1mm

Attributorientierte Programmierung mit XDoclet 1.2

Thomas Kraus

26. April 2004

Inhalt

260(10,200)

- Motivation
- Tag Definitionen
- Templates
- Exkurs Enterprise Java Beans
- Das Modul EJBDoclet
- Integration in IDE's
- XDoclet anpassen
- XDoclet erweitern
- Ausblick Java 1.5
- Zusammenfassung
- Ressourcen
- Folienübersicht

Motivation

260(10,200)

- Bei der Entwicklung werden oft identische oder ähnliche Codeblöcke benötigt (besonders J2EE)
- Kopieren ist zeitintensiv und fehleranfällig
- Continous Programming, Attribute Oriented Programming
- Nahtlose Integration in den Entwicklungszyklus ohne Werkzeugbruch
- Automatisiereung des Entwicklungsprozesses(Codegenerierung)
- Mögliche Lösung: **XDoclet** - Opensource Java Tool zur Codegenerierung
- Auf Basis von im Quellcode enthaltenen Metainformationen Generierung von:
  - Deployment-Descriptoren(Konfigurationsdateien)
  - Schnittstellen(Interfaces)
  - Quellcode
  - selbstdefinierte Reporte
- Daraus resultiert:
  - verkürzte Entwicklungszeiten durch Redundanzvermeidung
  - besserer Überblick über den Entwicklungsprozess
  - Konzentration auf die Geschäftslogik

Tag Definitionen

260(10,200)

```
/**
 * @namespace.tagname attributename="value" ...
 */
```

- XDoclet verwendet den eigenen Parser XJavaDoc
- XJavaDoc ist schneller und effizienter als Suns JavaDoc
- Intern wird Objektmodell der eingelesenen Klassen erzeugt
- Zugriff erfolgt über Template-Tags

Auswahl einiger Namespaces

260(10,200)

Namespace	Beschreibung
ejb	Standard EJB Informationen(nicht herstellerspezifisch)
web	erzeugt `web.xml` (Java Servlets)
portlet	Deskriptoren für die Java-Portlet API
jsp	Tag Library Extension Deskriptoren
soap	erzeugt SOAP Deskriptoren
jmx	erzeugt Java Management Extensions
struts	erzeugt `struts-config.xml`
jdo, cator, hibernate	Objektrelationales Mapping
jboss, websphere, weblogic	Application Server

Templates

260(10,200)

- Code-Schablonen, die die Ausgestaltung der zu erzeugenden Klassen oder Texte definieren
- bereits Templates für viele Dateitypen vorhanden
- verwenden XML-ähnliche Sprache
- Unterscheidung zwischen Block-Tags und Content-Tags
  - Block-Tags: für iterative und kausale Programmlogik

```
<XDtClass:forAllClasses>
</XDtClass:forAllClasses>
<XDtMethod:ifHasMethodTag tagName="deprecated">
</XDtMethod:ifHasMethodTag>
```

  - Content-Tags: erzeugen Ausgabe

```
<XDtMethod:methodType/>
<XDtPackage:packageName/>
```

Templates - Beispiel

260(10,200)

- Template `signature.xdt`:

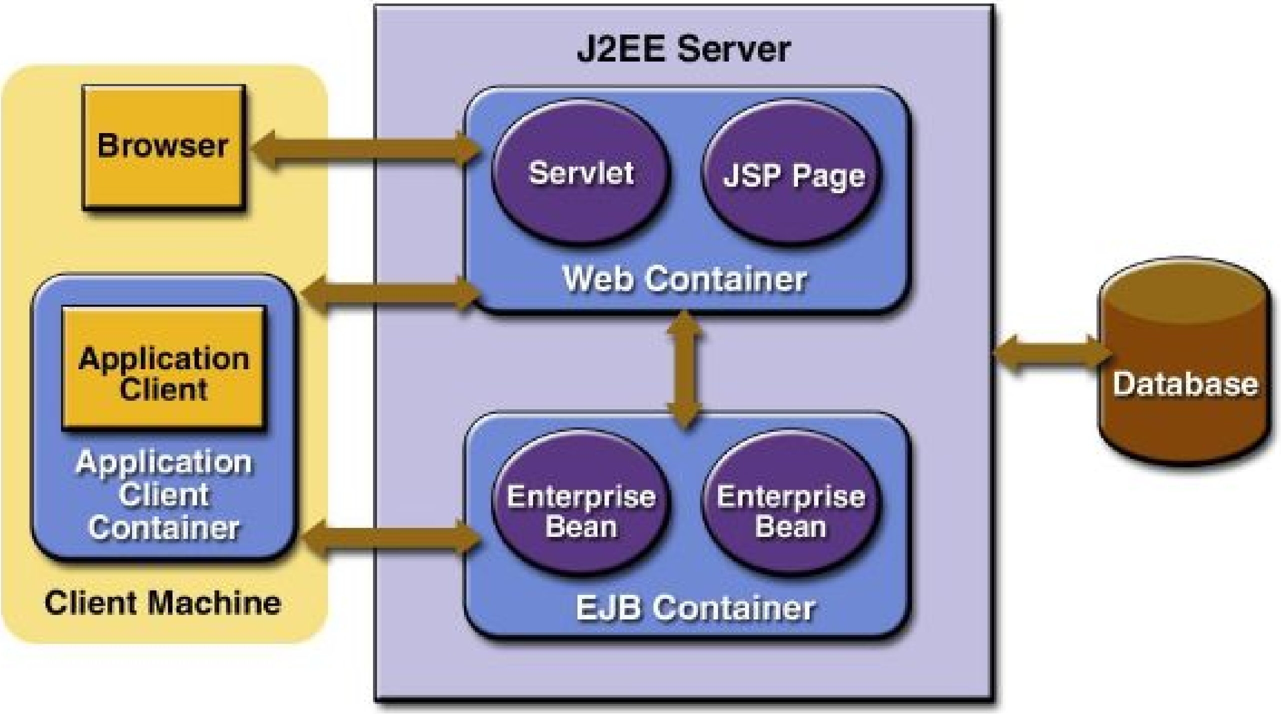
```
<XDtClass:forAllClasses>
   Class: <XDtPackage:packageName/>.<XDtClass:className/>
   <XDtMethod:forAllMethods>
      Method: <XDtMethod:methodType/><XDtMethod:methodName/>
              (<XDtParameter:parameterList/>)
   </XDtMethod:forAllMethods>
</XDtClass:forAllClasses>
```

- Aufruf als Ant Task:

```
<target name="class.index">
   <taskdef name="signature" classname="xdoclet.DocletTask" />
   <signature destDir="$dir.build">
     <fileset dir="$dir.src">
         <include name="**/*.java" />
     </fileset>
     <template templateFile="signature.xdt"
               destinatonFile="signature.txt" />
   </signature>
</target>
```

Exkurs Enterprise Java Beans

260(10,200)



Quelle: Sun Microsystems

Exkurs Enterprise Java Beans(2)

260(10,200)

- CMP komponenten Darstellen, Bean, Remote, Home, ValueObject, ejb-jar, Containerkonfiguration

Das Modul EJBDoclet

260(10,200)

Subtask	Codegenerierung
deploymentdescriptor	Serverunabhängige Deploymentdescriptoren
entitybmp	Entity Bean Klasse(BMP)
entitycmp	Entity Bean Klasse(CMP)
entitypk	Primärschlüssel-Klasse
homeinterface	Remote Home Interface
remoteinterface	Remote Interface
localhomeinterface	Local Home Interface
localinterface	Local Interface
jboss	JBoss Deployment-Descriptor

EJB-Tags auf Klassenebene

260(10,200)

```
/**
* m-Side of the relation
*
* @ejb.bean name="Customer"
*           display-name="Customer"
*           view-type="local"
*           type="CMP"
*           local-jndi-name="ejb/Customer"
*
* @ejb.pk class="edu.xdoclet.interfaces.CustomerPK"
*
**/
public abstract class CustomerEJB implements javax.ejb.EntityBean {
 ...
```

Tag	Bedeutung
ejb.bean	allgemeine Informationen über die Bean
ejb.ejb-external-ref	Referenz auf eine externe Bean
ejb.finder	definiert eine Finder-Methode
ejb.pk	definiert Primärschlüssel
ejb.persistence	Informationen über Persistenz (bei CMP)

EJB-Tags auf Methodenebene

260(10,200)

```
/**
* Retrieves Primary Key for a Customer.
*
* @return Returns a String representing CustomerPK.
*
* @ejb.persistence
* @ejb.pk-field
*
* @ejb.interface-method view-type="local"
* @ejb.transaction type="Required"
*
**/
public abstract String getCustomerPK();
```

Tag	Bedeutung
ejb.interface-method	Sichtbarkeit in den Schnittstellen
ejb.persistence	Informationen über Persistenz
ejb.pk-field	markiert Feld als Primärschlüssel
ejb.relation	definiert Relation zwischen Beans
ejb.transaction	definiert Transaktionsverhalten

EJBDoclet - Aufruf

260(10,200)

