Xdoclet Introduction

Ross Sponholtz

rsponholtz@connectria.com

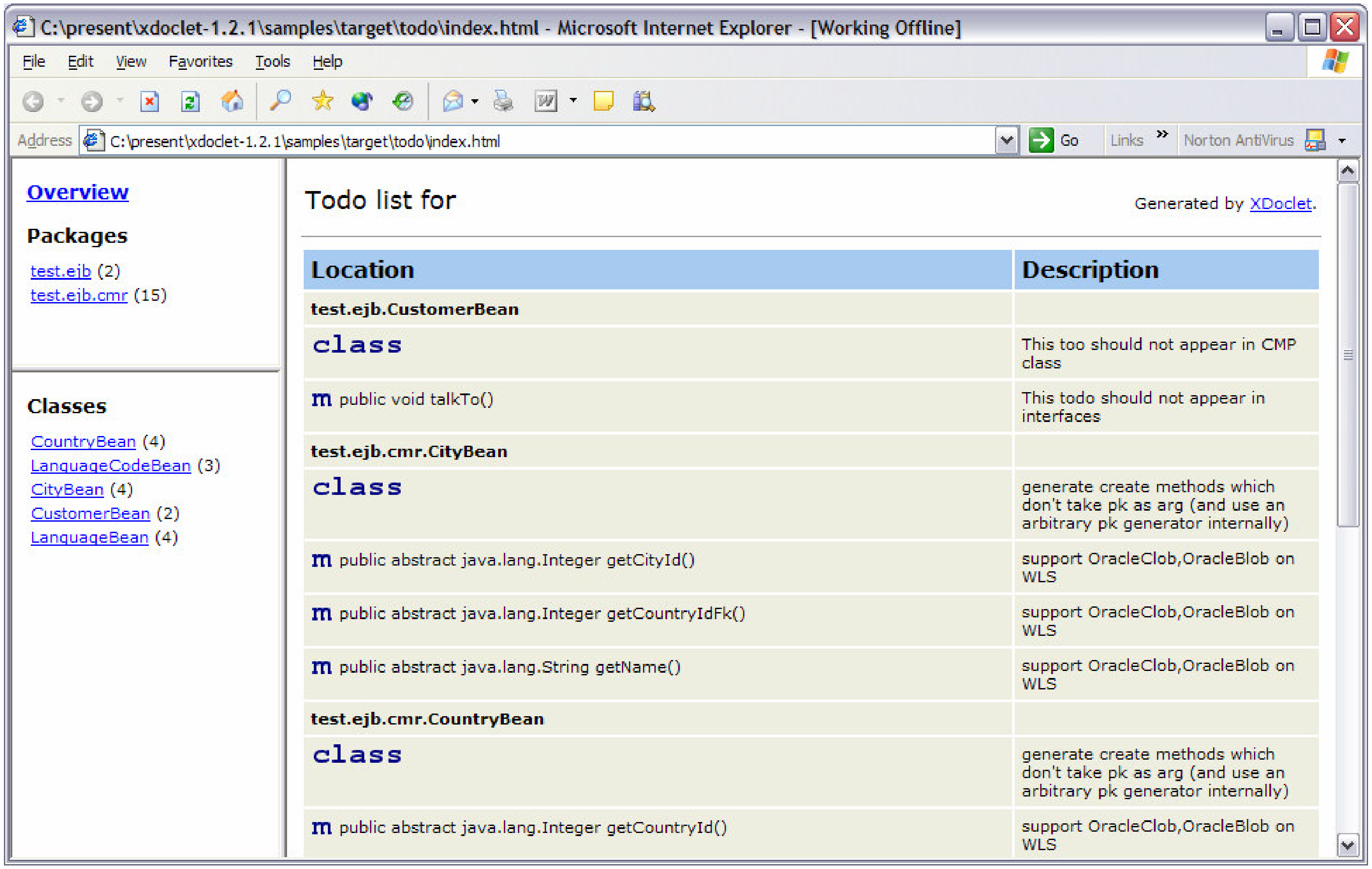
Agenda

- XDoclet Overview
- Todo List
- EJB File Generation
- Servlet File Generation
- Hibernate File Generation
- Custom File Generation

What is Xdoclet

- Code generation framework
- Attribute based (@ doclet tags)
- Popular (95,000 downloads in 2003)
- Originated as EJBDoclet to generate EJB artifacts (interfaces, deployment descriptors, etc.)
- Evolved to generate even more stuff (Hibernate mapping files, web.xml, TLD files, etc.)
- Built in support for many leading tools (Jboss, WebLogic, WebSphere, Castor, Hibernate, Struts, etc.)

