Interceptors

Example interceptors can be browsed at https://github.com/apache/tomee/tree/master/examples/interceptors

Help us document this example! Click the blue pencil icon in the upper right to edit this page.

ClassLevelInterceptorOne

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class ClassLevelInterceptorOne {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

ClassLevelInterceptorSuperClassOne

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class ClassLevelInterceptorSuperClassOne {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

ClassLevelInterceptorSuperClassTwo

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class ClassLevelInterceptorSuperClassTwo extends
SuperClassOfClassLevelInterceptor {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

ClassLevelInterceptorTwo

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class ClassLevelInterceptorTwo {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

DefaultInterceptorOne

```
package org.superbiz.interceptors;

import javax.annotation.PostConstruct;
import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class DefaultInterceptorOne {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }

    @PostConstruct
    protected void postConstructInterceptor(InvocationContext ic) throws Exception {
        Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

DefaultInterceptorTwo

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class DefaultInterceptorTwo {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

FullyIntercepted

```
package org.superbiz.interceptors;

import java.util.List;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public interface FullyIntercepted {

    List<String> businessMethod();

    List<String> methodWithDefaultInterceptorsExcluded();
}
```

FullyInterceptedBean

```
package org.superbiz.interceptors;

import javax.ejb.Local;
import javax.ejb.Stateless;
import javax.interceptor.AroundInvoke;
import javax.interceptor.Interceptors;
import javax.interceptor.InvocationContext;
import java.util.ArrayList;
import java.util.List;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
@Stateless
@Local
@Interceptors({ClassLevelInterceptorOne.class, ClassLevelInterceptorTwo.class})
public class FullyInterceptedBean extends FullyInterceptedSuperClass implements
FullyIntercepted {

    @Interceptors({MethodLevelInterceptorOne.class, MethodLevelInterceptorTwo.class})
    public List<String> businessMethod() {
        List<String> list = new ArrayList<String>();
        list.add("businessMethod");
        return list;
    }

    @Interceptors({MethodLevelInterceptorOne.class, MethodLevelInterceptorTwo.class})
    public List<String> methodWithDefaultInterceptorsExcluded() {
        List<String> list = new ArrayList<String>();
        list.add("methodWithDefaultInterceptorsExcluded");
        return list;
    }

    @AroundInvoke
    protected Object beanClassBusinessMethodInterceptor(InvocationContext ic) throws
Exception {
        return Utils.addClassSimpleName(ic, "beanClassBusinessMethodInterceptor");
    }
}
```

FullyInterceptedSuperClass

```
package org.superbiz.interceptors;

import javax.interceptor.Interceptors;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
@Interceptors({ClassLevelInterceptorSuperClassOne.class,
ClassLevelInterceptorSuperClassTwo.class})
public class FullyInterceptedSuperClass {
}
```

MethodLevelInterceptorOne

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class MethodLevelInterceptorOne {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

MethodLevelInterceptorOnlyIntf

```
package org.superbiz.interceptors;

import java.io.Serializable;
import java.util.List;

public interface MethodLevelInterceptorOnlyIntf<T extends Serializable> {
    public List<T> makePersistent(T entity);
}
```

MethodLevelInterceptorOnlyParent

```
package org.superbiz.interceptors;

import java.util.List;

public interface MethodLevelInterceptorOnlyParent extends
MethodLevelInterceptorOnlyIntf<String> {

    public List<String> makePersistent(String entity);
}
```

MethodLevelInterceptorOnlySLSBean

```
package org.superbiz.interceptors;

import javax.ejb.Local;
import javax.ejb.Stateless;
import javax.interceptor.Interceptors;
import java.util.ArrayList;
import java.util.List;

@Local(MethodLevelInterceptorOnlyParent.class)
@Stateless
public class MethodLevelInterceptorOnlySLSBean implements
MethodLevelInterceptorOnlyParent {

    @Interceptors(MethodLevelInterceptorOne.class)
    public List<String> makePersistent(String entity) {
        List<String> list = new ArrayList<String>();
        list.add("makePersistent");
        return list;
    }
}
```

MethodLevelInterceptorTwo

```
package org.superbiz.interceptors;

import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class MethodLevelInterceptorTwo {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

