

- REF: PD059
- NOMBRE: SPRING FRAMEWORK
- HORAS: 25

Objetivos:

Este curso esta formado por la integración de diferentes tecnologías y su aprovechamiento en el proceso de desarrollo e integración de un proyecto empresarial.

Con este curso el desarrollador reforzará y aprovechará mucho mejor el uso de tecnologías de Acceso a Datos, de Integración, entre otras.

En concreto se alcanzarán los siguientes objetivos:

Entender los conceptos "Inyección de dependencias" e "Inversión del Control"
Entender y aplicar los conceptos de AOP (Aspect Oriented Programming)
Diseñar aplicaciones con capas y contratos bien definidos
Aplicar las mejores practicas para acceso a datos usando JDBC y Hibernate
Aprender a construir aplicaciones web con SpringMVC
Conocer como exportar servicios usando diversas técnicas de Remoting (EJBs, WebServices, JMS)

Dirigido a:

A aquellos técnicos informáticos con interés en estas tecnologías.:

Requisitos Previos:

A aquellos técnicos informáticos con:

- *Amplio conocimiento del lenguaje Java*
- *Conocimiento de JDBC, deseable Hibernate*
- *Experiencia en desarrollo de aplicaciones web con Java*
- *Conocimiento de JEE (EJBs)*

Contenido:

CURSO DE SPRING FRAMEWORK

1. Introducción

- Vistazo rápido
- Arquitecturas posibles

2. Novedades en Spring 2.5

- Contenedor de IoC
- Configuración en XML mas sencilla
- Configuración basada en Anotaciones
- Auto detección de componentes en el classpath
- AOP
- Configuración en XML mas sencilla
- Nuevos pointcuts (beanname)
- Capa integración
- Configuración mas sencilla de transacciones
- JPA
- JMS Asíncrono
- JDBC
- Capa Web
- Convención sobre configuración en SpringMVC
- Framework para portlets
- Controllers basados en anotaciones
- Otras novedades
- Soporte a pruebas mejorado
- Soporte a Java 5 (Tiger)

3. El contenedor de IoC

- Introducción
- Conceptos básicos (Contenedores y beans)
- El contenedor
- Configuración
- Instanciación del contenedor
- Configuración basada en composición de XML
- Los beans
- Nombrado de beans
- Instancia de beans
- Uso del contenedor
- Dependencias
- Inyección de dependencias

- Inyección por constructor
- Inyección por mutador (setter)
- Ejemplos
- Dependencias y configuración en detalle
- Valores directos (primitivos, String, etc.)
- Referencias a otros beans (Colaboradores)
- Beans internos
- Colecciones
- Valores Nulos
- Atajos y otras opciones convenientes para configuración basada en XML
- Nombres de propiedades compuestas
- Uso de "depends on"
- Instanciación de beans perezosa
- Auto cableado de colaboradores
- Exclusiones
- Verificación de dependencias
- Inyección de métodos
- búsqueda
- reemplazo arbitrario de métodos
- Alcance de los beans (bean scope)
- Scope singleton
- Scope prototype
- Beans singleton con dependencias a beans prototype
- Otros alcances
- Configuración Web inicial
- Scope Request
- Scope Session
- Scope Global Session
- Scoped beans as dependencies
- Scopes personalizados
- Creación
- Uso
- Personalización de la naturaleza de un bean
- Ciclo de vida de retollamadas (Callbacks)
- Callbacks de inicialización
- Callbacks de liberación (destroy)
- Inicialización por defecto y métodos de liberación
- Combinación de mecanismos de ciclo de vida
- Apagado del contenedor de IoC en aplicaciones no Web
- BeanFactoryAware
- BeanNameAware
- Definición de herencia de beans

- Puntos de extensión del contenedor
- Personalización de beans usando BeanPostProcessors
- Personalización de configuración usando BeanFactoryPostProcessors
- Personalización de lógica de instanciación usando FactoryBeans
- El ApplicationContext
- BeanFactory o ApplicationContext
- Internacionalización
- Eventos
- Acceso a recursos de bajo nivel
- Instanciación de ApplicationContext en aplicaciones Web
- Código acoplado y el villano Singleton
- Configuración basada en anotaciones
- @Autowired
- Afinación de auto cableado basado en anotaciones con calificadores
- CustomAutowireConfigurer
- @Resource
- @PostConstruct and @PreDestroy
- Escaneo de Classpath para componentes administrados
- @Component and further stereotype annotations
- Auto detección de componentes
- Filtros para personalización del escaneo
- Nombrado de componentes auto detectados
- Alcance de componentes auto detectados

4. Recursos

- Introducción
- The Resource interface
- Implementaciones preconstruidas
- UriResource
- ClassPathResource
- FileSystemResource
- ServletContextResource
- InputStreamResource
- ByteArrayResource
- ResourceLoader
- Interface ResourceLoaderAware
- Recursos como dependencias
- Application contexts y rutas de Recursos
- Construcción de application contexts
- Construcción de ClassPathXmlApplicationContext atajos
- comodines para construir application context
- Antstyle Patterns
- The classpath*: prefix

