

## **.NET: beprövade lösningar och designmönster - 4 dagar**

### **Skapa lyckade applikationer med beprövade metoder**

*kurser 511*

- Du får lära dig att**
- Implementera beprövade metoder för att bygga anpassningsbara, pålitliga och effektiva applikationer för .NET Web och smarta klienter
  - Lösa komplexa programmeringsproblem med industristandardens designmönster
  - Skapa buggfri kod med hjälp av testdriven utveckling och Visual Studio
  - Skapa flerlayersarkitekturer för ökad återanvändbarhet
  - Använda bästa praxis för att förbättra klassdesign och säkerhet
  - Förenkla komplexa och repetitiva uppgifter med hjälp av .NET reflection och anpassade generiska klasser
- Sammanfattning** Kunskap om enbart .NET-språken och -biblioteken är inte tillräcklig för att utveckla robusta applikationer. Beprövade designmönster och metoder, som bygger på expertkunskap, hjälper dig att skapa applikationer som vilar på en solid grund. Denna kurs ger dig de färdigheter du behöver för att lösa programutvecklingsproblem i verkliga livet samt ta fram snabba och pålitliga applikationer.
- Vem bör delta** Programmerare, systemarkitekter och utvecklare av .NET-applikationer. Programmeringserfarenhet i nivå med kurs 419, "C#-programmering", eller kurs 503, "Visual Basic 2008: programmering", förutsätts.
- Praktiska övningar** Du får erfarenhet av att implementera .NET:s bästa praxis och designmönster. Övningarna som utförs i VB eller C#, omfattar:
- Lösa olika bearbetningsproblem med hjälp av Strategy-mönstret
  - Förenkla ett komplext system med Facade-mönstret
  - Använda Microsoft Entity Framework för dataåtkomst och uppdateringar
  - Skapa en Business Domain Object Model
  - Skapa automatiserade testfall
  - Konstruera ett testbart användargränssnitt med mönstret Model View Controller
  - Fånga upp och återanvända tester inom Visual Studio

## .NET: beprövade lösningar och designmönster - 4 dagar

### Skapa lyckade applikationer med beprövade metoder

*kurser 511*

#### Introduktion

- Programmera med hjälp av "best practices"
- Förenkla program med designmönster

#### Förenkla komplex programmering med beprövade designmönster

##### Tillämpa enkla gränssnitt på invecklade algoritmer

- Föreina gränssnitt från subsystem för smidigare användning
- Implementera Facade-mönster

#### Variera funktionaliteten genom att programmera till gränssnitt

- Förbättra anpassningsbarheten och flexibiliteten i tillämpningen
- Utnyttja Strategy-mönstret

#### Utöka objektbeteende dynamiskt

- Öka funktionalitet utan att påverka befintlig kod
- Sammanställa objekt med Decorator-mönstret

#### Åstadkomma återanvändning och flexibilitet

- Undvika duplicering av kod genom att forma en bas för en algoritm
- Använda Template Method-mönstret

#### Samordna oförenliga klasser

- Omvandla ett gränssnitt för att ge mervärde till befintlig kod
- Utnyttja Adapter-mönstret

#### Tillämpa testdrivna utvecklingsmetoder

##### Automatisera enhetstestning

- Korta utvecklingstiderna med automatiserade tester
- Förbättra kvaliteten med konsekvent testtäckning
- Eliminera regressionsfel med återanvändbara tester

##### Integrera testning och kodning

- Tillämpa programmeringspraxis att testa först för att driva på koddesign
- Generera direkt godkännande för att förbättra kodkvalitet och korta funktioners utvecklingscykel
- Organisera, koordinera och köra testfall med Visual Studio

#### Utforma en flerlayersapplikation

#### Utforma applikationsarkitekturen

- Skikta arkitekturer för återanvändbarhet, varaktighet och skalbarhet
- Få tillgång till data från affärsskiktet
- Separera objektskapande med Factory-mönstret
- Bevara objektidentitet med mönstret Identity Map

#### Programmera lager i applikationer

- Konstruera en rik klientapplikation med Observer-mönstret
- Skapa återanvändbara affärsbasklasser och gränssnitt med mönstret Layer Supertype
- Organisera tillståndsrika applikationer med mönstret State
- Omstrukturera databastabeller utan att påverka applikationskoden

#### Utforma ett affärsområde

- Frigöra en rikhaltig affärsobjektmodell från databasstrukturer med Domain Model-mönstret
- Tilldela rikhaltiga affärsobjekt till databastabeller med Data Mapper-mönstret
- Bryta ner hierarkier för affärsobjekt med mönstret Inheritance Mapper
- Använda deklarativ programmering i Microsoft Entity Framework för att implementera datamappingsklasser

#### Tillämpa bästa praxis

##### Konfigurera kodåtkomstsäkerhet

- Godkänna kod
- Behörighetsuppsättningar
- Bevis
- Behörighetsnivåer
- Kodgrupper
- Be om tillstånd
- Få tillgång till isolerad lagring

##### Bästa praxis för klassdesign

- Skydda mot rigiditet med Öppen/Stängd-principen
- Extrahera nya klasser med Single Responsibility-principen
- Effektivt utnyttjande av arv

##### Automatisera repetitiva uppgifter

##### Använda kod för dataåtkomst

- Minska åtkomstkod för databaser med hjälp av Entity Navigation Properties
- Eliminera databasens uppdateringskod med Entity change tracking

#### Förbättra koden i applikationer

- Automatisera designgranskning med FxCop och Visual Studio Analysis
- Eliminera kodduplicering med genom refaktorisering till designmönster