ToDo List Example


C:\present\xdoclet-1.2.1\samples\target\todo\index.html - Microsoft Internet Explorer - [Working Offline]
File Edit View Favorites Tools Help
Address C:\present\xdoclet-1.2.1\samples\target\todo\index.html
Go
Links
Norton AntiVirus

Overview

Packages

test.ejb (2)
test.ejb.cmr (15)

Classes

CountryBean (4)
LanguageCodeBean (3)
CityBean (4)
CustomerBean (2)
LanguageBean (4)

Todo list for

Generated by XDoclet.

Location	Description
test.ejb.CustomerBean	
class	This too should not appear in CMP class
m public void talkTo()	This todo should not appear in interfaces
test.ejb.cmr.CityBean	
class	generate create methods which don't take pk as arg (and use an arbitrary pk generator internally)
m public abstract java.lang.Integer getCityId()	support OracleClob,OracleBlob on WLS
m public abstract java.lang.Integer getCountryIdFk()	support OracleClob,OracleBlob on WLS
m public abstract java.lang.String getName()	support OracleClob,OracleBlob on WLS
test.ejb.cmr.CountryBean	
class	generate create methods which don't take pk as arg (and use an arbitrary pk generator internally)
m public abstract java.lang.Integer getCountryId()	support OracleClob,OracleBlob on WLS

Usage of todo tag

```
/**
 * This is a customer bean. It is an example of how to use the XDoclet tags.
 *
 * ...
 *
 * @todo This too should not appear in CMP class
 */

public abstract class CustomerBean
        extends PersonBean {
    private EntityContext ctx;
```

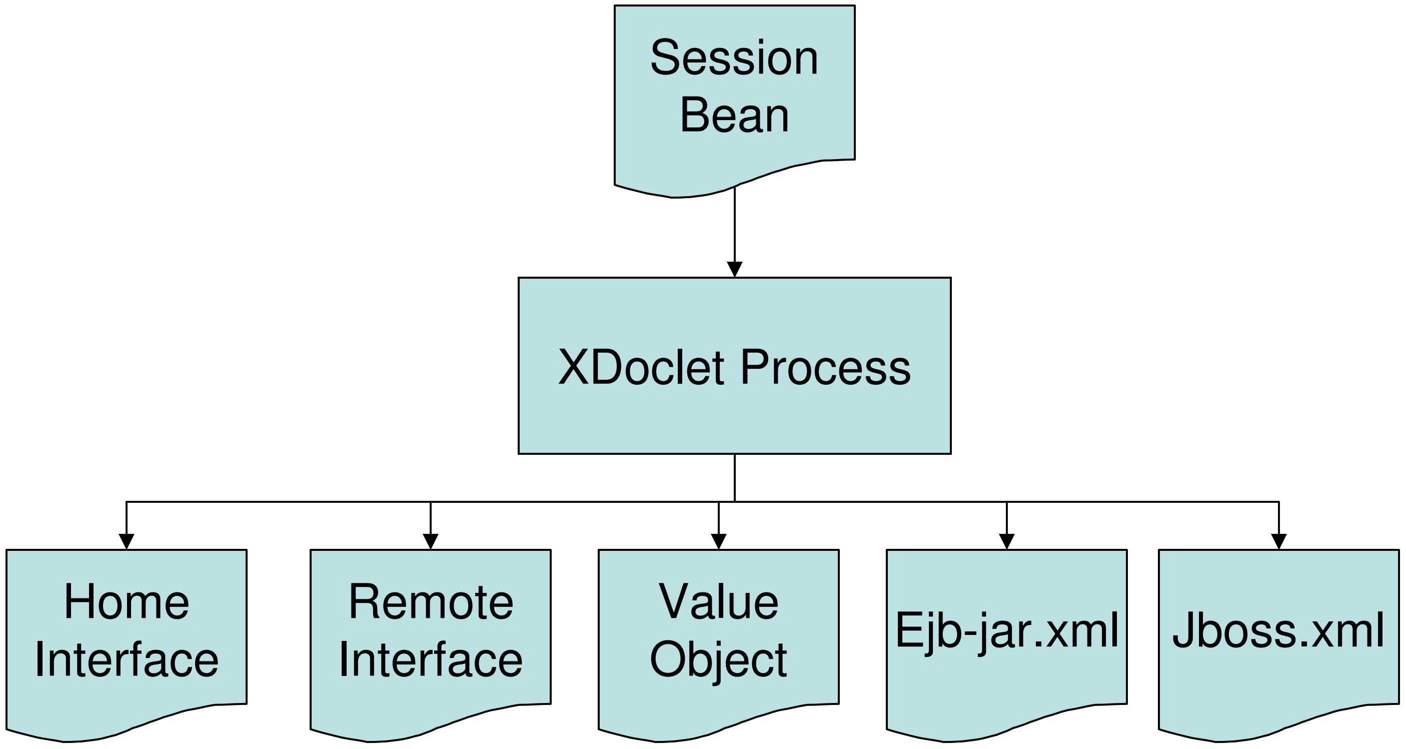
Ant task

```
•  <taskdef name="documentdoclet"
•       classname="xdoclet.modules.doc.DocumentDocletTask"
•       classpathref="samples.class.path"
•  />
•

•  <target name="todo" depends="prepare">
•          <echo>+-----------------------------------------+</echo>
•          <echo>| R U N N I N G   T O D O                 |</echo>
•          <echo>+-----------------------------------------+</echo>

•      <mkdir dir="${samples.todo.dir}"/>
•      <documentdoclet destdir="${samples.todo.dir}" >
•           <fileset dir="${samples.java.dir}">
•                <include name="**/*.java"/>
•           </fileset>
•           <info tag="todo"/>
•      </documentdoclet>
•  </target>
```

