Simple Stateful with callback methods

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

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

CallbackCounter

```
package org.superbiz.counter;

import javax.annotation.PostConstruct;
import javax.annotation.PreDestroy;
import javax.ejb.PostActivate;
import javax.ejb.PrePassivate;
import javax.ejb.Stateful;
import javax.ejb.StatefulTimeout;
import javax.interceptor.AroundInvoke;
import javax.interceptor.InvocationContext;
import java.io.Serializable;
import java.util.concurrent.TimeUnit;

@Stateful
@StatefulTimeout(value = 1, unit = TimeUnit.SECONDS)
public class CallbackCounter implements Serializable {

    private int count = 0;

    @PrePassivate
    public void prePassivate() {
        ExecutionChannel.getInstance().notifyObservers("prePassivate");
    }

    @PostActivate
    public void postActivate() {
        ExecutionChannel.getInstance().notifyObservers("postActivate");
    }

    @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 count() {
        return count;
    }

    public int increment() {
        return ++count;
    }

    public int reset() {
        return (count = 0);
    }
}
```

ExecutionChannel

```
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

```
package org.superbiz.counter;

public interface ExecutionObserver {

    void onExecution(Object value);

}
```

CounterCallbacksTest

```
package org.superbiz.counter;

import junit.framework.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.*;

public class CounterCallbacksTest implements ExecutionObserver {
    private static 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 {
        final Map<String, Object> p = new HashMap<String, Object>();
        p.put("MySTATEFUL", "new://Container?type=STATEFUL");
        p.put("MySTATEFUL.Capacity", "2"); //How many instances of Stateful beans can
our server hold in memory?
        p.put("MySTATEFUL.Frequency", "1"); //Interval in seconds between checks
        p.put("MySTATEFUL.BulkPassivate", "0"); //No bulkPassivate - just passivate
entities whenever it is needed
        final EJBContainer container = EJBContainer.createEJBContainer(p);

        //this is going to track the execution
        ExecutionChannel.getInstance().addObserver(this);
```

```
        {
            final Context context = getContext();

            CallbackCounter counterA = (CallbackCounter) context.lookup(
"java:global/simple-stateful-callbacks/CallbackCounter");
            Assert.assertNotNull(counterA);
            Assert.assertEquals("postConstruct", received.remove(0));

            Assert.assertEquals(0, counterA.count());
            Assert.assertEquals("count", received.remove(0));

            Assert.assertEquals(1, counterA.increment());
            Assert.assertEquals("increment", received.remove(0));

            Assert.assertEquals(0, counterA.reset());
            Assert.assertEquals("reset", received.remove(0));

            Assert.assertEquals(1, counterA.increment());
            Assert.assertEquals("increment", received.remove(0));

            System.out.println("Waiting 2 seconds...");
            Thread.sleep(2000);

            Assert.assertEquals("preDestroy", received.remove(0));

            try {
                counterA.increment();
                Assert.fail("The ejb is not supposed to be there.");
            } catch (javax.ejb.NoSuchEJBException e) {
                //excepted
            }

            context.close();
        }

        {
            final Context context = getContext();

            CallbackCounter counterA = (CallbackCounter) context.lookup(
"java:global/simple-stateful-callbacks/CallbackCounter");
            Assert.assertEquals("postConstruct", received.remove(0));

            Assert.assertEquals(1, counterA.increment());
            Assert.assertEquals("increment", received.remove(0));

            ((CallbackCounter) context.lookup("java:global/simple-stateful-
callbacks/CallbackCounter")).count();
            Assert.assertEquals("postConstruct", received.remove(0));
            Assert.assertEquals("count", received.remove(0));
```

```
            ((CallbackCounter) context.lookup("java:global/simple-stateful-
callbacks/CallbackCounter")).count();
            Assert.assertEquals("postConstruct", received.remove(0));
            Assert.assertEquals("count", received.remove(0));

            System.out.println("Waiting 2 seconds...");
            Thread.sleep(2000);
            Assert.assertEquals("prePassivate", received.remove(0));

            context.close();
        }
        container.close();

        Assert.assertEquals("preDestroy", received.remove(0));
        Assert.assertEquals("preDestroy", received.remove(0));

        Assert.assertTrue(received.toString(), received.isEmpty());
    }

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

Running

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.counter.CounterCallbacksTest
INFO -
********************************************************************************
INFO - OpenEJB http://tomee.apache.org/
INFO - Startup: Sat Jul 21 08:18:28 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-
stateful-callbacks
INFO - openejb.base = /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateful-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 - Configuring Service(id=MySTATEFUL, type=Container, provider-id=Default Stateful
Container)
INFO - Creating TransactionManager(id=Default Transaction Manager)
INFO - Creating SecurityService(id=Default Security Service)
INFO - Creating Container(id=MySTATEFUL)
INFO - Using directory /tmp for stateful session passivation
INFO - Beginning load: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateful-callbacks/target/classes
INFO - Configuring enterprise application:
/home/boto/dev/ws/openejb_trunk/openejb/examples/simple-stateful-callbacks
INFO - Auto-deploying ejb CallbackCounter: EjbDeployment(deployment-
id=CallbackCounter)
INFO - Configuring Service(id=Default Managed Container, type=Container, provider-
id=Default Managed Container)
INFO - Auto-creating a container for bean org.superbiz.counter.CounterCallbacksTest:
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-stateful-callbacks" loaded.
INFO - Assembling app: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateful-callbacks
INFO - Jndi(name="java:global/simple-stateful-
callbacks/CallbackCounter!org.superbiz.counter.CallbackCounter")
INFO - Jndi(name="java:global/simple-stateful-callbacks/CallbackCounter")
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 225 ms.
INFO - Created Ejb(deployment-id=CallbackCounter, ejb-name=CallbackCounter,
container=MySTATEFUL)
INFO - Started Ejb(deployment-id=CallbackCounter, ejb-name=CallbackCounter,
container=MySTATEFUL)
INFO - Deployed
Application(path=/home/boto/dev/ws/openejb_trunk/openejb/examples/simple-stateful-
callbacks)
Test step -> postConstruct
Test step -> count
Test step -> increment
Test step -> reset
Test step -> increment
```

```
Waiting 2 seconds...
Test step -> preDestroy
INFO - Removing the timed-out stateful session bean instance
583c10bfdbd326ba:57f94a9b:138a9798adf:-8000
INFO - Activation failed: file not found /tmp/583c10bfdbd326ba=57f94a9b=138a9798adf=-
8000
Test step -> postConstruct
Test step -> increment
Test step -> postConstruct
Test step -> count
Test step -> postConstruct
Test step -> count
Waiting 2 seconds...
Test step -> prePassivate
INFO - Passivating to file /tmp/583c10bfdbd326ba=57f94a9b=138a9798adf=-7fff
Test step -> preDestroy
INFO - Removing the timed-out stateful session bean instance
583c10bfdbd326ba:57f94a9b:138a9798adf:-7ffe
Test step -> preDestroy
INFO - Removing the timed-out stateful session bean instance
583c10bfdbd326ba:57f94a9b:138a9798adf:-7ffd
INFO - Undeploying app: /home/boto/dev/ws/openejb_trunk/openejb/examples/simple-
stateful-callbacks
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 7.487 sec

Results :

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

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 15.803s
[INFO] Finished at: Sat Jul 21 08:18:35 EDT 2012
[INFO] Final Memory: 11M/247M
[INFO] ------------------------------------------------------------------------
```