# C#.NET TRAINING

## Overview

C#.NET training teaches non-experienced people how to create the Solutions/Applications using C#.NET. C#.NET is Microsoft's entry into the world of managed programming. Using a syntax that is deliberately from Java, C++ and C, C#.NET achieves a natural trade-off of terseness and clarity, enabling programmers to express concepts in a clear and maintainable form. The recent enhancements to the language have made it even more powerful than before, allowing programmers to work with C#.NET in both an Object-Oriented and partially functional style.

## Prerequisite

No prior experience is presumed.

## Course Duration

- Normal Track 60 Working days, daily one and half hours
- Fast Track 40 Working days, daily two hours

## Course Contents

1. Learn the fundamentals of C# programming in Visual Studio.
2. Using .Net Framework
3. Working with variables, data types
4. Work with standard programming skills
5. Exception Handling in C#
6. Object oriented programming in C#
7. Object oriented techniques
8. Working with Arrays
9. Create Generic classes and methods.
10. Collections in C#
11. Generate and test your own classes using the Class Designer and Object Test Bench tools.
12. Use delegate types to provide flexibility and type safety.
13. Use anonymous types, lambda expressions, extension methods, object initializers, and implicit type declarations

# Course Contents Details

**Introduction/Overview of .Net**

- Introduction to .Net
- Platform for the .Net
- Drawbacks of Current Trend
- Net Framework – BCL & CLR l Key design goals
- CLR, CTS, MSIL & other tools.
- Multiple Language Interaction & support l Moving from Project to Assemblies…
- Security in .NET – CAS

**.Net Framework [Advanced]**

- Advantages/Disadvantages
- Features of .Net
- Assemblies in Detail
- GAC, Strong Names
- Language Interop
- Reflection

**Visual C#.Net Language**

- Advantages/Disadvantages
- Why C#/ Why Not C#
- Where does C# Fit in
- C, C++ to Visual C#
- Features of C#
- .NET Namespaces

**.Net Installations/C#.Net**

- .NET Versions – 1.1/2.0/3.0/3.5 Beta
- Visual Studio.NET 2003/2005/Orcas/2008
- Windows Vista – New Look
- Gadgets/SideBars/UAC – relation with .NET
- Hardware/Software Requirements
- FAQ's with detailed answers

**Programming Using Visual C#.Net**

- The start of the application
- C#.Net Program Design
- Variables and types
- Value types and reference types (CTS)
- Strings and arrays
- The Console class
- String formatting
- Statements and flows
- Programming Structures
- Command-line arguments
- VS.NET to Create C#.NET Apps
- C# 3.0/3.5 features – Implicit types , Extention Methods and more

**Introduction To Windows Forms – I**

- Windows forms library – WinForms
- Layout Enhancements
- Forms and controls – Hierarchy
- Creating simple GUI by hand
- Event handling
- Basic controls
- Windows forms – buttons, check boxes, radio buttons, panels, group boxes, list boxes, picture boxes…

**Windows Forms – II**

- Menus
- Built-in dialog boxes and printing
- Extender Controls
- ToolStrips, StatusStrips and progress bars
- A new MDI forms strategy
- Inheritance with forms
- New Controls – Web Browser, Property Grid etc

**Object Oriented Concepts (Basic)**

- Classes & objects
- Abstract & override methods
- Creating and using your own classes l Data members and member methods l Instantiate an object
- This keyword
- Properties – Read Only Write Only…
- Build process using windows class library l Generate classes for other clients
- How to use classes as part of project

# Course Contents

**Object Oriented Concepts (Advanced)**

- Accessibility levels, specifiers
- Constructors
- Method overloading
- Class (static) variables & methods
- Object destruction
- 'ref' and 'out' parameters
- Constant values
- Enumerations
- Inheritance and Polymorphism
- The root of all classes
- Creating derived classes
- Method overriding and hiding
- Polymorphism and virtual functions
- Casting objects
- Abstract classes
- Sealed classes
- Static classes