Generation of EJB Artifacts



Session Bean Example

```
• /**
•  * This is a teller bean. *
•  * @ejb.bean         name="Teller"
•  *                   description="Teller example bean"
•  *                   jndi-name="ejb/bank/Teller"
•  *                   type="Stateless"
•  *
•  */
• public abstract class TellerBean extends BaseTellerBean
  implements SessionBean {
• …
• }
```

Session Bean Details

```
•	/**
•	 * This is a teller bean. It is an example of how to use the XDoclet tags.
•	 *
•	 * @ejb.bean         name="Teller"
•	 *                   description="Teller example bean"
•	 *                   jndi-name="ejb/bank/Teller"
•	 *                   type="Stateless"
•	 *
•	 * @ejb.ejb-ref      ejb-name="Account"
•	 *                   ref-name="ejb/bank/Account"
•	 *
•	 * @ejb.ejb-ref      ejb-name="Customer"
•	 *
•	 * @ejb.security-role-ref role-link="Administrator"
•	 *                        role-name="admin"
•	 *
•	 * @ejb.permission   role-name="Teller"
•	 * @ejb.permission   role-name="Administrator"
•	 *
•	 * @ejb.transaction  type="Required"
•	 * @ejb.transaction-type type="Container"
•	 *
•	 * @ejb.resource-ref res-auth="Container"
•	 *                   res-name="jdbc/DBPool"
•	 *                   res-type="javax.sql.DataSource"
•	 *
•	 * @soap.service    urn="TellerService"
•	 *
•	 * @jboss.resource-ref res-ref-name="jdbc/DBPool"
•	 *                     resource-name="MyDataSourceManager"
•	 *
•	 * @jboss.container-configuration name="Standard Stateless
	SessionBean"
•	 *
•	 * @jboss.ejb-ref-jndi jndi-name="ejb/bank/Account"
•	 *                     ref-name="bank/Account"
•	 *
•	 * @weblogic.pool      initial-beans-in-free-pool="1"
•	 *                     max-beans-in-free-pool="3"
•	 *
•	 * @weblogic.stateless-clustering stateless-bean-call-router-class-
	name="Whatever"
•	 *                                stateless-bean-is-clusterable="True"
•	 *                                stateless-bean-load-algorithm="Whazzup"
•	 *                                stateless-bean-methods-are-idempotent="Sure"
•	 */
```

