\equiv



07 Cadeia de interceptadores

PRÓXIMA ATIVIDADE



ATIVIDADES 7 de 11

FÓRUM DO CURSO

VOLTAR PARA DASHBOARD





```
Dada a implementação seguinte:
```

```
//imports omitidos
public class LogInterceptador {
    @AroundInvoke
    public Object loga(InvocationContext ctx)
throws Exception {
        System.out.print("1");
        Object retorno = ctx.proceed();
        System.out.print("2");
        return retorno;
    }
}
```

```
@AroundInvoke
  public Object audita(InvocationContext ctx)
throws Exception {
        System.out.print("3");
        System.out.print("4");
        return null;
    }
}
```

public class AuditoriaInterceptador {

```
@Interceptors({LogInterceptador.class,
AuditoriaInterceptador.class})
@Stateless
public class ContaDao{

   public void salva(Conta conta) {
       System.out.print("5");
   }
}
COPIAR CÓDIGO
```



Assumindo que o código compila e foi publicado corretamente,

1 of 2 11/10/2021 19:51





64%

ATIVIDADES 7 de 11

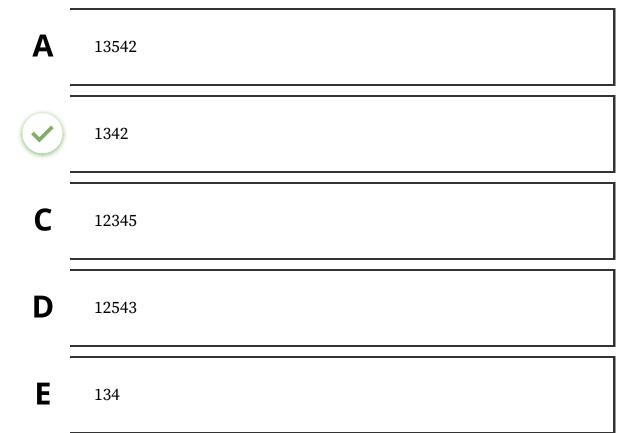
FÓRUM DO CURSO

VOLTAR PARA DASHBOARD





qual seria a saída ao chamar o método salva(..) da classe ContaDao ?



Nesse exemplo, a execução começa com o LogInterceptador e vai imprimir 1. Depois de chamar proceed() a execução vai para o próximo interceptador e entra no método audita(..) da classe AuditoriaInterceptador. Nesse método imprimimos 3 e 4. Repare que não há a chamada do método proceed(). Ou seja, nunca vamos chamar o próximo na cadeia. A execução não vai chegar a classe ContaDao, por isso retornamos para a classe LogInterceptador e imprimimos 2. O resultado final é 1342.

Como vimos, interceptadores podem ser chamados em uma cadeia. É bom pensar que um interceptador é como uma *casca* do Session Bean. Faz parte dele e podemos adicionar quantas cascas quisermos.

PRÓXIMA ATIVIDADE



9

2 of 2 11/10/2021 19:51