SecondStatelessInterceptedBean

```
package org.superbiz.interceptors;

import javax.ejb.Stateless;
import javax.interceptor.AroundInvoke;
import javax.interceptor.Interceptors;
import javax.interceptor.InvocationContext;
import java.util.ArrayList;
import java.util.List;

/**
 * @version $Rev: 808273 $ $Date: 2009-08-26 20:42:06 -0700 (Wed, 26 Aug 2009) $
 */
@Stateless
@Interceptors({ClassLevelInterceptorOne.class, ClassLevelInterceptorTwo.class})
public class SecondStatelessInterceptedBean implements SecondStatelessInterceptedLocal
{

    @Interceptors({MethodLevelInterceptorOne.class, MethodLevelInterceptorTwo.class})
    public List<String> methodWithDefaultInterceptorsExcluded() {
        List<String> list = new ArrayList<String>();
        list.add("methodWithDefaultInterceptorsExcluded");
        return list;
    }

    @AroundInvoke
    protected Object beanClassBusinessMethodInterceptor(InvocationContext ic) throws
Exception {
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

SecondStatelessInterceptedLocal

```
package org.superbiz.interceptors;

import java.util.List;

/**
 * @version $Rev: 808273 $ $Date: 2009-08-26 20:42:06 -0700 (Wed, 26 Aug 2009) $
 */
public interface SecondStatelessInterceptedLocal {
    List<String> methodWithDefaultInterceptorsExcluded();
}
```

SuperClassOfClassLevelInterceptor

```
package org.superbiz.interceptors;

import javax.annotation.PostConstruct;
import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

/**
 * @version $Rev: 607077 $ $Date: 2007-12-27 06:55:23 -0800 (Thu, 27 Dec 2007) $
 */
public class SuperClassOfClassLevelInterceptor {

    @AroundInvoke
    protected Object businessMethodInterceptor(InvocationContext ic) throws Exception
{
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }

    @PostConstruct
    protected void postConstructInterceptor(InvocationContext ic) throws Exception {
        Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

ThirdSLSBean

```
package org.superbiz.interceptors;

import javax.ejb.Stateless;
import javax.interceptor.AroundInvoke;
import javax.interceptor.ExcludeClassInterceptors;
import javax.interceptor.ExcludeDefaultInterceptors;
import javax.interceptor.Interceptors;
import javax.interceptor.InvocationContext;
import java.util.ArrayList;
import java.util.List;

/**
 * @version $Rev: 1090810 $ $Date: 2011-04-10 07:49:26 -0700 (Sun, 10 Apr 2011) $
 */
@Stateless
@Interceptors({ClassLevelInterceptorOne.class, ClassLevelInterceptorTwo.class})
@ExcludeDefaultInterceptors
public class ThirdSLSBean implements ThirdSLSBeanLocal {

    @Interceptors({MethodLevelInterceptorOne.class, MethodLevelInterceptorTwo.class})
    public List<String> businessMethod() {
        List<String> list = new ArrayList<String>();
        list.add("businessMethod");
        return list;
    }

    @Interceptors({MethodLevelInterceptorOne.class, MethodLevelInterceptorTwo.class})
    @ExcludeClassInterceptors
    public List<String> anotherBusinessMethod() {
        List<String> list = new ArrayList<String>();
        list.add("anotherBusinessMethod");
        return list;
    }


    @AroundInvoke
    protected Object beanClassBusinessMethodInterceptor(InvocationContext ic) throws
Exception {
        return Utils.addClassSimpleName(ic, this.getClass().getSimpleName());
    }
}
```

ThirdSLSBeanLocal

```
package org.superbiz.interceptors;

import java.util.List;

/**
 * @version $Rev: 607320 $ $Date: 2007-12-28 12:15:06 -0800 (Fri, 28 Dec 2007) $
 */
public interface ThirdSLSBeanLocal {
    List<String> businessMethod();

    List<String> anotherBusinessMethod();
}
```

Utils

```
package org.superbiz.interceptors;

import javax.interceptor.InvocationContext;
import java.util.ArrayList;
import java.util.List;

/**
 * @version $Rev: 808273 $ $Date: 2009-08-26 20:42:06 -0700 (Wed, 26 Aug 2009) $
 */
public class Utils {

    public static List<String> addClassSimpleName(InvocationContext ic, String
classSimpleName) throws Exception {
        List<String> list = new ArrayList<String>();
        list.add(classSimpleName);
        List<String> listOfStrings = (List<String>) ic.proceed();
        if (listOfStrings != null) {
            list.addAll(listOfStrings);
        }
        return list;
    }
}
```

