- Coordinating components through the CLI
- Accessing metadata
- Handling cross-language differences

Handling Exceptions

- Error Handling and its importance
- Exceptions Hierarchy
- Creating and throwing exception types
- Multiple Exception Types.

Phase II – Application Development and C# 10.0 new features

Working with Collections & Generics

- ArrayList class
- Generics and behaviour
- List class, Generics and List
- Dictionary class
- HashTable,Stack and other collection classes

Starting with classical Windows Forms Development

- Understanding GUI Development
- Understanding Event Driven Programming
- Working with controls – Types of Windows Forms Controls

Working with WPF as current Desktop development

- WPF as more preferred Desktop Development
- XAML as new Design Markup Language
- WPF Controls & Concepts

Combining Object Oriented Concepts with WPF

- Building a simple module

SOLID Principles in C#

- Understand the basic concepts and principles of software architecture
- Learn how to organize code in different layers and tie them together
- Recognize the 4 Meta-principles of software architecture
- Demonstrate an understanding of the SOLID principles of object-oriented design
- Develop simple yet powerful C# code that is based on the above principles
- Outline the best practices of software architecture design

  - Construct maintainable and extendable code that is easy to debug, modify and reuse
  - Find new ways to apply the concepts of software architecture to create efficient applications

- **C# 10.0 New Features**
  - Record structs
  - Improvements of structure types
  - Interpolated string handlers
  - global using directives
  - File-scoped namespace declaration
  - Extended property patterns
  - Improvements on lambda expressions
  - Allow const interpolated strings
  - Record types can seal ToString()
  - Improved definite assignment
  - Allow both assignment and declaration in the same deconstruction
  - Allow AsyncMethodBuilder attribute on methods
  - CallerArgumentExpression attribute
  - Enhanced #line pragma
  - Warning wave 6

Phase III (Files , Database’ s, XML & Cloud Storage)

- **Working with Storage – Understand various storage options**
- **Files & Streams as most basic & important storage options**
  - File Handling in C#.
  - Streams & some basic Classes.
  - System.IO
- **ADO.NET to work with Database’s**
  - Databases overview from C#.
  - ADO.Net and C#.
  - ADO.Net as the new model for building db applications. (old DAO,RDO)
  - Advantages and Disadvantages of ADO.Net.
  - Different ADO.NET versions starting from v1.0 to 7.0
  - Drawbacks of ADO.NET 2.0 model and new models available for development starting from v3.0 to v4.0.
  - Starting with ADO.NET 2.0
  - System.Data, System.Data.oledb, System.data.sqlclient, System.Data.OracleCl namespaces.....
  - Command Object in ADO.Net, Running commands and Stored Procedures using Command Object.
  - Datasets, In and Out of Datasets. Complete usage with examples.
  - Layers & N-Tier architecture. Demo combining OOPS, ADO.NET & UI.
  - Database Concepts Continued…
- **ADO.NET Linq To Sql**
  - ORM implementation for 1st time in MS Environment
  - ORM detailed – its advantages and disadvantages
  - LINQ and its importance in .NET
  - Extension methods
  - Object & collection initialization syntax

- Anonymous types
  - Lambda expressions
  - Various Scenarios with Linq to Sql
- **ADO.NET Entity Framework**
  - Rich Implementation of ORM
  - Most advanced environment for DB handling
  - Design Patterns support and implementation
  - Repository, Unit of Work patterns
  - Linq , Linq to Entities in Detail
- **XML, System.Xml and its classes**
  - XML and its uses.
  - DataSet with XML
  - System.XMl namespace and its consumption in our applications.
  - XmlTextReader, XmlTextWriter, XmlDocument as key objects.
- **LINQ To XML**
- **Understanding Cloud Storage & programming with Cloud Storage**

Phase IV (Distributed Programming)

- **Building Multi-Tier applications**
  - Leveraging solid architectures (MVC and EDM)
  - Substituting the user interface
  - Coding industry-standard design patterns in C#
- **Windows Services**
  - Understanding Windows Services
  - Creating Windows Services
  - Installing and Uninstalling Windows services
- **DCOM, RMI & CORBA**
- **.NET Remoting ---Micros servies**
  - Understanding Distributed Architecture
  - Drawbacks of DCOM
  - Remoting Advantages
  - Remoting Vs web services
  - Creating & Using Remote Applications
- **WCF as preferred Distributed Environment**
- **Remoting/WCF in .NET**
- **Deployment of Applications**
  - **Setup and Packaging of Applications**
  - **Fast Deployment Methods**