Session Bean Method

```
•   /**
•        * Transfer money between accounts.
•        *
•        * @ejb.interface-method view-type="remote"
•        */
•       public void transfer(Account from, Account to,
                               float amount) {
•           try {
•               from.withdraw(amount);
•               to.deposit(amount);
•           }
•           catch (java.rmi.RemoteException e) {
•               throw new EJBException(e);
•           }
•       }
```

Ant Task

```
<target name="ejbdoclet" depends="prepare">
        <ejbdoclet
            destdir="${samples.gen-src.dir}"
            mergedir="parent-fake-to-debug"
            excludedtags="@version,@author,@todo"
            addedtags="@xdoclet-generated at ${TODAY},@copyright The XDoclet Team,@author
    XDoclet,@version ${version}"
            ejbspec="2.0"
            force="${samples.xdoclet.force}"
            verbose="false"
            >

            <fileset dir="${samples.java.dir}">
                <include name="test/ejb/*Bean.java"/>
                <include name="test/ejb/cmr/*Bean.java"/>
                <include name="test/ejb/jdo/*.java"/>
                <exclude name="test/ejb/Base*.java"/>
                <exclude name="test/ejb/SecurityOfficerBean.java"/>
            </fileset>

            <packageSubstitution packages="ejb" substituteWith="interfaces"/>

            <remoteinterface/>
            <localinterface/>
            <homeinterface/>
            <localhomeinterface/>

            <dataobject/>
            <valueobject/>

            <entitypk/>

            <entitycmp/>
            <entitybmp/>
            <session/>
```

Ant Task (cont.)

```
        <deploymentdescriptor
            destdir="${samples.meta-inf.dir}"
            validatexml="true"
            mergedir="fake-to-debug"
            description="bæbæ"
            >
            <configParam name="clientjar" value="blah.jar"/>
        </deploymentdescriptor>


        <jboss
            version="3.2"
            unauthenticatedPrincipal="nobody"
            xmlencoding="UTF-8"
            destdir="${samples.meta-inf.dir}"
            validatexml="true"
            preferredrelationmapping="relation-table"
            />

    </ejbdoclet>

</target>
```

Generated EJB Descriptor

```
 <session >
         <description><![CDATA[Teller example
   bean]]></description>

         <ejb-name>Teller</ejb-name>

         <home>test.interfaces.TellerHome</home>
         <remote>test.interfaces.Teller</remote>
         <local-home>
test.interfaces.TellerLocalHome</local-home>
         <local>test.interfaces.TellerLocal</local>
         <ejb-class>test.ejb.TellerSession</ejb-class>
         <session-type>Stateless</session-type>
         <transaction-type>Container</transaction-type>
```

Generated EJB Descriptor

```
<ejb-ref >
            <ejb-ref-name>ejb/bank/Account</ejb-ref-name>
            <ejb-ref-type>Entity</ejb-ref-type>
            <home>test.interfaces.AccountHome</home>
            <remote>test.interfaces.Account</remote>
            <ejb-link>Account</ejb-link>
         </ejb-ref>
         <ejb-ref >
            <ejb-ref-name>ejb/Customer</ejb-ref-name>
            <ejb-ref-type>Entity</ejb-ref-type>
            <home>test.interfaces.CustomerHome</home>
            <remote>test.interfaces.Customer</remote>
            <ejb-link>Customer</ejb-link>
         </ejb-ref>
```

Generated EJB Descriptor

```
<security-role-ref>
           <role-name>admin</role-name>
           <role-link>Administrator</role-link>
        </security-role-ref>

        <resource-ref >
           <res-ref-name>jdbc/DBPool</res-ref-name>
           <res-type>
               javax.sql.DataSource</res-type>
           <res-auth>Container</res-auth>
        </resource-ref>

     </session>
```

