eFID 系統中ANT 的應用

Ant的使用可以方便我們在javaproject模式下開發的工程進行webproject的整理。

讓我們分析一下，eFID系統中的Ant的應用

<project name="eFID" default="help" basedir=".">

Project 是Ant中的基本組成，每一個Ant的build.xml文件都至少有一個project。

Default 是在默認的情況下，執行哪一個target

Basesdir 用來指定基路徑

<property environment="env"/>

Property 用來在xml文件中定義一些屬性變量，environment="env"是一個特殊的屬性，用來獲取系統的環境變量。

<!-- 本範本版本 -->

<property name="template.version" value="4.3"/>

<!-- 專案名稱 -->

<property name="project.name" value="eFID"/>

<!-- JAR 壓縮工具 -->

<property name="jar.builtby" value="ANT ${ant.version}"/>

<!-- JAR 壓縮者 -->

<property name="jar.createdby" value="Foxconn"/>

<!-- JAR 使用編碼 -->

<property name="jar.encoding" value="UTF-8"/>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<!-- 專案路徑與檔名 -->

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<!-- 編譯時用到的 jar -->

<property name="lib.path" value="./lib"/>

<!-- 壓縮應用程式的 WAR 名稱 -->

<property name="war.name" value="eFID.war"/>

<!-- 原始程式放置目錄 -->

<property name="src.path" value="./src"/>

<!-- 封裝好的 EAR,WAR,JAR 檔案路徑 -->

<property name="bin.path" value="./bin"/>

<!-- 編譯後的 .class 放置路徑 -->

<property name="build.path" value="./classes"/>

<!-- 文件目錄 -->

<property name="doc.path" value="./doc"/>

<!-- SA 文件目錄 -->

<property name="sa.path" value="${doc.path}/SA"/>

<property name="sa.structure.path" value="${sa.path}/Architect"/>

<property name="sa.req.path" value="${sa.path}/Requirement"/>

<!-- SD 文件目錄 -->

<property name="sd.path" value="${doc.path}/SD"/>

<property name="sd.spec.path" value="${sd.path}/Spec"/>

<property name="sd.guide.path" value="${sd.path}/Guide"/>

<property name="sd.manual.path" value="${sd.path}/Manual"/>

<!-- API 文件目錄 -->

<property name="javadoc.path" value="${doc.path}/javadoc"/>

<!-- WEB apps (JSP) 路徑 -->

<property name="web.path" value="./web"/>

<property name="web.classes.path" value="${web.path}/WEB-INF/classes"/>

<property name="checkstyle.jar" value="${lib.path}/checkstyle-all-4.3.jar"/>

<property name="j2h.jar" value="${lib.path}/j2h.jar"/>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<!-- javadoc 文件參數 -->

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<property name="javadoc.packages" value="iPLM.\*"/>

<property name="javadoc.title" value="${project.name}"/>

<property name="javadoc.copyright" value="${jar.createdby}"/>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<!-- CVS 連結變數 -->

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<property name="cvs.uid" value="anoncvs"/>

<property name="cvs.passwd" value="anoncvs"/>

<property name="cvs.module" value="irm.project.e-FID"/>

<property name="cvs.server" value="lhsc01"/>

<property name="cvs.passfile" value="./cvspass"/>

<property name="cvs.repository" value="/eFID"/>

<property name="cvs.work.path" value="${basedir}"/>

<property name="cvs.root" value=":pserver:${cvs.uid}:${cvs.passwd}@${cvs.server}:${cvs.repository}"/>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<!-- SpringFramework 變數 -->

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<property name="springframework.sql.url" value="jdbc:oracle:thin:@10.153.24.96:1521:irmdb"/>

<property name="springframework.sql.driver" value="oracle.jdbc.driver.OracleDriver"/>

<property name="springframework.sql.uid" value="efid"/>

<property name="springframework.sql.passwd" value="test"/>

<property name="springframework.sms.wsdl" value="http://10.153.24.188:8008/ESMS/SMSMessageService?wsdl"/>

<property name="springframework.fibc.wsdl" value="http://10.153.24.116:8080/web/services/MessageService?wsdl"/>

<property name="springframework.ftp.realPath" value="c:\eFID"/>

<property name="springframework.ftp.image.realPath" value="c:\Images"/>

<property name="springframework.kpi.path" value="c:\\KPI\\"/>

<property name="springframework.kpi.weekly.url" value="http://fid.efih-foxconn.com/WeeklyKPI"/>

<property name="springframework.kpi.monthly.url" value="http://fid.efih-foxconn.com/MonthlyKPI"/>

<property name="springframework.ftp.root" value="ftp://efid:8jamnia3@fid.efih-foxconn.com/eFIDSZ"/>

