DOT NET : LEVEL 1

OBJECTIVES

The DWIT Training - Dot Net course is targeted for beginners who want to:

- Learn how to think and write meaningful piece of code in Dot Net.
- Learn how to read DOT NETcode that has been written by somebody else.
- Learn how to map literary description of a problem (requirement) to an application/library coded in Dot Net. In summary, this course teaches how to program using Dot Net programming language.

This is a core basic level course that is essential for anyone who have no prior programming experience but wish to be a professional Dot Net engineer in future

TARGET GROUP

Anyone who has some basic knowledge about programming and wants to learn to write applications in Dot Net for any purpose e.g. curiosity, hobby, to complete an academic project, to work towards a career as Dot Net programmer, to help in project management, etc.

Prerequisites

- Basic knowledge about programming, bits/bytes, procedures, classes, computer architecture, etc. If you just have a theoretical knowledge that is perfectly okay but you should have strong convictions on what programming is, and what you hope to achieve from this class.
- Willing and eager to spend at least 10-20 hours (varying from student-to-student) per week outside of the training class to read/write codes in Dot Net (self-study and practice).
- There are no prior educational level requirement for this course. Anyone from 10+2 student to someone who is doing her PHD in Genetic Engineering is welcome to take this course.
- If you are only interested in theory and have no interest/patience in spending at least 10 hours every week throughout the duration of the course, then this course is clearly not for you.
- If you have absolutely no idea about programming or do not see yourself doing programming in the next six -odd months, then this class may not be for you!

TRAINING METHOD

The course is spread over 40 hours that consists of lecture and lab work. There will be approximately 10 hours of lectures and 30 hours of hands-on lab work.

- Lab exercises are mandatory, have a fixed deadline, and are graded. The course puts heavy emphasis on lab exercises because software programming can only be learnt well by explicitly putting into practice the principles that have been taught (i.e. in simpler terms – by doing lots and lots of coding). Late submission (past the deadline) of exercises incur some penalty from total points.
- Instructors may provide relevant lecture/lab notes to students as (and when) necessary in the form of printed handouts and or via emails.
- Instructors may provide supplementary code snippets to students via email or in lab class to support the theory and or lab material that is being taught.
- At the end of the course, students may have to give an exam (which will be optional), that will test their knowledge on the material covered during the course. This exam may be practical and/or theoretical and is mandatory for any student wishing to join a higher level.
- Students are graded on the basis of attendance, lab exercises and exam in the increasing order of importance.

In summary, the only effective way to learn programming is to write lots of code. So in order to really make this training productive, students are encouraged to spend as much time as necessary to complete the lab exercises on time. As part of the course, students will spend at least 30 hours in the lab but specially if you are new to programming or are coming from a non-computer-science background, it is recommended that you spend at least 10-20 hours per week outside of the class on your own to practice coding in Dot Net.

COURSE DURATION

- 40 hours
- Classes
  -Morning/Evening

COURSE BREAKDOWN

Theory

- **STARTING WITH VISUAL STUDIO 2010**
  - Creating Console Application Project
  - Project vs. Solution
  - How to compile the project?

- **THE MAIN() METHOD – WHERE IT ALL BEGINS**
- **CLASS IN BRIEF**
- **WHAT IS A METHOD?**
  - Argument List
  - Return Type
  - Breaking down solution (to a problem) to one or more methods
- **NAMESPACE**
  - Alias
  - Global scope
- **VARIABLES**
- **BASE CLASSES – BASIC**
  - How to write to Console?
  - How to read from Console?
- **WHAT IS FLOW CONTROL AND WHY DO WE NEED IT?**
- **FLOW CONTROL – CONDITIONAL STATEMENTS**
  - If-Else
  - Switch
- **FLOW CONTROL – ITERATION AND JUMPS**
  - For
  - While
  - Do While
  - Break
  - Continue
  - Goto
  - Foreach
- **OPERATORS - BASIC**
  - Arithmetic
  - Increment/Decrement
  - Comparison
  - Logical
  - Bitwise
  - Bit Shifting
  - Assignment

- **SCOPE OF A VARIABLE**
- **CONSTANTS**
- **DATA TYPES**
  - Value Types
  - Reference Types
- **VALUE TYPES IN DETAIL**
  - Signed vs. Unsigned
  - byte, sbyte
  - short, ushort
  - int, uint
  - long, ulong
  - float
  - double
  - decimal
  - bool
  - char
- **ARRAYS**
  - Single Dimensional Arrays, introducing the [] operator
  - Multiple Dimensional Arrays
  - Jagged Arrays
  - Array is a Reference Type
- STRING
  - String is a Reference Type
  - What is a string made up of – understanding the char type
  - Ways of constructing strings
  - How to copy one string to another
  - Using the [] operator
  - Converting strings from lower case to upper and vice-versa
  - Searching for specific characters in a string
  - Searching for specific words in a sentence
  - Complex String operations
  - StringBuilder
  - Format Strings
  - Immutability
- **ENUMERATIONS**

- **PREPROCESSOR DIRECTIVES**
  - Define
  - Undef
  - If, Elif, Else, Endif
  - Warning
  - Error
  - Region, Endregion
- **COMPILING WITH MULTIPLE MAIN() METHODS**
- **VISUAL STUDIO 2010 REVISITED – HOW TO DEBUG YOUR CODE**
- **STRUCTURES**
- **HOW TO EFFECTIVELY DESIGN AND WRITE YOUR OWN CLASSES?**
- **WHERE TO GO NEXT?**

Labs

- Lab assignments will focus on the practice and mastery of contents covered in the lectures; and introduce critical and fundamental problem solving techniques to the students.