JBoss Descriptor

```
<session>
        <ejb-name>Teller</ejb-name>
        <jndi-name>ejb/bank/Teller</jndi-name>
        <local-jndi-name>TellerLocal</local-jndi-name>
        <configuration-name>
            Standard Stateless SessionBean</configuration-name>
        <ejb-ref>
           <ejb-ref-name>ejb/bank/Account</ejb-ref-name>
           <jndi-name>ejb/bank/Account</jndi-name>
        </ejb-ref>
        <resource-ref>
           <res-ref-name>jdbc/DBPool</res-ref-name>
           <resource-name>MyDataSourceManager</resource-name>
        </resource-ref>

       <method-attributes>
       </method-attributes>
     </session>
```

WebLogic Descriptor

```
<weblogic-enterprise-bean>
      <ejb-name>Teller</ejb-name>
      <stateless-session-descriptor>
         <pool>
            <max-beans-in-free-pool>3
                </max-beans-in-free-pool>
            <initial-beans-in-free-pool>1
                </initial-beans-in-free-pool>
         </pool>
      </stateless-session-descriptor>
      <reference-descriptor>
      </reference-descriptor>
      <jndi-name>ejb/bank/Teller</jndi-name>
      <local-jndi-name>TellerLocal</local-jndi-name>
   </weblogic-enterprise-bean>
```

Home Interface

```
/*
 * Generated by XDoclet - Do not edit!
 */
package test.interfaces;

/**
 * Home interface for Teller.
 * @xdoclet-generated at 9-06-04
 * @copyright The XDoclet Team
 * @author XDoclet
 * @version ${version}
 */
public interface TellerHome
   extends javax.ejb.EJBHome
{
   public static final String COMP_NAME="java:comp/env/ejb/Teller";
   public static final String JNDI_NAME="ejb/bank/Teller";

   public test.interfaces.Teller create()
      throws javax.ejb.CreateException,java.rmi.RemoteException;

}
```

EJB Summary

- Write bean class
  - Everything is in one place
- Generate:
  - Descriptors
  - Interfaces
- Support for many app servers:
  - JBoss, BEA WebLogic, IBM WebSphere, Oracle IAS, Orion, Borland, MacroMedia JRun, Jonas, Pramati, Sybase EAServer and many more
- Entity & Message driven beans also
  - CMRs
  - Value Objects

Servlet

```
/**
 * Simple Servlet.
 * @web.servlet
 *     display-name="Simple Servlet"
 *     load-on-startup="1"
 *     name="SimpleServlet"
 *
 * @web.servlet-init-param
 *     name="param1"
 *     value="value1"
 *
 * @web.servlet-init-param
 *     name="param2"
 *     value="value2"
 *
 * @web.servlet-mapping
 *     url-pattern="/simple/*"
 */
public class SimpleServlet extends VelocityServlet {
    /**
     * Called by the server (via the service method) to allow a
     *   servlet to handle a POST request.
     */
    public void doPost(HttpServletRequest request, HttpServletResponse
    response)
            throws IOException, ServletException {
        // just print Hi!
        response.getWriter().println("Hi!");
    }
}
```

Servlet web.xml descriptor

```
<servlet>
     <servlet-name>SimpleServlet</servlet-name>
     <display-name>Simple Servlet</display-name>
    <servlet-class>test.web.SimpleServlet</servlet-class>

     <init-param>
        <param-name>param1</param-name>
        <param-value>value1</param-value>
     </init-param>
     <init-param>
        <param-name>param2</param-name>
        <param-value>value2</param-value>
     </init-param>

     <load-on-startup>1</load-on-startup>

  </servlet>

  <servlet-mapping>
     <servlet-name>SimpleServlet</servlet-name>
     <url-pattern>/simple/*</url-pattern>
  </servlet-mapping>
```

Servlet Filter

```
/**
 *
 * @web.filter
 *     display-name="Timer Filter"
 *     name="TimerFilter"
 *
 * @web.filter-init-param
 *     name="param1"
 *     value="value1"
 *
 * @web.filter-mapping
 *     url-pattern="*.xml"
 *
 */
public class TimerFilter implements Filter {
    ...
}
```