- Notas
- Advertencias sobre FileSystemResource
- Validación, databinding, BeanWrapper y PropertyEditors
- Introducción
- Validación usando la interfaz Spring Validator
- Resolución de códigos de error a mensajes
- manipulación de Bean y el BeanWrapper
- alteración y obtención de propiedades simples y anidadas
- implementaciones preconstruidas de PropertyEditor
- Registro adicional de PropertyEditor

5. AOP con Spring

- Introducción
- Conceptos de AOP
- Capacidades y objetivos de Spring AOP
- AOP Proxies
- AOP Programática
- Soporte @AspectJ
- Habilitación del soporte @AspectJ
- Declaración de aspectos
- Declaración de pointcuts
- Supported Pointcut Designators
- Combining pointcut expressions
- Sharing common pointcut definitions
- Examples
- Declaración de Advices
- Before advice
- After returning advice
- After throwing advice
- After (finally) advice
- Around advice
- Advice parameters
- Advice ordering
- Introductions
- Aspect instantiation models
- Soporte de Schemabased AOP
- Declaración de aspectos
- Declaración de pointcuts
- Declaración de Advices
- Before advice
- After returning advice
- After throwing advice
- After (finally) advice

- Around advice
- Advice parameters
- Advice ordering
- Introductions
- Aspect instantiation models
- Advisors
- Uso de AspectJ con aplicaciones Spring

6. Pruebas

- Introducción
- Pruebas de unidad
- Objetos Mock
- JNDI
- Servlet API
- Portlet API
- Clases de soporte para pruebas de unidad

7. Acceso a Datos

- Administración de transacciones
- Introducción
- Motivaciones
- Abstracciones clave
- Sincronización de recursos con transacciones
- Alto nivel
- Bajo nivel
- TransactionAwareDataSourceProxy
- Administración de transacciones declarativa
- Entendimiento
- Rollback
- Configuración de semánticas diferentes para diferentes beans
- <tx:advice/> settings
- Using @Transactional
- @Transactional settings
- Propagación de transacciones
- Required
- RequiresNew
- Nested
- Transacciones programáticas
- TransactionTemplate
- PlatformTransactionManager

8. Soporte a DAO's

- Introducción
- Jerarquía de Excepciones
- Clases abstractas para soporte a DAO's

9. Acceso a datos con JDBC

- Introducción
- Elección del estilo
- Jerarquía de paquetes
- Uso de las clases Core de JDBC para un control básico de procesamiento JDBC y proceso de errores
- JdbcTemplate
- NamedParameterJdbcTemplate
- SimpleJdbcTemplate
- DataSource
- SQLExceptionTranslator
- Executing statements
- Running Queries
- Updating the database
- Retrieving autogenerated keys
- Control de las conexiones a la base de datos
- DataSourceUtils
- SmartDataSource
- AbstractDataSource
- SingleConnectionDataSource
- DriverManagerDataSource
- TransactionAwareDataSourceProxy
- DataSourceTransactionManager
- NativeJdbcExtractor
- Operaciones en Batch
- Batch operations with the JdbcTemplate
- Batch operations with the SimpleJdbcTemplate
- Simplificación de operaciones con SimpleJdbc
- Modelación de operaciones JDBC con objetos Java
- SqlQuery
- MappingSqlQuery
- SqlUpdate
- StoredProcedure
- SqlFunction
- Acceso a datos con ORM's
- Intro
- Hibernate
- Resource management
- SessionFactory setup in a Spring container
- The HibernateTemplate
- Implementing Springbased DAOs without callbacks
- Implementing DAOs based on plain Hibernate 3 API

- Programmatic transaction demarcation
- Declarative transaction demarcation
- Transaction management strategies
- Container resources versus local resources
- Spurious application server warnings when using Hibernate
- SpringMVC
- Intro
- Controllers
- AbstractController and WebContentGenerator
- Other simple controllers
- The MultiActionController
- Command controllers
- Handler mappings
- BeanNameUrlHandlerMapping
- SimpleUrlHandlerMapping
- Intercepting requests the HandlerInterceptor interface
- Views and resolving them
- Resolving views the ViewResolver interface
- Chaining ViewResolvers
- Redirecting to views
- RedirectView
- The redirect: prefix
- The forward: prefix
- Locales
- AcceptHeaderLocaleResolver
- CookieLocaleResolver
- SessionLocaleResolver
- LocaleChangeInterceptor
- Themes
- Fileupload
- Tag Library
- Handling exceptions
- Convention over configuration
- Configuración basada en anotaciones
- Integración
- Remoting
- WebServices
- XFire
- JMS
- EJB's
- Acceso
- Implementacion

- JMS
- JmsTemplate

