

Utforma Java Enterprise-applikationer med designmönster - 4 dagar

kurser 318

Du får lära dig att

- Bygga Java EE-arkitekturer med arbetsrutiner i enlighet med branschstandard
- Välja designmönster och identifiera deras användningsområden
- Skapa flexibla och kraftfulla konstruktioner för central affärslogik
- Utforma ett dataskikt som hanterar transaktioner och optimerar frågor
- Centralisera styrlogik i presentationsskiktet med Java EE-mönster
- Jämföra designen hos populära Java EE-ramverk och välja rätt till dina projekt

Sammanfattning

Det finns många företagsapplikationer i Java och det är en utmaning att designa ett effektivt Java-system. Java EE:s designmönster hjälper dig på traven genom att visa på bästa praxis, designidéer och beprövade metoder. På denna kurs får du erfarenhet av att utveckla skalbara och underhållsbara Java EE-applikationer. Du lär dig att tillämpa Java EE designmönster för att lösa vanligt förekommande konstruktionsproblem.

Vem bör delta

Alla som utformar eller utvecklar Java EE-applikationer. Eftersom tyngdpunkten ligger på programvarudesign, är det en förutsättning att man förstår Javakod i nivå med de kunskaper man får på kurs 471, "Javaprogrammering: omfattande introduktion". Praktisk erfarenhet av Java krävs och kunskaper om Java EE är en fördel.

Praktiska övningar

Kursen ger erfarenhet av att utveckla flexibla och robusta Java EE-applikationer. Bland övningarna märks att:

- Välja en kravbaserad topologi
- Skriva en enkelt distribuerad chat-applikation
- Konstruera en flexibel domänmodell
- Använda persistence-mekanismer på integrationsskiktet
- Konstruera detaljerade arbetsflöden för webbapplikationer
- Implementera en komplex, webbaserad Java EE-applikation
- Profilera prestandan hos JEE-applikationer

Utforma Java Enterprise-applikationer med designmönster - 4 dagar

kurser 318

Java EE och designmönster

Designprinciper och OO-designmönster

- Använda OO-designmönster som följer bästa praxis
- Bestämma lämpligt designmönster för krav
- Singleton
- Strategy
- Template
- Proxy
- Observer

Designmönster och Enterprise Java

- Analysera målen för Enterprise Java-applikationer
- Planera för distribuerade applikationer
- Kommunicera mellan JVM:er
- Implementera Remote Method Invocation

Bygga affärsskiktet

Modellera enheter och användningsfall

- Förverkliga en applikations domänmodell
- Business Object
- Application Service

Förhindra prestandaflaskhalsar

- Eliminera beroenden mellan skikt
- Service Facade
- Session Facade
- Business Delegate

Lokalisera objekt

- Singleton
- Factory
- Inversion of Control
- Service Locator

Implementera affärslogik med Session Beans

- Injicera tjänster till affärslogik med Session Beans
- Konversera med klienter med Stateful Session Beans

Kommunicera med meddelandetjänster

- Separera klientinteraktion med Java Message Service (JMS)
- Förenkla JMS
- Skicka och ta emot meddelanden med JMS
- Message Driven Beans

Hantera resurser i integrationsskiktet

Abstrahera dataskiktet

- Implementera effektiva Data Access Objects (DAO)
- Belysa svårigheter associerade med Object/Relational Mapping

- Analysera persistence-tekniker: Hibernate, JPA, EJB 3.0
- Optimer dataöverföring med Transfer Object Pattern

Web Services

- Exponera Beans som Web Services med metadata
- Web Service Broker-mönster

Hantera transaktioner effektivt

- Ta hänsyn till lokala och globala transaktionsbehov
- Välja optimistisk eller pessimistisk låsning

Strukturer presentationsskiktet

Skilja styr- och presentationslogiken åt

- Identifiera funktionen hos JSP och servlets
- Konstruera Model View Control (MVC)-arkitekturer

Planera och implementera komplexa arbetsflöden

- Front Controller
- Dispatcher View
- Service to Worker
- Hantera problem med dubbla formulär genom att lägga till en Synchronizer Token

Lokalisera disparat logik

- Förbättra underhållbarheten av algoritmer
- Intercepting Filter
- View Helper
- Composite View

Använda webbramverk

- Fastställa kriterier för utvärdering
- Hantera problem med dubbla formulär genom att lägga till en Synchronizer Token
- JSF
- Struts 2
- Spring MVC
- Google Web Toolkit (GWT)
- Tapestry
- Wicket

Använda lättviktsarkitekturer

Översikt över Spring Lightweight

Framework

- Designmönster med Inversion of Control (IoC)
- Konfigurering av IoC-containern Spring

Främja återanvändning av kod

- Aspektorienterad programmering
- Tillämpa återanvändning av komponenter med Spring
- Skicka e-post med Spring
- Använda mallar för dataåtkomst i Spring

Underhålla prestanda och skalbarhet

Konstruera för prestanda

- Distribuerade komponenter och prestanda
- Mäta körningsprestanda
- Optimera Java EE-applikationer
- Cachelagring
- Anslutningspool

Planera för skalbarhet

- Utvärdera konstruktionskompromisser i distribuerade arkitekturer
- Samla applikationer i kluster över servrar
- Hantera sessionstillstånd effektivt