

## Utveckla Enterprise Java-applikationer med Spring och Hibernate® - 4 dagar

*kurser 517*

- Du får lära dig att**
- Utveckla skalbara företagsapplikationer i Java med Spring 3.1 och Hibernate
  - Bygga infrastruktur för applikationer med Inversion of Control (IoC) och Dependency Injection (DI)
  - Modularisera funktionaliteten med hjälp av Aspect Oriented Programming (AOP)
  - Lägga till ett flexibelt användargränssnitt med Spring Model View Controller (MVC)
  - Implementera objektpersistens med Hibernate
  - Optimera dataåtkomst med Hibernate Query Language (HQL)
- Sammanfattning** Att utveckla kraftfulla företagsapplikationer i Java är en komplex process som ofta kräver omfattande infrastrukturkod. På den här kursen får Java-utvecklare lära sig hur man snabbt skapar företagsapplikationer i Java med de standardiserade ramverken Spring och Hibernate. Genom intensiva övningar får deltagarna lära sig implementera effektiva applikationer, samtidigt som utvecklingstiden reduceras.
- Vem bör delta** Kursen är värdefull för utvecklare, programmerare, ingenjörer, chefer och de som är inblandade i utvecklingen av komplexa företagsapplikationer i Java. Kunskaper i Java-programmering i nivå med kurs 471, "Java-programmering: omfattande introduktion", förutsätts.
- Praktiska övningar** Övningar som ger praktisk erfarenhet av att skapa företagsapplikationer omfattar att:
- Införa beroenden med Spring IoC
  - Effektivisera utvecklingen med Spring 3.1 JDBC template support
  - Använda modular kod med AspectJ style AOP
  - Implementera ett webblager med Spring MVC
  - Hantera transaktioner med Spring 3.1 annotationer
  - Lagra och hämta dataobjekt med Hibernate
  - Integrera Spring och Hibernate

## Utveckla Enterprise Java-applikationer med Spring och Hibernate® - 4 dagar

kurser 517

### Introduktion till ramverket Spring

#### Grundläggande Spring-arkitektur

- Identifiera delarna i Spring-arkitekturen
- Definiera n-tier-applikationsarkitekturen

#### Använda Inversion of Control (IoC) och Dependency Injection (DI)

- Delegera objektskapande till Spring bean factory
- Kontrollera skapande av beans scopes och factory.metoder
- Initialisera och förstöra beans

#### Minimera kod med Aspect-Oriented Programming (AOP)

##### Transparent tillämpa vanlig funktionalitet

- Utvärdera fördelarna med AOP
- Definiera råd, pointcuts och advisors
- Minimera konfigurationen med Aopproxying

##### AspectJ style AOP

- AspectJ pointcut expression language
- Använda AspectJ style med annotationer
- Bygga aspekter med POJO:s och XML-schemabaserad konfiguration

#### Konstruera ett effektivt lager för dataåtkomst med Spring

##### Förenkla dataåtkomst med JDBC-mallar

- Effektivisera runaway-kod med JDBC-mallar
- Strukturera frågor och motringningar för underhållbarhet

##### Skilja ut lagret för dataåtkomst

- Underhålla mönstret Data Access Object (DAO)
- Uppnå implementeringsoberoende med plattformssagnostiska undantag

#### Hantera transaktioner

- Analysera Java EE transaktionsstöd
- Kontrollera transaktioner med Spring transaction manager
- Deklarera transaktionsregler med XML och kommentarer

#### Bygga ett webblager med Spring MVC

##### Bearbeta webbfrågningar

- Analysera Spring Model View Controller-arkitekturen (MVC)
- Konfigurera frågehanteringsflöde med annotationer
- Bearbeta kommandon, formulär och enkla guider
- Validering på serversidan

#### Återge svaret

- Upplösa vyer med ViewResolvers
- Spring JSP support
- Se teknikalternativ med Velocity

#### Bygga Ajax-kontroller

- Fastställa kraven för Ajax-kontroller
- Implementera REST style URL:er
- Returnera JSON-data

#### Lagra objekt med Hibernate

##### Integrera Hibernate

- Förenkla dataåtkomst med O/R-mappning
- Förstå Hibernate-arkitekturen
- Använda och konfigurera Hibernate

#### Generera Hibernate-applikationer

- Utveckla den persistenta klassen
- Definiera reglerna för avbildning i Hibernate
- Lagra och ta fram Java-objekt

#### Hantera komplexa objektrelationer

##### Hibernate Sessions roll

- Etablera ett sessionsobjekt där man kan använda trådar
- Definiera objektillstånd: övergående, persistent, bortkopplad

##### Avbilda samlingar

- Åtkomst och lagring av samlingar
- Bevara samlingsordningen för att uppnå dataintegritet

##### Strategier när man skapar

##### objektassociationer

- Specificera relationer en-till-många och många-till-många
- Kontrollera associationsprocessen

##### Avbilda arvrelationer på ett effektivt sätt

- Använda klassregler för arv
- Metoder för avbildning av klasser och databaser

#### Optimera dataåtkomsten

##### Använda Hibernate Query Language

##### (HQL)

- Välja och filtrera frågor
- Förbättra strukturen med namngivna frågor
- Utöka HQL med ursprungligt SQL
- Maximera Hibernates prestanda
- Accelerera dataåtkomst med Hibernate cache

##### Integrera Spring och Hibernate

- Använda mallen Spring Hibernate

- Konfigurera Hibernate-resurser i Spring