Chapter 1 Introduction C#

Chapter 2. Building C# Application

Chapter 3 Core C# Programming Construct

Chapter 4 Understanding Encapsulation

Chapter 5 understanding Inheritance and Polymorphism

Chapter 6 Exception Handling

Chapter 7 Interface

Chapter 8 Understanding Object Lifetime

Chapter 9 Collection And Generics

Chapter 10 Advanced C# Language Features

Chapter 11 Delegates, Events and Lambda Expression

Chapter 12 Linq

Chapter 13 Processes, AppDomains, and Load Contexts

Chapter 14 Multithreaded, Parallel, and Async

Chapter 16: Building and Configuring Class Libraries

Chapter 17: Type Reflection, Late Binding, Attribute, and Dynamic Types

Chapter 18: Understanding CIL and the Role of Dynamic Assemblies

Chapter 19: File I/O and Object Serialization

Chapter 20: Data Access with ADO.NET

Chapter 21: Introducing Entity Framework Core

Chapter 22: Exploring Entity Framework Core

Chapter 23: Build a Data Access Layer with Entity Framework Core

.net platform နဲ့ c# ကို 2002နှစ်မှာ စတင်မိတ်ဆက်ခဲ့တယ်။

.net platform မှာဆိုရင် c# အပြင် အခြား F#, vb.netလိုကောင်မျိုးတွေပါ ပါ၀င်တယ်။ အဲ့လိုပါ၀င်တဲ့တွက်ကြောင့် အဲ့programming language တခုနဲ့တခုကြားမှာ အပြန်အလှန်ချိတ်ဆက်ပီးသုံးနိုင်တယ်။ ဆိုလိုတာက c#နဲ့ရေးထားတာကို အခြား Vb.net, F#တွေက နေပီး ပြန်သုံးနိုင် ထပ်ပီးတော့ လုပ်ဆောင်ချက်အသစ်တွေထပ်တိုးနိုင်တာမျိုးကို လုပ်နိုင်တယ်။

စထွက်တော့ .net က window operating system မှာပဲ လုပ်ဆောင်နိုင်တယ်။

၂၀၁၆ .net coreကိုစတင်ကြေညာပီးနောက်မှာတော့ window system မှာတင်မဟုတ်ပဲ အခြား linux, iOSမှာပါ runလာနိုင်တယ်။

တခုသိသင့်တာက ကိုက .net version ကိုရွေးချယ်လိုက်တာနဲ့ အလိုလျောက် c# version ကိုလဲ ရွေးချယ်ပီးသားဆိုတာ သိရမယ်။

ဥပမာဆို c# 10ဆိုတာနဲ့ .net 6ကိုသုံးထားတယ်ဆိုတာ သိနိုင်တယ်။ အဲ့တာက ဘာတွေကွာလဲဆို version တခုမှာ သုံးလို့ရတဲ့ language ရဲ့ လုပ်ဆောင်ချက်တွေ လုပ်ဆောင်နိုင်မှုတွေကွာသွားတယ်။

**Key Benefits of the .NET Platform**

၁. Language တခုထပ်ပိုပီး support ပေးတယ် .net platformကိုသုံးပီး C# အပြင် F#, VB.Netတို့ကိုလဲ တည်ဆောက်နိုင်တယ်။

၂. Common runtime engineရဲ့ အကူအညီနဲ့ .net languagesတွေကြားမှာ တခုနဲ့တခုကို sharedလုပ်ပီးသုံးနိုင်တယ်။

အဲ့လိုမျိုးဘာလို့သုံးနိုင်လဲဆိုရင် Common Type System(CTS)ဆိုတဲ့ကောင်ရဲ့ အကူအညီနဲ့ကြောင်ဖြစ်တယ်။

CTSဆိုတာကို နောက်လာမည့်အပိုင်းမှာထပ်ပြောပါ့မယ်။

၃. Language Integration - ဆိုလိုတာက .net က cross language ကိုလုပ်ဆောင်ပေးတဲ့တွက်ကြောင့် .net language(C#, VB.net, F#) တွေကြားမှာ exception handling, inheritance တွေကို လုပ်ဆောင်နိုင်တယ်။

၄. BCL(Base Class Library)လို့ခေါ်တဲ့ build in လုပ်ဆောင်နိုင်တဲ့ library တွေများတယ် ဆိုလိုတာက printထုတ်ချင်တယ်ဆို codeတွေများကြီးရေးနေစရာမလိုပဲ print libraryတခုကိုပဲ ကောက်ခေါ်လိုက်ရင် ရနိုင်တာမျိုးဖြစ်တယ်။

၅.CLI(Command Line Interface)လို့ခေါ်တဲ့ command line ကိုပဲသုံးပီး .net application တွေကို developmentလုပ်ဆောင်နိုင်တယ်။ ဆ်ုလိုတာက application တခုဆောက်တာ လိုအပ်တဲ့ package တွေထည့်တာ publish လုပ်တာတွေကို command line နဲ့ပဲလုပ်ဆောင်နိုင်တာကိုဆိုလိုတာဖြစ်တယ်။

Understanding the .NET Support Lifecycle