Guidelines, Tips and Tricks for Using Java EE 5

Inderjeet Singh, Google, Inc.

Roger Kitain, Sun Microsystems, Inc.

Mahesh Kannan, Sun Microsystems, Inc.

Marina Vatkina, Sun Microsystems, Inc.

TS-4593

Goal of the Talk

Java™ Platform, Enterprise Edition (Java EE) 5 Guidelines, Tips and Puzzlers for developing enterprise applications

Agenda

- Web Tier—Inderjeet Singh and Roger Kitain
  - Annotations, Servlet Filters, NIO
  - Deployment Descriptor Gems
  - Unified Expression Language Evaluation
  - Ajax In The Web Tier
- EJB™ Architecture 3.0—Mahesh Kannan
  - Dependency Injection subtleties
  - Interceptors
- Java Persistence API—Marina Vatkina

Web Tier Puzzlers and Tips

What Is Wrong With the Highlighted Code?

```
public class MyServlet extends HttpServlet {
    @PersistenceContext (unitName="CatalogPU")
    private EntityManager em;

    @Resource UserTransaction utx;

    // ......
}
```

- @PersistenceContext annotation does not inject thread-safe object
  - Servlet and even Managed beans (except request scoped) are multi-threaded environments
- @Resource UserTransaction injects thread-safe object

How to Write It Correctly?

- Use request scoped JavaServer Faces Managed Beans
- Use Thread-safe variants
  - Use class-level annotations with J.N.D.I. API lookups
  - Use other annotations
  - Use Old fashioned J.N.D.I. API lookups
- Lessons for the API Designers
  - Unsafe annotations should be a compile time error
  - Make annotations thread-safe

Using Class Scoped Annotations With J.N.D.I. API Lookups

```
@PersistenceContext(name="EM" unitName = "CatalogPU")
public class MyServlet extends HttpServlet {
  public EntityManager getEntityManager() {
    Context ic = new InitialContext();
    return (EntityManager) ic.lookup("java:comp/env/EM");
}
```

- @PersistenceContext annotation is only declaring the environment dependency
- One lookup per method call => Thread-safe

Use Another Annotation

```
public class MyServlet extends HttpServlet {
    @PersistenceUnit (unitName="CatalogPu")
    private EntityManagerFactory emf;
  public EntityManager getEntityManager() {
    return emf.createEntityManager();
}
```

- @PersistenceUnit annotation injects Entity Manager Factory which is a thread-safe object
- Method create a new entity manager per call → thread-safe

What Is Wrong With This Code?

```
public class TransactionsFilter implements Filter {
  @UserTransaction utx;
  public void doFilter(ServletRequest request,
                       ServletResponse response) throws ... {
    try {
      utx.begin();
      chain.doFilter(request, response);
      utx.commit();
    } catch (Exception e) {
      utx.rollback(e);
    }
  }
}
```

- Filter's doFilter() may run in a different thread than servlet's service() method!
- In practice, it usually works
- Lesson to API Designers: Make it work

How to Use NIO With Servlet API?

- Servlet API is NOT NIO-based
- Use the wrapper method in Channels
- The example below sends the contents of an image file on the output stream of Servlet response

```
   // Copy imageFile to servlet output
   FileInputStream fis = new FileInputStream(imageFile);
   FielChannel in = fis.getChannel();
   WriteableByteChannel out =
Channels.newChannel(response.getOutputStream());
   in.transferTo(0, in.size(), out);
```

Anything Wrong With This Entry?

```
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <url-pattern>/status/compressed/*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <url-pattern>/photos/compressed/*</url-pattern>
</filter-mapping
```

- No—but you can do this instead

```
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <url-pattern>/status/compressed/*</url-pattern>
  <url-pattern>/photos/compressed/*</url-pattern>
</filter-mapping>
```

Anything Wrong With This Entry?

```
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <servlet-name>servletA</servlet-name>
</filter-mapping>
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <servlet-name>servletB</servlet-name>
</filter-mapping>
```

- No—but you can do this instead

```
<filter-mapping>
  <filter-name>compressResponse</filter-name>
  <servlet-name>*</servlet-name>
</filter-mapping>
```

Unified Expression Language Evaluation

- Unification of JavaServer Pages™ (JSP) and JavaServer Faces API expression languages
- JSP API—Immediate expression evaluation
  - When page is rendered "${cart.total}"
- JavaServer Faces API—Deferred expression evaluation
  - Immediate evaluation during page rendering
  - During postback value propagated to bean "#{cart.total}"
- "#" syntax is now reserved in JSP 2.1

What's Wrong With This Scenario?