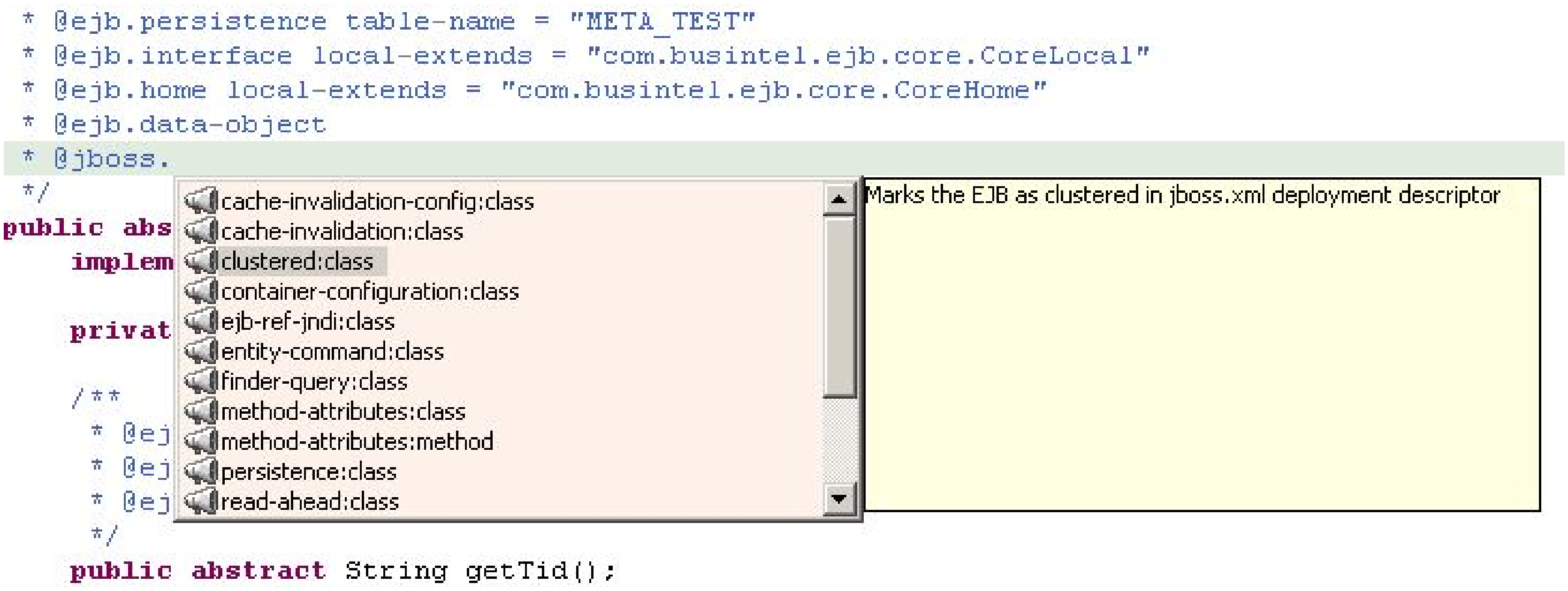
```
<target name="xdoclet">
  <taskdef name="ejbdoclet"
           classname="xdoclet.modules.ejb.EjbDocletTask"
           classpathref="xdocpath" />
  <ejbdoclet ejbspec="2.0"
             destDir="${gen.src}">
      <fileset dir="${src}">
         <include name="**/*EJB.java"/>
      </fileset>
      <deploymentdescriptor destdir="META-INF"/>
      <remoteinterface />
      <homeinterface />
  </ejbdoclet>
</target>
```

- Zu jeder Beanklasse `xxxEJB.java` werden die Interfaces `xxx.java`, `xxxHome.java` erzeugt
- Es wird die Datei `META-INF/ejb-jar.xml` mit entsprechenden Einträgen generiert

Integration in IDE's

260(10,200)

- Da als Ant-Task realisiert, in den Buildprozess jeder IDE mit Ant-Unterstützung integrierbar
- Live Templates für IntelliJ(IDEA)
- XDoclet Syntax-Highligting Profil für JEdit
- Über JBossIDE-Plugin in Eclipse integrierbar



```
 * @ejb.persistence table-name = "META_TEST"
 * @ejb.interface local-extends = "com.busintel.ejb.core.CoreLocal"
 * @ejb.home local-extends = "com.busintel.ejb.core.CoreHome"
 * @ejb.data-object
 * @jboss.
 */
public abs
    implem

    privat

    /**
      * @ej
      * @ej
      * @ej
      */
    public abstract String getTid();
```

Marks the EJB as clustered in jboss.xml deployment descriptor

XDoclet anpassen

260(10,200)