**Object Oriented Concepts (Implementation Oriented)**

- Case Studies
- Class Diagram in VS.Net
- Refactoring & others
- FAQ's

**Error Handling**

- Unstructured error handling support
- Structured error handling
- Error categories
- Debugging the application
- Debug and Trace classes
- Code Optimization
- Testing and strategies

**Ado.Net 1.1/2.0/Linq**

- History and background
- From DAO to ADO.NET
- ADO.NET LINQ
- ADO.NET design goals
- The ADO.NET architecture and its components
- ADO.NET in relation to the other .NET tools
- DataSet in RealTime Scenarious

**Ado.Net Components**

- Connected and disconnected environment
- ADO.NET object model
- Data sources, providers and connections
- Commands and data readers
- Data sets and data adaptors
- Data tables, rows and columns
- Constraints and relations
- Data-centric applications – New ADO.Net Hierarchy

**Data Sources And .Net Data Providers**

- Connecting to a data source
- SQL Server .NET data provider
- OLE-DB .NET data provider
- Connections and connection strings
- SQL-Server integrated security
- Connection pooling
- ADO.NET exceptions

**Accessing Data In The Connected Environment**

- Commands
- Creating and executing commands
- Reading data using a data reader
- Batch queries & single result queries
- Parameterised queries (input & output pa rameters)
- Adding, editing and deleting data
- Stored procedures

**Accessing Data In Disconnected Environment**

- Why using a disconnected environment?
- DataSet and DataAdapter features
- Filling data sets using data adapters
- Read data using data tables, rows and columns
- Batch queries and data sets
- Visual Studio Data Menu & tools
- DataSet Navigation
- DataSet Functionality
- DataSet Optimistic Concurrency

**Sorting, Searching And Filtering**

- Searching in data sets
- Find on primary keys

- Searching on any column
- Searching on row state
- Wildcard searches
- Sorting and filtering using data views
- Searching in a data view

**Live Case Study and Implementation Of ADO.NET in N-Tier**

- Client Server Basics
- N-Tier – Classical and New
- N-Tier importance w.r.t other .Net technologies
- Build User Interface Layer – importance
- Business Layer in N-Tier – advantages & disadvantages
- Data Access Layer – Generic/Specific Advantages
- N-Tier DataBase Application
- SQLHelper, CodeSmith etc tools awarene ss

**XML**

- XML Basics – Importance in Todays world
- XML designers/support in VS.NET
- XML Derived Technologies – XSD ,XSL , SOAP ,WSDL

**XML IN .NET**

- System.Xml Namespace
- Stream Model XML
- XML DOM
- XmlTextReader,TextWriter
- XmlTextReader, TextWriter XMLDocument/ XmlDataDocumentClasses
- Dom Objects – XMLNode/XMLNodeList
- XPATH- Query Language for XML
- DOM – Navigation & Access Case Studies

**Creating Custom Windows Controls**

- About user – defined controls
- Understanding the control class with Con tainer
- Add Properties/Methods/Events to Control
- Pack & use control in other windows applications
- Create & implement a windows control

**Windows Services***

- Understanding services
- Creating windows services
- Setting properties
- Compile, run & install services
- Event log services

**.Net Remoting**

- About Distributed Applications
- COM/DCOM in Distributed Environment
- Drawbacks of DCOM
- .NET Remoting – New distributed environment
- Advantages & Disadvantages
- Remoting – Web Services comparisons
- MBR, MBV
- Channels
- Formatters
- Programming Model – Object Styles & Lifestyles
- Activation
- Case Study – Implementation using Remoting

**Crystal Reports***

- Reporting Need in the application
- Crystal Reports – Reporting Tool
- Different Versions of Crystal Reports
- Developing a Crystal Report
- Different ways to Invoke/Deploy Crystal Reports

**Application Deployment**

- Packaging Code
- ClickOnce Deployment
- Configuring the .Net framework
- Deploying the application in Web Server
- Deployment – other methods