ejb-jar.xml

```
<ejb-jar xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd"
         version="3.0">
  <interceptors>
    <interceptor>
      <interceptor-class>org.superbiz.interceptors.DefaultInterceptorOne</interceptor-
class>
    </interceptor>
    <interceptor>
      <interceptor-class>org.superbiz.interceptors.DefaultInterceptorTwo</interceptor-
class>
    </interceptor>
  </interceptors>
  <assembly-descriptor>
    <interceptor-binding>
      <ejb-name>*</ejb-name>
      <interceptor-class>org.superbiz.interceptors.DefaultInterceptorOne</interceptor-
class>
    </interceptor-binding>
    <interceptor-binding>
      <ejb-name>*</ejb-name>
      <interceptor-class>org.superbiz.interceptors.DefaultInterceptorTwo</interceptor-
class>
    </interceptor-binding>
    <interceptor-binding>
      <ejb-name>FullyInterceptedBean</ejb-name>
      <exclude-default-interceptors>true</exclude-default-interceptors>
      <method>
        <method-name>methodWithDefaultInterceptorsExcluded</method-name>
      </method>
    </interceptor-binding>
    <interceptor-binding>
      <ejb-name>SecondStatelessInterceptedBean</ejb-name>
      <exclude-default-interceptors>true</exclude-default-interceptors>
    </interceptor-binding>
    <interceptor-binding>
      <ejb-name>MethodLevelInterceptorOnlySLSBean</ejb-name>
      <exclude-default-interceptors>true</exclude-default-interceptors>
    </interceptor-binding>
  </assembly-descriptor>
</ejb-jar>
```

FullyInterceptedTest

```
package org.superbiz.interceptors;
```

```
import junit.framework.TestCase;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import javax.naming.Context;
import javax.naming.InitialContext;
import java.util.ArrayList;
import java.util.List;
import java.util.Properties;

/**
 * @version $Rev: 1090810 $ $Date: 2011-04-10 07:49:26 -0700 (Sun, 10 Apr 2011) $
 */
public class FullyInterceptedTest extends TestCase {

    private InitialContext initCtx;

    @Before
    public void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        properties.setProperty("openejb.deployments.classpath.include",
".*interceptors/target/classes.*");

        initCtx = new InitialContext(properties);
    }

    @Test
    public void testBusinessMethod() throws Exception {

        FullyIntercepted fullyIntercepted = (FullyIntercepted) initCtx.lookup(
"FullyInterceptedBeanLocal");

        assert fullyIntercepted != null;

        List<String> expected = new ArrayList<String>();
        expected.add("DefaultInterceptorOne");
        expected.add("DefaultInterceptorTwo");
        expected.add("ClassLevelInterceptorSuperClassOne");
        expected.add("ClassLevelInterceptorSuperClassTwo");
        expected.add("ClassLevelInterceptorOne");
        expected.add("ClassLevelInterceptorTwo");
        expected.add("MethodLevelInterceptorOne");
        expected.add("MethodLevelInterceptorTwo");
        expected.add("beanClassBusinessMethodInterceptor");
        expected.add("businessMethod");

        List<String> actual = fullyIntercepted.businessMethod();
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
```

```
actual;
    }

    @Test
    public void testMethodWithDefaultInterceptorsExcluded() throws Exception {

        FullyIntercepted fullyIntercepted = (FullyIntercepted) initCtx.lookup(
"FullyInterceptedBeanLocal");

        assert fullyIntercepted != null;

        List<String> expected = new ArrayList<String>();
        expected.add("ClassLevelInterceptorSuperClassOne");
        expected.add("ClassLevelInterceptorSuperClassTwo");
        expected.add("ClassLevelInterceptorOne");
        expected.add("ClassLevelInterceptorTwo");
        expected.add("MethodLevelInterceptorOne");
        expected.add("MethodLevelInterceptorTwo");
        expected.add("beanClassBusinessMethodInterceptor");
        expected.add("methodWithDefaultInterceptorsExcluded");

        List<String> actual = fullyIntercepted.methodWithDefaultInterceptorsExcluded(
);
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
actual;
    }

    @After
    public void tearDown() throws Exception {
        initCtx.close();
    }
}
```

MethodLevelInterceptorOnlyTest

```
package org.superbiz.interceptors;

import junit.framework.TestCase;
import org.junit.Before;
import org.junit.Test;

import javax.naming.Context;
import javax.naming.InitialContext;
import java.util.ArrayList;
import java.util.List;
import java.util.Properties;

/**
 * @version $Rev: 895825 $ $Date: 2010-01-04 15:35:22 -0800 (Mon, 04 Jan 2010) $
 */
public class MethodLevelInterceptorOnlyTest extends TestCase {
    private InitialContext initCtx;

    @Before
    public void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        properties.setProperty("openejb.deployments.classpath.include",
".*interceptors/target/classes.*");

        initCtx = new InitialContext(properties);
    }

    @Test
    public void testInterceptedGenerifiedBusinessIntfMethod() throws Exception {
        MethodLevelInterceptorOnlyParent bean = (MethodLevelInterceptorOnlyParent)
initCtx.lookup("MethodLevelInterceptorOnlySLSBeanLocal");

        assert bean != null;

        List<String> expected = new ArrayList<String>();
        expected.add("MethodLevelInterceptorOne");
        expected.add("makePersistent");

        List<String> actual = bean.makePersistent(null);
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
actual;
    }
}
```