- This JSP API is executed in JSP 2.1

```
<%@ taglib uri="/sample.tld" prefix="sample" %>
<html>
......
<sample:hello location="apt #{2"/>
......
```

- Tell the container to allow "#{" as string literals by doing any of the following
  - Escape the "#{" characters as follows: \#{
  - Add *deferred-syntax-allowed-as-literal subelement to the jsp-property-group element and set it to true*
  - *Use deferredSyntaxAllowedAsLiteral in page directive*

```
<%@page...deferredSyntaxAllowedAsLiteral="true" %>
```

How Can I Cause a "Postback" From JavaScript™ Technology?

- "javax.faces.ViewState"
  - The state of the current view in JavaServer Faces
  - Standardized in JavaServer Faces 1.2
  - Indicates a "postback" to JavaServer Faces

```
<input type="hidden" name="javax.faces.ViewState"
  id="javax.faces.ViewState" value="....."/>
```

- Include 'javax.faces.ViewState" when posting from a JavaServer Faces page
- JavaServer Faces Ajax frameworks send "javax.faces.ViewState" as post data parameter

Localizing JavaServer Faces Applications

- Anything wrong with this page?

```
...
<f:view>
    <f:loadBundle basename="myapp.Resources var="bundle"/>
    <h:form id="form">
        <h:outputText value="#{bundle.text}"/>
    </h:form>
</f:view>
```

- No, but:
  - You need to define the f:loadBundle tag on every page that needs localized information
  - Less efficient—bundle is loaded on every request

Localizing JavaServer Faces Applications

- A better way:
  - Define application scope resource bundle in application's configuration file:

```
<resource-bundle>
  <var>bundle</var>
  <base-name>myapp.Resources</base-name>
</resource-bundle>
```

  - Access it in page:

```
<f:view>
   <h:form id="form">
      <h:outputText value="{bundle.text}"/>
   </h:form>
</f:view>
```

Mixing HTML and JavaServer Faces…

- You can still do this:

```
...
<h:outputText value="Hello there"/>
<f:verbatim>
   Fred
</f:verbatim>
...
```

- But why?

```
...
<h:outputText value="Hello there"/>
Fred
...
```

Managing Beans…

- Consider this scenario:

```
...
<managed-bean-name>OrdersBean</managed-bean-name>
<managed-bean-class>ex.Orders</managed-bean-class>
<managed-bean-scope>request</managed-bean-scope>
...
```

```
public class Orders {
  ...
     public List getOrders() {
          if (this.orders != null) {
               return this.orders;
          ...
  }
```

Managing Beans…

- We can specify what bean methods get executed at bean creation time:

```
public class Orders {
 ...
    @PostConstruct
    public void init() {
        ...
    }
    public List getOrders() {
        return this.orders;
    ...
}
```

- Init() method gets called once

Managing Beans…

- And we can also specify “cleanup” methods that get executed just before the bean goes out of scope:

```
public class Orders {
 ...
    @PreDestroy
    public void cleanUp() {
        ...
    }
    ...
}
```

EJB Component Architecture 3.0 Puzzlers and Tips

Dependency Injection in Java Platform, Enterprise Edition (Java EE Platform) v.5

Dependency injection

- Also called "IoC" (Inversion of Control)
- Supported only inside "managed" components
  - Servlets, EJB components, Interceptors, etc.
- Obviates the need for ServiceLocator pattern
- @Resource, @EJB, @PersistenceContext, etc.

