Exercícios de Java EE 7

Exercício 9 – Queries em JPA com JPQL e Criteria

Projeto: biblioteca-query

Objetivos: Partindo da aplicação desenvolvida no exercício anterior, criar métodos para realizar diversos tipos de consultas em exemplares, livros, autores, editoras, usuários e assuntos, usando as API Criteria e a linguagem JPQL.

Este exercício possui duas partes. A primeira consiste em criar consultas simples e estáticas, que serão exibidas em uma página. Na segunda parte as pesquisas devem ser incorporadas na aplicação para filtrar listas de livros, autores, editoras, etc.

a) A primeira parte (**/projeto-parte-1**) envolve dados estáticos. Veja na página queries-9.xhml e QueryBean.java as 6 consultas JPQL que devem ser implementadas. Implemente uma de cada vez e veja os resultados. O único arquivo que precisa ser alterado é **QueryBean.java**.

Para testar os resultados é preciso ter dados de teste. Se desejar use a infraestrutura existente para inserir livros, autores, editoras, ou use os links disponibilizados para configurar um ambiente de testes na página setup-testes.xhtml. Um bean (ConfigBean) e um EJB (TestServiceEJB) foram disponibilizados para esvaziar as tabelas e incluir dados de teste. Esvazie as tabelas e depois aperte o segundo botão da página setup-testes.xhtml para criar os dados de teste.

Observação: como não foi implementado mapeamento de cascade entre Livro, Autor, Editora e Exemplar, a remoção das entidades em setup-testes.xhtml precisa ser feita em ordem: primeiro Autor, depois Livro, Exemplar, Editora. Fica como **exercício** implementar o cascade-delete para evitar esse problema (aproveite e implemente também o cascade-persist e merge, e depois simplifique o código no TestServiceEJB.java onde indicado.)

b) Na segunda parte (**/projeto-parte-2**) devem ser implementadas pesquisas interativas.

1) Incluir um campo de texto na lista de autores para filtrar pelo nome.
2) Incluir um campo de texto na lista de editoras para filtrar pelo nome.
3) Incluir campos de texto na lista de assuntos para filtrar pela descrição e pelo código, além de um menu para selecionar qual índice usar. (resolvido)
4) Incluir campos de texto na lista de livros para filtrar pelo título, autor, editora e assunto, e um menu para filtrar pelo idioma. (resolvido)

A pesquisa deve alterar a lista de itens exibidos em tempo real (com delay de um segundo, para campos de texto, portanto é preciso que os componentes usem Ajax, renderizem novamente a tabela para cada alteração, e que os beans preservem o

estado dos dados durante as várias requisições que durar a pesquisa (use @ConversationScope)

Nesta versão estão resolvidos os exercícios 3 e 4 acima, que usam a API Criteria. Não deixe de analisar o código das classes e páginas envolvidas para entender o seu funcionamento, antes de tentar fazer os outros. As classes incluem EJBs (onde são implementados os queries JPQL ou Criteria – veja apostila de JPA) e managed beans (onde as consultas são delegadas para EJBs, e dados são mapeados a formulários e componentes JSF). As páginas são acopladas aos managed beans onde os componentes são mapeados. Veja mais informações sobre os componentes usados e o uso de <f:ajax> na apostila de JSF.

Para os exercícios desta seção (1 e 2) os arquivos a serem alterados são AutorBean, EditoraBean, AutorEJB, EditoraEJB, autores.xhtml, editoras.xhtml. Procure as indicações (comentários) em cada arquivo para mais detalhes, e use os exercícios resolvidos como modelo.

A solução destes exercícios está em **/solucao**. Se você achar que o exercício está muito simples e quiser um desafio maior, use como ponto de partida a solução do exercício anterior.