

Implementando uma Loja Virtual com Java EE 5 JPA e EJB

Por Rodrigo Urubatan Ferreira Jardim
<http://www.urubatan.com.br>

Sobre o Palestrante

- Rodrigo Urubatan - SCJP 1.4 e SCWCD
- Trabalha com arquitetura de sistemas J2EE e treinamento
- Já desenvolveu projetos utilizando as linguagens Delphi, C++, PHP, ASP, ColdFusion, Leather, Assembly, Perl, Ruby ...
- Trabalha com Java/J2EE a 4 anos e com desenvolvimento de sistemas a 9 anos
- Atualmente colabora com pequenas correções a alguns projetos Open Source como o GJ2, Lomboz e Veloclipse e faz parte da coordenação do RSJUG
- Já ministrou palestras em universidades (UCS, ULBRA, UNISC) e diversos eventos (Just Java, FISL, Seminário do RSJUG, Maratona 4 Java, Infosul) tutoriais para o RSJUG e já teve um artigo publicado na revista Mundo Java
- Atualmente trabalha como gerente de tecnologia e qualidade na Tech Office IT, ministra cursos e desenvolve sistemas para clientes.
- É o principal desenvolvedor do projeto Spring-Annotation

Problema proposto

- Criação de uma aplicação completa durante o mini curso
 - Loja de CDs
 - Cadastro de clientes
 - Pesquisa de satisfação

Ambiente de desenvolvimento

- NetBeans 6.0 M10
- Servidor de aplicações GlassFish
- JSF 1.2
- EJB 3.0
- JPA 1.0
- RichFaces + Ajax4JSF

O que é Java EE?

O que é Java EE

- Padrões pelo JCP (Java Community Process)
- Conjunto de bibliotecas
- Containers
 - Trasações
 - Remoting transparente
 - Alta disponibilidade
 - Distribuição de carga

Por que utilizar padrões?

Por que utilizar Padrões

- Uma pessoa tola, erra e não aprende
- Uma pessoa inteligente, erra e aprende com os próprios erros
- Uma pessoa sábia aprende com os erros dos outros

Começando pela persistência

- O que é o JPA
- Quais as vantagens do JPA
- Implementações
- Configuração via anotações
- Alteração de padrões via XML

Anotações

- @Entity
- @Id
- @GeneratedValue
- @Column
- Basic
- @Enumerated
- @Lob

Mapeando Relacionamentos

- @MappedSuperClass
- @Embedded
- @Embeddable
- @ManyToMany
- @ManyToOne
- @OneToMany
- @OneToOne
- @JoinTable
- @JoinColumns
- @JoinColumn
- @MapKey
-

Mapeando Objetos

- Quais os objetos que serão persistidos na aplicação?
- Quais os relacionamentos entre eles?
- Criando o persistence.xml
- Gerando o schema do banco de dados.

Um pouco de lógica

- @PostLoad
- @PrePersist
- @PostPersist
- @PreRemove
- @PostRemove
- @PostUpdate
- @PreUpdate

API do JPA

- Persistence
- EntityManagerFactory
- EntityManager
- Query

Testando a persistência

- Escrevendo testes unitários para as entidades
- Testando a lógica dos relacionamentos
- Uma configuração para runtime e outra para os testes

Regras de negócios

- O que são realmente os EJBs
- Tipos de EJB
 - Session Beans
 - Message Driven Beans
- Criando o primeiro EJB

Anotações

- @EJB
- @Local
- @LocalHome
- @Remote
- @RemoteHome
- @MessageDriven
- @Init
- @Remove
- @Stateless
- @Statefull
- @Timeout
- @TransactionAttribute
- @PostConstruct
- @PreDestroy

Comunicação entre EJBs

- Chamando um EJB de outro EJB
 - @EJB
- Como reutilizar código

Negócios + Persistência

- Escrevendo código de persistencia dentro dos EJBs
 - @PersistenceContext
 - @PersistenceUnit

Implementando os EJBs da aplicação

Testando a regra de negócios

- Testes unitários para os EJBs
 - O que testar com os testes unitários
- Testes de integração para os EJBs
 - Acessando EJBs de dentro dos testes unitários

Tudo Funcionando?

Agora que a aplicação já funciona!
(pelo menos o backend)

Dúvidas?

E agora?

- Opções para implementação de clientes para os EJBs.
- Cliente WEB
- Clientes console e desktop.
 - Suporte a JNLP nativo!