simple-cdi-interceptor

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

Simple CDI Interceptor

Let's write a simple application that would allow us to book tickets for a movie show. As with all applications, logging is one cross-cutting concern that we have.

(Relevant snippets are inlined but you can check out the complete code, from the links provided)

How do we mark which methods are to be intercepted ? Wouldn't it be handy to annotate a method like

```
@Log
public void aMethod(){...}

Let's create an  annotation that would "mark" a method for interception.

@InterceptorBinding
@Target({ TYPE, METHOD })
@Retention(RUNTIME)
public @interface Log {
}
```

Sure, you haven't missed the @InterceptorBinding annotation above ! Now that our custom annotation is created, lets attach it (or to use a better term for it, "bind it") to an interceptor.

So here's our logging interceptor. An @AroundInvoke method and we are almost done.

```
@Interceptor
@Log  //binding the interceptor here. now any method annotated with @Log would be
intercepted by logMethodEntry
public class LoggingInterceptor {
    @AroundInvoke
    public Object logMethodEntry(InvocationContext ctx) throws Exception {
        System.out.println("Entering method: " + ctx.getMethod().getName());
        //or logger.info statement
        return ctx.proceed();
    }
}
```

Now the @Log annotation we created is bound to this interceptor.

That done, let's annotate at class-level or method-level and have fun intercepting !

```
@Log
@Stateful
public class BookShow implements Serializable {
    private static final long serialVersionUID = 6350400892234496909L;
    public List<String> getMoviesList() {
        List<String> moviesAvailable = new ArrayList<String>();
        moviesAvailable.add("12 Angry Men");
        moviesAvailable.add("Kings speech");
        return moviesAvailable;
    }
    public Integer getDiscountedPrice(int ticketPrice) {
        return ticketPrice - 50;
    }
    // assume more methods are present
}
```

The `@Log` annotation applied at class level denotes that all the methods should be intercepted with `LoggingInterceptor`.

Before we say "all done" there's one last thing we are left with ! To enable the interceptors !

Lets quickly put up a [beans.xml file]

```
<beans>
  <interceptors>
    <class>org.superbiz.cdi.bookshow.interceptors.LoggingInterceptor
    </class>
  </interceptors>
</beans>
```

```
in META-INF
```

Those lines in beans.xml not only "enable" the interceptors, but also define the "order of execution" of the interceptors. But we'll see that in another example on multiple-cdi-interceptors.

Fire up the test, and we should see a 'Entering method: getMoviesList' printed in the console.

Tests

```
Apache OpenEJB 4.0.0-beta-2    build: 20111103-01:00
http://tomee.apache.org/
INFO - openejb.home = /media/fthree/Workspace/open4/openejb/examples/cdi-simple-
interceptors
INFO - openejb.base = /media/fthree/Workspace/open4/openejb/examples/cdi-simple-
interceptors
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:
/media/fthree/Workspace/open4/openejb/examples/cdi-simple-interceptors/target/classes
INFO - Beginning load: /media/fthree/Workspace/open4/openejb/examples/cdi-simple-
interceptors/target/classes
INFO - Configuring enterprise application:
/media/fthree/Workspace/open4/openejb/examples/cdi-simple-interceptors
INFO - Configuring Service(id=Default Managed Container, type=Container, provider-
id=Default Managed Container)
INFO - Auto-creating a container for bean cdi-simple-interceptors.Comp:
Container(type=MANAGED, id=Default Managed Container)
INFO - Configuring Service(id=Default Stateful Container, type=Container, provider-
id=Default Stateful Container)
INFO - Auto-creating a container for bean BookShow: Container(type=STATEFUL,
id=Default Stateful Container)
INFO - Enterprise application "/media/fthree/Workspace/open4/openejb/examples/cdi-
simple-interceptors" loaded.
INFO - Assembling app: /media/fthree/Workspace/open4/openejb/examples/cdi-simple-
interceptors
INFO - Jndi(name="java:global/cdi-simple-
interceptors/BookShow!org.superbiz.cdi.bookshow.beans.BookShow")
INFO - Jndi(name="java:global/cdi-simple-interceptors/BookShow")
INFO - Created Ejb(deployment-id=BookShow, ejb-name=BookShow, container=Default
Stateful Container)
INFO - Started Ejb(deployment-id=BookShow, ejb-name=BookShow, container=Default
Stateful Container)
INFO - Deployed Application(path=/media/fthree/Workspace/open4/openejb/examples/cdi-
simple-interceptors)
Entering method: getMoviesList
```