DISCLAIMER

Please note that Deerwalk Institute of Technology reserves the right to change the course syllabus of DWIT Training - Dot Net – Level 1 course at any time without prior notification.

DOT NET : LEVEL 2

OBJECTIVES

The DWIT Training - Dot Net – Level 2 course is targeted for trainees:

• Who have had some prior beginner level hands-on programming experience in Dot Net programming language.
• Who have programming experience in some other programming language (e.g. Dot Net, Obj-C, PHP, C, C++, etc.) and want to learn Dot Net .

TARGET GROUP

- High school and university students (undergraduate, graduate, etc.) who want to do coursework (e.g. project, etc.) in DOT NET.
- Someone who has experience in some other programming language (e.g. C/C++, PHP, Perl, etc.), but has never done programming in ANDROID.
- Someone who is already working as a professional VB.NET developer and wants to switch to ANDROID.
- Someone who did her undergraduate in Economics, has been working in Media sector since graduation, and also working as a professional freelance PHP developer.
- Electrical/Electronic undergraduates in their 3rd semester who want to beef up their software skills prior to graduation.

Prerequisites

- Successfully complete the entrance test with score of at least 40% (for trainees directly applying to this level).
- Successfully complete the DWIT Training - Dot Net – Level 1 course (not applicable to trainees directly applying to this level).
- Successfully complete the interview.
- o Willing and eager to spend at least 10-20 hours (varying from student-to-student) per week outside of the training class to read/write codes in Dot Net (self-study and practice).

TRAINING METHOD

The course is spread over 40 hours that consists of approximately 15 hours of lecture and 25 hours of hands-on lab work.

- Lab exercises are mandatory, have a fixed deadline, and are graded. The course puts heavy emphasis on lab exercises because software programming can only be learnt well by explicitly putting into practice the principles that have been taught (i.e. in simpler terms – by doing lots and lots of coding). Late submission (past the deadline) of exercises incur some penalty from total points.
- Instructors may provide relevant lecture/lab notes to students as (and when) necessary in the form of printed handouts and or via emails.
- Instructors may provide supplementary code snippets to students via email or in lab class to support the theory and or lab material that is being taught.
- At the end of the course, students may have to give an exam (which will be optional), that will test their knowledge on the material covered during the course. This exam may be practical and/or theoretical and is mandatory for any student wishing to join a higher level.
- Students are graded on the basis of attendance, lab exercises and exam in the increasing order of importance.

In summary, the only effective way to learn programming is to write lots of code. So in order to really make this training productive, students are encouraged to spend as much time as necessary to complete the lab exercises on time. As part of the course, students will spend at least 30 hours in the lab but specially if you are new to programming or are coming from a non-computer-science background, it is recommended that you spend at least 10-20 hours per week outside of the class on your own to practice coding in Dot Net.

COURSE DURATION

- 40 hours
- Classes
  - Morning/Evening

- **CLASS IN DETAIL**
  - Data Members
  - Function Members
  - Access Modifiers
  - Data Encapsulation
  - Set and Get methods
  - Passing parameters by Value
  - Passing parameters by Reference
  - Using keywords ref, out, and params in methods
  - Named Arguments
  - Optional Arguments
  - Method Overloading
  - Properties and Accessors
  - Constructors (and Destructors)
  - Partial Classes
  - Static Class and Static Methods
  - Static Constructor, readonly fields

- **OBJECTORIENTED PROGRAMMING**
  - Implementation Inheritance
  - Interface Inheritance
  - Multiple Inheritance
  - The Object class
  - Polymorphism, Virtual Methods
  - Abstract Class and Abstract Methods
  - Sealed Class
  - Exceptions
  - Structures Revisited
  - Using Constructors
  - Inheritance
  - Coding Conventions and Guidelines
  - Properly Writing Comments

- **ADVANCED**

  - Use of operators – checked, unchecked
  - Use of operators – as, is, typeof, unsafe, sizeof
  - Nullable types and operations
  - Null Coalescing operator
  - Type Inference
  - Anonymous Type
  - Boxing and Unboxing
  - Data Conversions – Implicit and Explicit

- Four ways to compare objects for equality
- Operator Overloading
- User Defined Casts
- Indexers
- Generics
- Type Safety
- Constraints
- Default
- Inheritance
- Interface
- Statics
- Structures
- Delegates
- Covariance
- Contra-variance
- Delegates and Events

