05

# Mãos a obra: Specializes

Vamos fazer com que o comportamento para gerenciar transação possa ser sobrescrito caso exista essa necessidade.

```
public interface Transacionado extends Serializable{
    Object executaComTransacao( InvocationContext context );
}
```

Altere o interceptor para usar a interface `Transacionado` .

```
@Interceptor
@Transacional
public GerenciadorDeTransacao implements Serializable {

    private Transacionado transacionado;

    @Inject
    public GerenciadorDeTransacao(Transacionado transacionado){
        this.transacionado = transacionado;
    }


    @AroundInvoke
    public Object executaComTransacao(InvocationContext context){
        return    transacionado.executaComTransacao(context);
    }

}
```

Vamos criar uma implementação padrão para o `Transacionado` .

```
public class TransacionadoPadrao implements Transacionado {

    @Inject
    protected EntityManager em;

    @Override
    public Object executaComTransacao( InvocationContext context ) {
        em.getTransaction().begin();

        try {

            Object resultado = context.proceed();
            em.getTransaction().commit
            return resultado;
        } catch( Exception e){
            em.getTransaction().rollback()
            throw new RuntimeException(e);
```

```
        }
    }
}
```

Instale o `jar` no repositório local: Clique com o botão direito do mouse sobre o projeto `Alura-lib` e selecione `Run as > Maven Install`.

---

Vamos atualizar o projeto `Livraria`: Clique com o botão direito do mouse sobre o projeto `Livraria` e selecione `Maven > Update project...`.

Execute o projeto `Livraria` e verifique se tudo continua funcionando.

Após isso vamos criar uma nossa própria implementação para gerenciar transação.

```
@Specializes
public class MeuGerenciadorDeTransacao extends TransacionadoPadrao {


    public Object executaComTransacao( InvocationContext context ) {

        System.out.println("Antes de abrir a transação");
        em.getTransaction().begin();

        try {

            System.out.println("Antes de executar o método interceptado");
            Object resultado = context.proceed();

            System.out.println("Antes de efetuar o commit da transação");
            em.getTransaction().commit();

            return resultado;

        } catch (Exception e) {
            System.out.println("Antes de efetura o rollback");
            em.getTransaction().rollback();
            throw new RuntimeException(e);
        }
    }

}
```