SecondStatelessInterceptedTest

```
package org.superbiz.interceptors;

import junit.framework.TestCase;
import org.junit.Before;
import org.junit.Test;

import javax.naming.Context;
import javax.naming.InitialContext;
import java.util.ArrayList;
import java.util.List;
import java.util.Properties;

/**
 * @version $Rev: 1090810 $ $Date: 2011-04-10 07:49:26 -0700 (Sun, 10 Apr 2011) $
 */
public class SecondStatelessInterceptedTest extends TestCase {

    private InitialContext initCtx;

    @Before
    public void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        properties.setProperty("openejb.deployments.classpath.include",
".*interceptors/target/classes.*");

        initCtx = new InitialContext(properties);
    }

    @Test
    public void testMethodWithDefaultInterceptorsExcluded() throws Exception {
        SecondStatelessInterceptedLocal bean =
                (SecondStatelessInterceptedLocal) initCtx.lookup(
"SecondStatelessInterceptedBeanLocal");

        assert bean != null;

        List<String> expected = new ArrayList<String>();
        expected.add("ClassLevelInterceptorOne");
        expected.add("ClassLevelInterceptorTwo");
        expected.add("MethodLevelInterceptorOne");
        expected.add("MethodLevelInterceptorTwo");
        expected.add("SecondStatelessInterceptedBean");
        expected.add("methodWithDefaultInterceptorsExcluded");

        List<String> actual = bean.methodWithDefaultInterceptorsExcluded();
```

```
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
actual;
    }
}
```

ThirdSLSBeanTest

```
package org.superbiz.interceptors;

import junit.framework.TestCase;
import org.junit.Before;
import org.junit.Test;

import javax.naming.Context;
import javax.naming.InitialContext;
import java.util.ArrayList;
import java.util.List;
import java.util.Properties;

/**
 * @version $Rev: 1090810 $ $Date: 2011-04-10 07:49:26 -0700 (Sun, 10 Apr 2011) $
 */
public class ThirdSLSBeanTest extends TestCase {
    private InitialContext initCtx;

    @Before
    public void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        properties.setProperty("openejb.deployments.classpath.include",
".*interceptors/target/classes.*");

        initCtx = new InitialContext(properties);
    }

    @Test
    public void testMethodWithDefaultInterceptorsExcluded() throws Exception {
        ThirdSLSBeanLocal bean = (ThirdSLSBeanLocal) initCtx.lookup("
ThirdSLSBeanLocal");

        assert bean != null;

        List<String> expected = new ArrayList<String>();
        expected.add("ClassLevelInterceptorOne");
        expected.add("ClassLevelInterceptorTwo");
        expected.add("MethodLevelInterceptorOne");
        expected.add("MethodLevelInterceptorTwo");
        expected.add("ThirdSLSBean");
```

```
        expected.add("businessMethod");

        List<String> actual = bean.businessMethod();
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
actual;
    }

    @Test
    public void testMethodWithDefaultAndClassInterceptorsExcluded() throws Exception {
        ThirdSLSBeanLocal bean = (ThirdSLSBeanLocal) initCtx.lookup("
ThirdSLSBeanLocal");

        assert bean != null;

        List<String> expected = new ArrayList<String>();
        expected.add("MethodLevelInterceptorOne");
        expected.add("MethodLevelInterceptorTwo");
        expected.add("ThirdSLSBean");
        expected.add("anotherBusinessMethod");

        List<String> actual = bean.anotherBusinessMethod();
        assert expected.equals(actual) : "Expected " + expected + ", but got " +
actual;
    }
}
```

Running

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.interceptors.FullyInterceptedTest
Apache OpenEJB 4.0.0-beta-1    build: 20111002-04:06
http://tomee.apache.org/
INFO - openejb.home = /Users/dblevins/examples/interceptors
INFO - openejb.base = /Users/dblevins/examples/interceptors
INFO - Configuring Service(id=Default Security Service, type=SecurityService,
provider-id=Default Security Service)
INFO - Configuring Service(id=Default Transaction Manager, type=TransactionManager,
provider-id=Default Transaction Manager)
INFO - Using 'openejb.deployments.classpath.include=.*interceptors/target/classes.*'
INFO - Found EjbModule in classpath:
/Users/dblevins/examples/interceptors/target/classes
INFO - Beginning load: /Users/dblevins/examples/interceptors/target/classes
INFO - Configuring enterprise application:
/Users/dblevins/examples/interceptors/classpath.ear
INFO - Configuring Service(id=Default Stateless Container, type=Container, provider-
id=Default Stateless Container)
```

