11

Mãos à obra: Ensinando o CDI a criar Entity Managers

Estamos pedindo em diversas partes do projeto para o CDI injetar para nós um `EntityManager`, só que temos um problema: O CDI tenta dar um `new EntityManager()`, quando pedimos a ele para criar um novo, mas isto não pode ser feito, já que o `EntityManager` é uma `interface` e não pode ser instanciada!

```
EntityManager em = new EntityManager(); //não funciona!
```

Pare resolver isso, precisamos ensinar ao CDI como criar corretamente um `EntityManager`. E nós já sabemos como fazer isso! Dentro da classe `JPAUtil` já existe o método `getEntityManager`, que nós dá um EntityManager prontinho.

```
public class JPAUtil {

    private static EntityManagerFactory emf = Persistence
            .createEntityManagerFactory("livraria");

    //método que sabe criar EntityManager
    public EntityManager getEntityManager() {
        return emf.createEntityManager();
    }

    public void close(EntityManager em) {
        em.close();
    }
}
```

Para dizer ao CDI que ele deve utilizar este método quando criar um `EntityManager`, temos que anotá-lo com uma anotação específica, você se lembra qual é ? É a `@Produces`.

Além de ensiná-lo a criar o EntityManager, precisamos anotar o método `getEntityManager` também com a anotação `@RequestScoped`, para dizer que devemos criar um EntityManager a cada requisição.

Só falta um pequeno detalhe: o nosso `DAO` genérico estava responsável por fechar o `EntityManager`. Se você conferir lá, vai ver que encontramos diversos `em.close()`. Mas como estamos utilizando um EntityManager por requisição, devemos deixar essa responsabilidade para o CDI:

```
public void close(@Disposes EntityManager em) {
        em.close();
}
```