What Does the lookup() Inside the doPost() Return?

```
public class MyServlet... {
        @Resource(name="jdbc/MyDS")
        private DataSource ds;

        public void doPost(...) {
          InitialContext ic = new InitialContext();
          DataSource ds2 = (DataSource)
                  ic.lookup("java:comp/env/jdbc/MyDS");
        }
}
```

@Resource not only declares ds as target of injection

It also declares a dependency in the component env

So ic.lookup("java:comp/env/jdbc/MyDS") works!!

What Does the test() Method Return?

```
@Stateful public ShoppingCartBean
     implements ShoppingCart {...}

@Stateless public HelloBean
     implements Hello {...}

@Stateless
public class MyTestBean implements MyTest {
     @EJB private ShoppingCart  cart1;
     @EJB private ShoppingCart  cart2;

     @EJB private Hello hello1;
     @EJB private Hello hello2;

     public boolean test() {
            return cart1.equals(cart2)
                && hello1.equals(hello2);
     }
}
```

// Returns false
// Returns true

What if test() needs 10 Shopping Carts?

Using @EJB at Class Level

- ```
  @EJB(name="ejb/MySC",   beanInterface=ShoppingCart.class)
  @Stateless public class MyTestBean implements MyTest {
        @Resource SessionContext ctx;

        public void stressTest() {
           InitialContext ic = new InitialContext();
           for (...) {
                 ic.lookup("java:comp/env/ejb/MySC");
                 ctx.lookup("ejb/MySC"); // preferred way
           }
        }
  }
  ```
- @EJB at class level declares a dependency
- Tip: EJBContext.lookup() is always relative to "java:comp/env"
  ```

What Is Wrong With the Following Util Class?

```
@Stateless public class MyTestBean implements MyTest {
        @Resource(name="jdbc/MyDS")
        private DataSource ds;

        public void stressTest() {
            (new Util()).doSomething();
        }
    }

    public class Util {
        @Resource DataSource ds2;
        public void doSomething() {

          Connection connection = ds2.getConnection();
          //do something with connection
        }
    }
```

What Is Wrong With the Following Util Class?

```
@Stateless public class MyTestBean implements MyTest {
      @Resource(name="jdbc/MyDS")
      private DataSource ds;

      public void stressTest() {
          (new Util()).doSomething();
      }
}

public class Util {  // Not a Managed component
  @Resource DataSource ds2;  // Will NOT work
  public void doSomething() {
    InitialContext ic = new InitialContext();
    // The following will work
ds2 = (DataSource)
ic.lookup("java:comp/env/jdbc/MyDS");
//do something with connection
      }
}
```

Tip: ***Helper classes are not managed classes. But they can access "current" component environment***

Interceptors in Java EE Platform v.5

EJB 3.0 Component Architecture Interceptor

- **Used to intercept business and Lifecycle events**
- **Container Managed**
  - **Has the same life cycle as the bean instance with which they are associated**
- **Interceptors are still application code**
- **Written by application developer**

What Is Wrong With the reverse() Method?

```
@Stateless
   public class StringBean implements StringUtil {
       public String reverse(String str) {
          String rStr = "";
          for (int i=str.length()-1; i>=0; i--) {
              rStr += str.charAt(i);
          }
          return rStr;
       }
   }
```

What Is Wrong With the reverse() Method?

```
@Stateless @Interceptor(Validator.class)
  public class StringBean implements StringUtil {
    public String reverse(String str) {
      String rStr = "";
        for (int i=str.length()-1; i>=0; i--) {
          rStr += str.charAt(i);
        }
        return rStr;
      }
    }
  public class Validator {
    @AroundInvoke
   private Object validate(InvocationContext ctx) {
      Method m = ctx.getMethod();
      if (m.getName().equals("reverse")) {
        String str = (String) ctx.getParameters()[0];
        if ((str == null) || (str.length() < 2)) {
          return str;
        }  else { return ctx.proceed(); }
      }
    }
  }
```

Using Multiple Interceptors

```
@Stateful
   public class MyShoppingCartBean ... {
        public void addToCart(Item item) {...}
   }

   public Validator {
        @AroundInvoke
        private Object validate(InvocationContext ctx) {
            //use ctx.getParameters() to do some validations
           return ctx.proceed();
        }
   }

   public Profiler {
        @AroundInvoke private Object profile(....) {
      return ctx.proceed();
        }
   }
```

Using Multiple Interceptors

