Simple Stateless with callback methods

Example simple-stateless-callbacks can be browsed at https://github.com/apache/tomee/tree/master/examples/simple-stateless-callbacks

This example shows how to create a stateless session bean that uses the @PostConstruct, @PreDestroy and @AroundInvoke annotations.

CalculatorBean

```java
package org.superbiz.stateless.basic;

import javax.annotation.PostConstruct;
import javax.annotation.PreDestroy;
import javax.ejb.Stateless;
import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;

@Stateless
public class CalculatorBean {

    @PostConstruct
    public void postConstruct() {
        ExecutionChannel.getInstance().notifyObservers("postConstruct");
    }

    @PreDestroy
    public void preDestroy() {
        ExecutionChannel.getInstance().notifyObservers("preDestroy");
    }

    @AroundInvoke
    public Object intercept(InvocationContext ctx) throws Exception {
        ExecutionChannel.getInstance().notifyObservers(ctx.getMethod().getName());
        return ctx.proceed();
    }

    public int add(int a, int b) {
        return a + b;
    }

    public int subtract(int a, int b) {
        return a - b;
    }

    public int multiply(int a, int b) {
        return a * b;
    }

    public int divide(int a, int b) {
        return a / b;
    }

    public int remainder(int a, int b) {
        return a % b;
    }

}
```

ExecutionChannel

```java
package org.superbiz.counter;

import java.util.ArrayList;
import java.util.List;

public class ExecutionChannel {
    private static final ExecutionChannel INSTANCE = new ExecutionChannel();

    private final List<ExecutionObserver> observers = new ArrayList<ExecutionObserver
>();

    public static ExecutionChannel getInstance() {
        return INSTANCE;
    }

    public void addObserver(ExecutionObserver observer) {
        this.observers.add(observer);
    }

    public void notifyObservers(Object value) {
        for (ExecutionObserver observer : this.observers) {
            observer.onExecution(value);
        }
    }
}
```

ExecutionObserver

```java
package org.superbiz.counter;

public interface ExecutionObserver {

    void onExecution(Object value);

}
```

CalculatorTest

```java
package org.superbiz.stateless.basic;

import org.junit.Assert;
import org.junit.Test;
import javax.ejb.embeddable.EJBContainer;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import java.util.ArrayList;
import java.util.List;
import java.util.Properties;

public class CalculatorTest implements ExecutionObserver {
    private static final String JNDI = "java:global/simple-stateless-
callbacks/CalculatorBean";

    private List<Object> received = new ArrayList<Object>();

    public Context getContext() throws NamingException {
        final Properties p = new Properties();
        p.put(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        return new InitialContext(p);

    }

    @Test
    public void test() throws Exception {
        ExecutionChannel.getInstance().addObserver(this);

        final EJBContainer container = EJBContainer.createEJBContainer();

        {
            final CalculatorBean calculator = (CalculatorBean) getContext().lookup
(JNDI);

            Assert.assertEquals(10, calculator.add(4, 6));

            //the bean is constructed only when it is used for the first time
            Assert.assertEquals("postConstruct", this.received.remove(0));
            Assert.assertEquals("add", this.received.remove(0));

            Assert.assertEquals(-2, calculator.subtract(4, 6));
            Assert.assertEquals("subtract", this.received.remove(0));

            Assert.assertEquals(24, calculator.multiply(4, 6));
            Assert.assertEquals("multiply", this.received.remove(0));

            Assert.assertEquals(2, calculator.divide(12, 6));
            Assert.assertEquals("divide", this.received.remove(0));

            Assert.assertEquals(4, calculator.remainder(46, 6));
            Assert.assertEquals("remainder", this.received.remove(0));
        }
        {
            final CalculatorBean calculator = (CalculatorBean) getContext().lookup
(JNDI);

            Assert.assertEquals(10, calculator.add(4, 6));
            Assert.assertEquals("add", this.received.remove(0));

        }

        container.close();
        Assert.assertEquals("preDestroy", this.received.remove(0));
        Assert.assertTrue(this.received.isEmpty());
    }

    @Override
    public void onExecution(Object value) {
        System.out.println("Test step -> " + value);
        this.received.add(value);
    }
}
```

Running

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.stateless.basic.CalculatorTest
INFO -
********************************************************************************
INFO - OpenEJB http://tomee.apache.org/
INFO - Startup: Sat Jul 21 09:23:38 EDT 2012
INFO - Copyright 1999-2012 (C) Apache OpenEJB Project, All Rights Reserved.
INFO - Version: 4.1.0
INFO - Build date: 20120721
INFO - Build time: 04:06
INFO -
********************************************************************************
INFO - openejb.home = /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateless-callbacks
INFO - openejb.base = /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateless-callbacks
INFO - Created new singletonService
org.apache.openejb.cdi.ThreadSingletonServiceImpl@527736bd
INFO - Succeeded in installing singleton service
INFO - Using 'javax.ejb.embeddable.EJBContainer=true'
INFO - Cannot find the configuration file [conf/openejb.xml].  Will attempt to create
one for the beans deployed.
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 - Creating TransactionManager(id=Default Transaction Manager)
INFO - Creating SecurityService(id=Default Security Service)
INFO - Beginning load: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateless-callbacks/target/classes
INFO - Configuring enterprise application:
/home/boto/dev/ws/openejb_trunk/openejb/examples/simple-stateless-callbacks
INFO - Auto-deploying ejb CalculatorBean: EjbDeployment(deployment-id=CalculatorBean)
INFO - Configuring Service(id=Default Stateless Container, type=Container, provider-
id=Default Stateless Container)
INFO - Auto-creating a container for bean CalculatorBean: Container(type=STATELESS,
id=Default Stateless Container)
INFO - Creating Container(id=Default Stateless Container)
INFO - Configuring Service(id=Default Managed Container, type=Container, provider-
id=Default Managed Container)
INFO - Auto-creating a container for bean org.superbiz.stateless.basic.CalculatorTest:
Container(type=MANAGED, id=Default Managed Container)
INFO - Creating Container(id=Default Managed Container)
INFO - Using directory /tmp for stateful session passivation
INFO - Enterprise application
"/home/boto/dev/ws/openejb_trunk/openejb/examples/simple-stateless-callbacks" loaded.
INFO - Assembling app: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateless-callbacks
INFO - Jndi(name="java:global/simple-stateless-
callbacks/CalculatorBean!org.superbiz.stateless.basic.CalculatorBean")
INFO - Jndi(name="java:global/simple-stateless-callbacks/CalculatorBean")
INFO - Existing thread singleton service in SystemInstance()
org.apache.openejb.cdi.ThreadSingletonServiceImpl@527736bd
INFO - OpenWebBeans Container is starting...
INFO - Adding OpenWebBeansPlugin : [CdiPlugin]
INFO - All injection points are validated successfully.
INFO - OpenWebBeans Container has started, it took 111 ms.
INFO - Created Ejb(deployment-id=CalculatorBean, ejb-name=CalculatorBean,
container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorBean, ejb-name=CalculatorBean,
container=Default Stateless Container)
INFO - Deployed
Application(path=/home/boto/dev/ws/openejb_trunk/openejb/examples/simple-stateless-
callbacks)
Test step -> postConstruct
Test step -> add
Test step -> subtract
Test step -> multiply
Test step -> divide
Test step -> remainder
Test step -> add
INFO - Undeploying app: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateless-callbacks
Test step -> preDestroy
```

```
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.884 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
```