- **WR ITING W INDOWS FORMS APPLICATIONS**
  - What makes Windows Forms application different from Console Application
  - Human Computer Interaction
  - Slight Diversion: The WinMain() loop
  - Class Hierarchy
  - Controls
    - User Interface
    - User Interaction
    - How it all works
  - Forms class
  - Standard Controls
    - Button
    - CheckBox
    - CheckedListBox
    - ComboBox
    - DataGridView
    - DateTimePicker
    - ErrorProvider
    - ImageList
    - Label
    - ListBox
    - ListView
    - MaskedTextBox
    - MenuStrip
    - Panel
    - PictureBox
    - ProgressBar
    - RadioButton

- RichTextBox
- TabControl
- TabPages
- TextBox

Labs

Lab assignments will focus on the practice and mastery of contents covered in the lectures; and introduce critical and fundamental problem solving techniques to the students.

DISCLAIMER

Please note that Deerwalk Institute of Technology reserves the right to change the course syllabus of DWIT Training - Dot Net – Level 2 course at any time without prior notification.

DOT NET : LEVEL 3

OBJECTIVES

This course builds on the foundation laid by DWIT Training - Dot Net – Level 3 to prepare trainees for a career as Dot Net software engineer.

TARGET GROUP

Prerequisites

- Successfully completed the DWIT Training - Dot Net – Level 3 or obtained at least 40% score on the entrance exam.
- The latter case applies for new students that are directly attempting this training.
- Successfully complete the interview.
- Willing and eager to spend at least 10-20 hours (varying from student-to-student) per week outside of the training class to read/write codes in Dot Net (self-study and practice).
- Please note that this is a lab intensive course where the students will be expected to work on lab exercises for approximately half the duration of the session.

TRAINING METHOD

- The course is spread over 40 hours that consists of approximately 15 hours of lecture and 25 hours of lab work.
- Lab exercises are mandatory, have a fixed deadline, and are graded. The course puts heavy emphasis on lab exercises because software programming can only be learnt well by explicitly putting into practice the principles that have been taught (i.e. in simpler terms – by doing lots and lots of coding). Late submission (past the deadline) of exercises incur some penalty from total points.
- Instructors may provide relevant lecture/lab notes to students as (and when) necessary in the form of printed handouts and or via emails.
- Instructors may provide supplementary code snippets to students via email or in lab class to support the theory and or lab material that is being taught.

- At the end of the course, students may have to give an exam (which will be optional), that will test their knowledge on the material covered during the course. This exam may be practical and/or theoretical and is mandatory for any student wishing to join a higher level.

- Students are graded on the basis of attendance, lab exercises and exam in the increasing order of importance.

COURSE DURATION

- 40 hours
- Classes
  -Morning/Evening

COURSE BREAKDOWN

- **DATA STRUCTURES**
  - Introduction
  - Key Interfaces
  - Array Class
  - Lists
  - Queue
  - Stack
  - Linked List
  - Sorted List
  - Dictionaries
  - Sets
  - Bit Arrays
  - Trees
  - Graphs
  - Deciding which data structure to use
  - Thinking about performance

- **MANAGING FILES**
  - File I/O Revisited
  - Serialize / Deserialize
  - Moving, Copying and Deleting Files
  - Collecting Drive Information
  - Memory Mapped File

- **WORKING WITH XML**
  - Standards in .NET
  - XML I/O
  - DOM
  - XPATH

- **INSTRUMENTATION**
  - Event Logging
  - Tracing
  - Perfmon
  - Contracts

- **WRITING MULTI-THREADED APPLICATIONS**
  - Asynchronous Delegates
  - Thread Class
  - Thread Pools
  - Tasks
  - Race Conditions and Deadlock
  - Lock statement
  - WaitHandle
  - Mutex
  - Semaphore
  - Timers

- **WRITING NETWORK APPLICATIONS**

- **REFLECTION AND ASSEMBLIES**
  - Custom Attributes
  - System.Type
  - Assembly Class
  - Overview of Assemblies
  - What constitutes Assemblies?
  - Structure
  - Manifest
  - Attributes
  - Private, Shared, Satellite
  - Creating and Loading Assemblies
  - Application Domains
  - Versioning
  - GAC
  - Shared Assemblies
  - Strong Names
  - Delayed Signing

- **WINDOWS FORMS**
  - Multiple Document Interface
  - User Controls

- **INTRODUCTION TO ASP.NET**
  - What is ASP.NET?
  - ASP.NET Life-cycle explained
  - Web Sites vs. Web Applications
  - Managing States
  - Coding Models
  - ASP.NET Web Forms
  - ASP.NET Server Controls

Labs

- Lab assignments will focus on the practice and mastery of contents covered in the lectures; and introduce critical and fundamental problem solving techniques to the students.

DISCLAIMER

Please note that Deerwalk Institute of Technology reserves the right to change the course syllabus of DWIT Training - .NET – Level 3 course at any time without prior notification.