```
@Stateful
   @Interceptors({Validator.class, Profiler.class})
   public class MyShoppingCartBean ... {
        public void addToCart(Item item) {...}
   }

   public Validator {
        @AroundInvoke
        private Object validate(InvocationContext ctx) {
            //use ctx.getParamters() to do some validations
           return ctx.proceed();//Calls Profiler.profile()
        }
   }

   public Profiler {
        @AroundInvoke
        private Object profile(InvocationContext ctx) {
               return ctx.proceed();      //Calls bean method
        }
   }
```

More on Interceptors

```
public OrderValidator { //an Interceptor
   @Resource SessionContext ssnCtx;
   @EJB private OrderFacade order;

   @AroundInvoke private Object validate(...) {
       try {
         if (invalidOrder(invCtx)) {
           ssnCtx.setRollbackOnly();
           throw new EJBException("Invalid order");
         } else {
           return invCtx.proceed();
         }
       } finally {
         if (! ssnCtx.getRollbackOnly()) {
           order.doSomething(...);
         }
      }
   }
```

How Do I Intercept Only A Particular Method?

```
@Stateful public class ShoppingCartBean
      implements ShoppingCart {

      public int getCartSize() {...}

      public void checkout() {....}

  }
```

getCartSize() does NOT need to be intercepted

How Do I Intercept Only A Particular Method?

```
@Stateful public class ShoppingCartBean
      implements ShoppingCart {

      public int getCartSize() {...}

      public void checkout() {....}

  }
```

```
  @Stateful public class ShoppingCartBean
      implements ShoppingCart {

      public int getCartSize() {...}

      @Interceptors(OrderValidator.class)
      public void checkout() {....}
  }
```

How to Intercept All Methods in All Beans in a ejb-jar File?

```
@Stateful public class ShoppingCartBean
      implements ShoppingCart {
      public int getCartSize() {...}
      public void checkout() {....}
  }
  @Stateless public class StringUtilBean
      implements StringUtil {
      public String reverse(String str) {...}
      public boolean isPalindrome(String str) {....}
  }
```

How to Intercept All Methods in All Beans in a ejb-jar file?

```
@Stateful public class ShoppingCartBean
      implements ShoppingCart {
      public int getCartSize() {...}
      public void checkout() {....}
  }
  @Stateless public class StringUtilBean
      implements StringUtil {
      public String reverse(String str) {...}
      public boolean isPalindrome(String str) {....}
  }
```

```
<interceptor-binding>
    <ejb-name>*</ejb-name>
    <interceptor-class>MethodProfiler</interceptor-
  class>
    <interceptor-class>MethodLogger</interceptor-
  class>
  </interceptor-binding>
```

Java Persistence API Puzzlers and Tips

What Is the Simplest persistence.xml for Java EE Platform Environment?

- This is all that you need to use the defaults

```
<persistence
xmlns="http://java.sun.com/xml/ns/persistence"
        version="1.0">
    <persistence-unit name="MyPU"/>
</persistence>
```

- Defaults provided by the container
  - Provider class
  - Default datasource J.N.D.I. API name
- Specification defined defaults
  - Transaction Type–JTA
- Dynamically constructed list of managed classes

Java Platform, Standard Edition (Java SE) or Non Java EE v.5 Container?

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
    version="1.0">
    <persistence-unit name="MyPU">
        <provider>com.acme.PersistenceProvider</provider>
        <class>Entity1</class>
        <class>Entity2</class>
        <properties>
            <!-- JDBC access properties -->
        </properties>
    </persistence-unit>
    </persistence>
```

Packaging Entities in a WAR file?

- Package entities into a jar

  foo.war/

  WEB-INF/lib/entities.jar

  META-INF/persistence.xml

  pkg/Entity1.class

  pkg/Entity2.class

- **Package entities under WEB-INF/classes**

  foo.war/

  WEB-INF/classes/

  META-INF/persistence.xml

  pkg/Entity1.class

  pkg/Entity2.class

Packaging Entities in an EAR File?