```
INFO - Auto-creating a container for bean FullyInterceptedBean:
Container(type=STATELESS, id=Default Stateless Container)
INFO - Enterprise application "/Users/dblevins/examples/interceptors/classpath.ear"
loaded.
INFO - Assembling app: /Users/dblevins/examples/interceptors/classpath.ear
INFO - Jndi(name=FullyInterceptedBeanLocal) --> Ejb(deployment-
id=FullyInterceptedBean)
INFO -
Jndi(name=global/classpath.ear/interceptors/FullyInterceptedBean!org.superbiz.intercep
tors.FullyIntercepted) --> Ejb(deployment-id=FullyInterceptedBean)
INFO - Jndi(name=global/classpath.ear/interceptors/FullyInterceptedBean) -->
Ejb(deployment-id=FullyInterceptedBean)
INFO - Jndi(name=ThirdSLSBeanLocal) --> Ejb(deployment-id=ThirdSLSBean)
INFO -
Jndi(name=global/classpath.ear/interceptors/ThirdSLSBean!org.superbiz.interceptors.Thi
rdSLSBeanLocal) --> Ejb(deployment-id=ThirdSLSBean)
INFO - Jndi(name=global/classpath.ear/interceptors/ThirdSLSBean) --> Ejb(deployment-
id=ThirdSLSBean)
INFO - Jndi(name=SecondStatelessInterceptedBeanLocal) --> Ejb(deployment-
id=SecondStatelessInterceptedBean)
INFO -
Jndi(name=global/classpath.ear/interceptors/SecondStatelessInterceptedBean!org.superbi
z.interceptors.SecondStatelessInterceptedLocal) --> Ejb(deployment-
id=SecondStatelessInterceptedBean)
INFO - Jndi(name=global/classpath.ear/interceptors/SecondStatelessInterceptedBean) -->
Ejb(deployment-id=SecondStatelessInterceptedBean)
INFO - Jndi(name=MethodLevelInterceptorOnlySLSBeanLocal) --> Ejb(deployment-
id=MethodLevelInterceptorOnlySLSBean)
INFO -
Jndi(name=global/classpath.ear/interceptors/MethodLevelInterceptorOnlySLSBean!org.supe
rbiz.interceptors.MethodLevelInterceptorOnlyParent) --> Ejb(deployment-
id=MethodLevelInterceptorOnlySLSBean)
INFO - Jndi(name=global/classpath.ear/interceptors/MethodLevelInterceptorOnlySLSBean)
--> Ejb(deployment-id=MethodLevelInterceptorOnlySLSBean)
INFO - Created Ejb(deployment-id=ThirdSLSBean, ejb-name=ThirdSLSBean,
container=Default Stateless Container)
INFO - Created Ejb(deployment-id=SecondStatelessInterceptedBean, ejb-
name=SecondStatelessInterceptedBean, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=FullyInterceptedBean, ejb-name=FullyInterceptedBean,
container=Default Stateless Container)
INFO - Created Ejb(deployment-id=MethodLevelInterceptorOnlySLSBean, ejb-
name=MethodLevelInterceptorOnlySLSBean, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=ThirdSLSBean, ejb-name=ThirdSLSBean,
container=Default Stateless Container)
INFO - Started Ejb(deployment-id=SecondStatelessInterceptedBean, ejb-
name=SecondStatelessInterceptedBean, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=FullyInterceptedBean, ejb-name=FullyInterceptedBean,
container=Default Stateless Container)
INFO - Started Ejb(deployment-id=MethodLevelInterceptorOnlySLSBean, ejb-
name=MethodLevelInterceptorOnlySLSBean, container=Default Stateless Container)
INFO - Deployed Application(path=/Users/dblevins/examples/interceptors/classpath.ear)
```

```
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.564 sec
Running org.superbiz.interceptors.MethodLevelInterceptorOnlyTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.004 sec
Running org.superbiz.interceptors.SecondStatelessInterceptedTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.003 sec
Running org.superbiz.interceptors.ThirdSLSBeanTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.004 sec

Results :

Tests run: 6, Failures: 0, Errors: 0, Skipped: 0
```