CDI @ApplicationScoped

Example cdi-application-scope can be browsed at https://github.com/apache/tomee/tree/master/examples/cdi-application-scope

This example show the use of @ApplicationScoped annotation for injected objects. An object which is defined as @ApplicationScoped is created once for the duration of the application.

Example

This example depicts a similar scenario to cdi-request-scope. A restaurant guest orders a soup from the waiter. The waiter then delivers the soup back to the guest. Another guest can order the same soup that was ordered by the previous client - this is where the application scope is used.

Waiter

The Waiter session bean receives a request from the test class via the orderSoup() method and sets the name for the soup field. The orderWhatTheOtherGuyHad() method returns the name of the soup field.

```
@Stateless
public class Waiter {

    @Inject
    public Soup soup;

    public String orderSoup(String name){
        soup.setName(name);
        return soup.getName();
    }

    public String orderWhatTheOtherGuyHad() {
        String name = soup.getName();
        return name;
    }
}
```

Soup

The Soup class is an injectable POJO, defined as @ApplicationScoped. This means that an instance will be created only once for the duration of the whole application. Try changing the @ApplicationScoped annotation to @RequestScoped and see what happens.

```
@ApplicationScoped
public class Soup {

    private String name = "Soup of the day";

    @PostConstruct
    public void afterCreate() {
        System.out.println("Soup created");
    }

    public String getName() {
        return name;
    }

    public void setName(String name){
        this.name = name;
    }
}
```

Test Case

This is the entry class for this example. First a soup is ordered via `orderSoup()` method. This initiates the `soup` field. Next, `orderWhatTheOtherGuyHad()` method returns the soup from the application context.

```
public class RestaurantTest {

    private static String TOMATO_SOUP = "Tomato Soup";
    private EJBContainer container;

    @EJB
    private Waiter joe;

    @Before
    public void startContainer() throws Exception {
        container = EJBContainer.createEJBContainer();
        container.getContext().bind("inject", this);
    }

    @Test
    public void orderSoup(){
        String someSoup = joe.orderSoup(TOMATO_SOUP);
        assertEquals(TOMATO_SOUP, someSoup);

        String sameSoup = joe.orderWhatTheOtherGuyHad();
        assertEquals(TOMATO_SOUP, sameSoup);
    }

    @After
    public void closeContainer() throws Exception {
        container.close();
    }
}
```

Running

In the output you can see that there is just one `Soup` instance created - one for the whole application.

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.cdi.applicationscope.RestaurantTest
Apache OpenEJB 7.0.0-SNAPSHOT    build: 20111224-11:09
http://tomee.apache.org/
INFO - openejb.home = C:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-
application-scope
INFO - openejb.base = C:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-
application-scope
INFO - Using 'javax.ejb.embeddable.EJBContainer=true'
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 - Found EjbModule in classpath:
c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-application-
scope\target\classes
INFO - Beginning load: c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-
application-scope\target\classes
INFO - Configuring enterprise application:
c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-application-scope
INFO - Configuring Service(id=Default Managed Container, type=Container, provider-
id=Default Managed Container)
INFO - Auto-creating a container for bean cdi-application-scope.Comp:
Container(type=MANAGED, id=Default Managed Container)
INFO - Configuring Service(id=Default Stateless Container, type=Container, provider-
id=Default Stateless Container)
INFO - Auto-creating a container for bean Waiter: Container(type=STATELESS, id=Default
Stateless Container)
INFO - Enterprise application
"c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-application-scope" loaded.
INFO - Assembling app: c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-
application-scope
INFO - Jndi(name="java:global/cdi-application-
scope/Waiter!org.superbiz.cdi.applicationscope.Waiter")
INFO - Jndi(name="java:global/cdi-application-scope/Waiter")
INFO - Created Ejb(deployment-id=Waiter, ejb-name=Waiter, container=Default Stateless
Container)
INFO - Started Ejb(deployment-id=Waiter, ejb-name=Waiter, container=Default Stateless
Container)
INFO - Deployed
Application(path=c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-application-
scope)
Soup created
INFO - Undeploying app: c:\Users\Daniel\workspaces\openejb\openejb\examples\cdi-
application-scope
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.42 sec

Results :

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