- Taskparameter
  - wichige Codeteile sind oft schon parameterisiert
  - `<remoteinterface ofType"edu.MyExtendedEJBInterface"` /¿
- Mergedirs
  - mischt den erzeugten Code mit bereits vordefiniertem Code
  - nützlich bei Deployment-Descriptoren (web.xml, ejb-jar.exl ...)
  - `<deploymetdescriptor mergeDir="META-INF" />`
- Mergepoints
  - erweitern die Templates während der Codeerzeugung
  - nütlich um zum Beispiel weitere Funktionen in der Klasse zu generieren
  - z.B. in der Datei `entitycmp.xdt` :

```
<XDtMerge:merge file="entitycmp-custom.xdt" />
```

XDoclet erweitern - Subtasks

260(10,200)

```
import xdoclet.TemplateSubTask;
import xdoclet.XDocletException;
/**
 * @ant.element name="mydoclet"
 *              display-name="Mydoclet-Subtask"
 *              parent="xdoclet.DocletTask"
 */
 public class MyDocletSubTask extends TemplateSubTask {

   private String id;

   public MyDocletSubTask() {
        setDestinatinFile("Test.java");
        setTemplateURL(getClass().getRessource("myTemplate.xdt"));
   }

   public void setId(String id) {
        this.id = id;
   }

   protected void generateForClass(XClass clazz)
        throws XDocletException;
        //hier eigener Code
        return super.generateForClass(clazz);
   }
}
```

XDoclet erweitern - Tasks

260(10,200)

```
import xdoclet.DocletTask;

public class MyDocletTask extends DocletTask {

    private String param;

    public void setParam(String param) {
        this.param = param;
    }

    protected void validateOptions() {
        super.validateOptions();
        if(param = null || "".equals(param))
            throw new BuildException("’param’ not valid");
    }
}
```

- Subtasks können die übergebenen Parameter mit folgender Methode auslesen:

```
getContext().getConfigParam("param");
```

XDoclet erweitern - Taghandler

260(10,200)

```
/**
 * @xdoclet.taghandler namespace="MyTags"
 */
 public class MyTagHandler extends XDocletTagSupport {

        public void iterator(String template) {
            for(int i=0; i<10;i++)
                generate(template);
        }

        public String version() {
            return "v0.98";
        }
 }
```

- `<XDtMyTags:version /> -> String MyTagHandler#version()`
- `<XDtMyTags:iterator>some code</XDtMyTags:iterator>`
- `        -> void MyTagHandler#iterator(come code)`

XDoclet erweitern - Module

260(10,200)

- Zusammenhängende Templates, Tasks, Subtasks werden als Jar-Modul gruppiert
- xxx-module.jar enthält Deloyment-Descriptor `META-INF/xdoclet.xml`

```
<xdoclet-module>
    <subtask name="mydoclet"
             implementation-class="MyDocletSubTask"
             parent-task-class="MyDocletTask"/>

    <taghandler namespace="MyTags"
                class="MyTagHandler"/>
</xdoclet-module>
```

Ausblick - Java 1.5 Annotations

260(10,200)

- Neues Feature von Java 1.5
- JSR 175: A Metadata Facility for the Java^{TM}Programming Language

```
package junit.framework;
import java.lang.annotation.*;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
public @interface Ignore {}
...

private boolean idIgnore(Method m) {
    return m.isAnnotationPresent(Ignore.class);
}

package mytests;

public class MyTestClass extends TestCase {

    @Ignore
    public void testCase() {
    }

    public void testNotIgnored() {
    }
}
```

Zusammenfassung

260(10,200)

- In den Buildprozess integrierbares Tool zur Codegenerierung
- Codegenerierung anhand von Attributen in Java Kommentaren
- Generierungsvorschrift liegt in .xdt - Templates vor
- zur Erzeugeung von Interfaces, Deployment-Desriptoren, Code, Reporten
- besonders effizient beim Programmieren mit der Java2 Enterprise Edition

Ressourcen

260(10,200)

- `sourceforge.net/projects/xdoclet` - Offizielle Prokektseite
- `http://ant.apache.org` - Apache Ant Build Tool
- Java Development with Ant
    - Enthält unter anderem ein Kapitel über XDoclet
    - `http://www.manning.com/antbook/`
- Codegenerierung mit XDoclet am Beispiel von Entity Beans
    - Seminararbeit von Dennis Hurrelmann an der Uni Mannheim
    - `http:???`

Folienübersicht

260(10,200)

Folienübersicht(2)

260(10,200)

- XDoclet anpassen
- XDoclet erweitern - Subtsks
- XDoclet erweitern - Tasks
- XDoclet erweitern - Taghandler
- XDoclet erweitern - Module
- Ausblick - Java 1.5 Metatags
- Zusammenfassung
- Ressorcen