

ProductRepository.java

```
1 package model.repository;
2
3 import java.io.Serializable;
16
17 public class ProductRepository implements ServletContextListener {
18
19     @PersistenceUnit(unitName = "store-pu")
20     private EntityManagerFactory emf;
21
22     private EntityManager em;
23
24     public ProductRepository() {
25         emf = (EntityManagerFactory)
26 Persistence.createEntityManagerFactory("store-pu");
27         System.out.println("[SERVLET-TEST-INF0]: ProductRepository
28 Constructor");
29     }
30
31     public ProductRepository(EntityManagerFactory emf, EntityManager em) {
32         this.emf = emf;
33         this.em = em;
34         System.out.println("[SERVLET-TEST-INF0]: ProductRepository
35 Constructor");
36     }
37
38     @Override
39     public void contextDestroyed(ServletContextEvent sce) {
40         if (emf.isOpen()) {
41             emf.close();
42         }
43     }
44
45     @Override
46     public void contextInitialized(ServletContextEvent sce) {
47         ServletContext context = sce.getServletContext();
48         context.setAttribute("productRepository", this);
49         System.out.println("[SERVLET-TEST-INF0]: Contexto inicializado!
50 ProductRepository");
51     }
52
53     public final EntityManager entityManager() {
54         if (em == null || !em.isOpen()) {
55             em = emf.createEntityManager();
56         }
57         return em;
58     }
59
60     public void persistOrMerge(Serializable entity, Serializable id) {
61         em = entityManager();
62         if (entity == null) {
63             throw new IllegalArgumentException("entity");
64         }
65     }
```

ProductRepository.java

```
62     try {
63         em.getTransaction().begin();
64         if (id == null) {
65             em.persist(entity);
66         } else {
67             em.merge(entity);
68         }
69         em.getTransaction().commit();
70     } finally {
71         em.close();
72     }
73 }
74
75 public Product newProduct() {
76     Product p = new Product();
77     persistOrMerge(p, p.getId());
78     return p;
79 }
80
81 public Product findProductById(Long id) {
82     em = entityManager();
83     return em.find(Product.class, id);
84 }
85
86 public List findAllProducts() {
87     em = entityManager();
88     Query q = em.createQuery("select p from Product p");
89     return q.getResultList();
90 }
91
92 public List findAllProductsAvailable() {
93     em = entityManager();
94     Query q = em.createQuery("select p from Product p where
95 p.productOrder = NULL");
96     return q.getResultList();
97 }
98
99 public List findAllProductsAvailableByNearestDate() {
100     em = entityManager();
101     Query q = em.createQuery("select p from Product p where
102 p.productOrder = NULL ORDER BY p.productInsertDate DESC");
103     return q.getResultList();
104 }
105
106
107 public List findAllProductsByCategory(String category) {
108     em = entityManager();
109     Query q = em.createQuery("select p from Product p where
110 p.productOrder = NULL AND p.productCategory.categoryName=:cat");
111     q.setParameter("cat", category);
112     return q.getResultList();
113 }
```

ProductRepository.java

```
112     }  
113  
114 }
```