Filter Descriptor

```
 <filter>
      <filter-name>TimerFilter</filter-name>
      <display-name>Timer Filter</display-name>
      <filter-class>test.web.TimerFilter</filter-class>
      <init-param>
         <param-name>param1</param-name>
         <param-value>value1</param-value>
      </init-param>
</filter>

   <filter-mapping>
      <filter-name>TimerFilter</filter-name>
      <url-pattern>*.xml</url-pattern>
   </filter-mapping>
```

Hibernate

```
package test.hibernate;
import java.util.Set;
/**
 * @author Administrator
 *
 * @hibernate.class
 *  table="ANIMALS"
 *  dynamic-update="true"
 */
public class Animal extends Persistent {

   private Set prey;
   private char sex;

   /**
    * Constructor for Animal.
    */
   public Animal() {
       super();
   }
```

Hibernate (cont)

```
    /**
     * @hibernate.set
     *  lazy="true"
     *  table="PREDATOR_PREY"
     *  order-by="PREY_ID"
     * @hibernate.collection-key
     *  column="PREDATOR_ID"
     * @hibernate.collection-many-to-many
     *  column="PREY_ID"
     * @return Set
     */
    public Set getPrey() {
         return prey;
    }

    /**
     * @hibernate.property
     *  not-null="true"
     * Returns the sex.
     * @return char
     */
    public char getSex() {
         return sex;
    }
}
```

Custom Generation

- Write your own templates for the standard subtask
- Use the <template> task
- Write tag handlers / subtasks in Java

Example of custom template

- /*
- * <XDtI18n:getString bundle="xdoclet.ejb.Messages" resource="do_not_edit"/>
- */
- package <XDtPackage:packageOf><XDtClass:fullClassName/></XDtPackage:packageOf>;
- <XDtClass:importedList currentClass="<XDtClass:fullClassName/>"/>
- import java.sql.*;
- import org.apache.log4j.Logger;
- /**
- <XDtClass:forAllClassTags superclasses="false">
- <XDtClass:ifTagNameEquals value="@ejb:bean">
- * @ejb:bean
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="type"> *   type="<XDtClass:classTagValue tagName="ejb:bean" paramName="type" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="cmp-version"> *   cmp-version="<XDtClass:classTagValue tagName="ejb:bean" paramName="cmp-version" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="name"> *   name="<XDtClass:classTagValue tagName="ejb:bean" paramName="name" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="jndi-name"> *   jndi-name="<XDtClass:classTagValue tagName="ejb:bean" paramName="jndi-name" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="local-jndi-name"> *   local-jndi-name="<XDtClass:classTagValue tagName="ejb:bean" paramName="local-jndi-name" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="view-type"> *   view-type="<XDtClass:classTagValue tagName="ejb:bean" paramName="view-type" />"</XDtClass:ifHasClassTag>
- <XDtClass:ifHasClassTag tagName="ejb:bean" paramName="primkey-field"> *   primkey-field="<XDtClass:classTagValue tagName="ejb:bean" paramName="primkey-field" />Impl"</XDtClass:ifHasClassTag>
- *   generate="true"
- *
- </XDtClass:ifTagNameEquals>
- <XDtClass:ifTagNameEquals value="@ejb:pk">
- </XDtClass:ifTagNameEquals>
- <XDtClass:ifTagNameNotEquals value="@ejb:bean">
- <XDtClass:ifTagNameNotEquals value="@ejb:interface">
- <XDtClass:ifTagNameNotEquals value="@ejb:pk">
- * <XDtClass:getTag/> </XDtClass:ifTagNameNotEquals></XDtClass:ifTagNameNotEquals></XDtClass:ifTagNameNotEquals>
- </XDtClass:forAllClassTags>
- */

Lessons Learned

- Errors may be difficult to track down
  - Error reporting is bad to non-existant
  - Get the tags *Exactly* right
- Don’t do source control on generated files!

What’s wrong with XDoclet 1.2?

- Based on JavaDoc (limited to Java).
- Difficult and fixed template language (XDT).
- LARGE codebase (largely made up of unmaintained modules).
- Tightly coupled with Ant.
- No unit-tests.
- No tag validation.

Future Developments

- Velocity Templating
- XDoclet2 development in progress
- JBoss-IDE support for XDoclet tags
  - Didn’t work for me

Conclusion

- Q&A
- Demos?