- Package entities as a library to share between EJB and WEB components

  a.ear/

  ejb1.jar

  ejb2.jar

  foo.war

  lib/entities.jar

- **Packaging into various components results in multiple (independent) versions of a Persistent Unit**

What Is Wrong With this Entity?

```
@Entity public class Entity1 {
    @Id
    private int key;
    private Entity2 entity2;

    @OneToOne
    public Entity2 getEntity2() {return entity2;}
}
```

Can NOT mix access types
Use single access type per entity hierarchy

How Do I Specify Access Type for My Entity Using Annotations?

- Field-based

```
@Id
private int key;

@OneToOne
private Entity2 entity2;
```

- Property-based

```
@Id
public int getKey() {return key;}

@OneToOne
public Entity2 getEntity2() {return entity2;}
```

How to Specify Access Type for Entity Using XML Overrides?

- Global PU setting

```
<entity-mappings ... >
    <persistence-unit-metadata>
        <persistence-unit-defaults>
            <access>PROPERTY</access>
        </persistence-unit-defaults>
    </persistence-unit-metadata>
</entity-mappings>
```

- **Single entity**

```
<entity-mappings ... >
    <entity name="MyEntity" class="Entity1"
access="FIELD">
    </entity>
</entity-mappings>
```

What Should I Know About JTA Entity Manager?

- Transactions are controlled through JTA
- Container-Managed Entity Manager
  - Always JTA
  - Container registers EM for transaction synchronization
- Application-Managed Entity Manager
  - Application is responsible for causing EM to join the transaction

How to Use Container-Managed JTA Entity Manager in an EJB component?

- Inject and use it:

```
@Stateless
public class MyBean implements MyInterface {

    @PersistenceContext (unitName="MyPU")
    private EntityManager em;

    public void createItem(...) {
    Item item = new Item();
    em.persist(item);
    }
    ...
```

How to Use Container-managed JTA Entity Manager in a Servlet?

- Look it up in the same method:

```
@PersistenceContext(name="EM" unitName = "MyPU")
 public class MyServlet extends HttpServlet {
public void doGet(...) {

        EntityManager em = (EntityManager)
              ic.lookup("java:comp/env/EM");

        Item item = new Item();
        utx.begin();
        em.persist(item);
        utx.commit();
        }
        ...
```

How About Application-managed JTA Entity Manager?

- Usually use it in a Servlet

```
public class MyServlet extends HttpServlet {
    public void doGet(...) {
        EntityManager em = emf.createEntityManager(); // 1
        Item item = new Item();
        utx.begin(); // 2
        em.joinTransaction(); // Not needed if 1 is after 2
        em.persist(item);
        utx.commit();
        }
        ...
}
```

What Should I Know About RESOURCE_LOCAL Entity Manager?

- Only Application-Managed
- Uses Entity Transactions

```
public class MyServlet extends HttpServlet {
    public void doGet(...) {
        EntityManager em = emf.createEntityManager();
        Item item = new Item();
        em.getTransaction().begin();
        em.persist(item);
        em.getTransaction().commit();
    }
    ...
}
```

Using Generics with Java Persistence

```
public interface Query() {
  public List getResultList();

  // NO Generic version available
}
```

- In most cases, you know the type of the result
  - So, use Generics

```
Query q = em.createQuery("SELECT i FROM Item i");
List<Item> items = q.getResultList();
```

For More Information

Other resources

- https://glassfish.dev.java.net/
- https://jsf-extensions.dev.java.net/
- https://ajax.dev.java.net/
- http://forums.java.net/jive/forum.jspa?forumID=56

Q&A

Inderjeet Singh

Marina Vatkina

Mahesh Kannan

Roger Kitain

Guidelines, Tips and Tricks for Using Java EE 5

Inderjeet Singh, Google, Inc.

Roger Kitain, Sun Microsystems, Inc.

Mahesh Kannan, Sun Microsystems, Inc.

Marina Vatkina, Sun Microsystems, Inc.

TS-4593