<property name="springframework.ftp.image.root" value="ftp://foxconncic:fox.123@10.153.24.94"/>

<property name="springframework.contextPath" value="http://fid.efih-foxconn.com"/>

<property name="springframework.news.contextPath" value="http://fid.efih-foxconn.com/news/"/>

<property name="springframework.hibernate.dialect" value="org.hibernate.dialect.OracleDialect"/>

<property name="springframework.lucene.indexDir" value="file:///c:/Index"/>

<property name="springframework.cost.chargeCode" value="R00908"/>

<property name="springframework.cost.factor" value="160.0f"/>

上面則定義了整個Ant build.xml文件所用到的屬性

<path id="class.path">

<fileset dir="${lib.path}">

<include name="\*.jar"/>

</fileset>

<fileset dir="${web.path}/WEB-INF/lib">

<include name="\*.jar"/>

</fileset>

<!-- Add this for hibernate reference -->

<pathelement location="${build.path}"/>

</path>

Path指明了路徑，講\*.jar保存在lib.path目錄下，還要將其保存在${web.path}/WEB-INF/lib目錄下，pathelement 引入要用到的路徑(不是很清楚)

<patternset id="java.files.pattern" includes="\*\*/\*.java"/>

引入要編譯的類，表示所有的以java結尾類都要對其進行編譯。

<target name="help">

<echo message="=============================================================="/>

<echo message="${project.name} 系統編譯器 ${template.version}"/>

<echo message="=============================================================="/>

<echo message="ANT: ${ant.version} "/>

<echo message="JDK: ${ant.java.version}"/>

<echo message="USR: ${user.name}"/>

<echo message="--------------------------------------------------------------"/>

<echo message="語法: ant [指令]"/>

<echo message="指令:"/>

<echo message="build : 編譯原始程式"/>

<echo message="clean : 清除所產生的所有可執行檔"/>

<echo message="cvs : 自 CVS 取得最新原始碼"/>

<echo message="db.create : 在資料庫上建立 Schema "/>

<echo message="db.schema : 產生資料庫建立腳本檔 "/>

<echo message="help : 說明 "/>

<echo message="hibernate : 產生 Hibernate 映射檔"/>

<echo message="init : 建立所需的資料夾"/>

<echo message="javadoc : 產生 API 文件"/>

<echo message="war : 包裝客戶端元件 WAR"/>

</target>

Help 指令，用來輸入幫助信息。 Echo是每一個信息元素。

<target name="init">

<mkdir dir="${build.path}"/>

<echo message="編譯後的程式放置於: ${build.path}"/>

<mkdir dir="${javadoc.path}"/>

<echo message="API 文件放置於: ${javadoc.path}"/>

<mkdir dir="${bin.path}"/>

<echo message="包裝後的應用程式放置於: ${bin.path}"/>

<echo message="更新 Springframework 變數 ..."/>

<propertyfile

file="${build.path}/spring.properties"

comment="SpringFramework variables properties">

<entry key="jbpm.jdbc.url" value="${springframework.sql.url}"/>

<entry key="jbpm.jdbc.username" value="${springframework.sql.uid}"/>

<entry key="jbpm.jdbc.password" value="${springframework.sql.passwd}"/>

<entry key="wsdl.sms" value="${springframework.sms.wsdl}"/>

<entry key="wsdl.fibc" value="${springframework.fibc.wsdl}"/>

<entry key="ftp.realPath" value="${springframework.ftp.realPath}"/>

<entry key="ftp.image.realPath" value="${springframework.ftp.image.realPath}"/>

<entry key="ftp.root" value="${springframework.ftp.root}"/>

<entry key="ftp.image.root" value="${springframework.ftp.image.root}"/>

<entry key="contextPath.url" value="${springframework.contextPath}"/>

<entry key="cost.chargeCode" value="${springframework.cost.chargeCode}"/>

<entry key="cost.factor" value="${springframework.cost.factor}"/>

<entry key="kpi.path" value="${springframework.kpi.path}"/>

<entry key="kpi.weekly.url" value="${springframework.kpi.weekly.url}"/>

<entry key="kpi.monthly.url" value="${springframework.kpi.monthly.url}"/>

<entry key="indexer.dir" value="${springframework.lucene.indexDir}"/>

<entry key="contextPath.news.url" value="${springframework.news.contextPath}"/>

</propertyfile>

<mkdir dir="${doc.path}/Report"/>

</target>

Init指令，用來對程序進行初始化，前三個mkdir用來在相應的路徑下生成三個文件夾，分別存放編譯後的程序，API文件以及WAR包。

Propertyfile 用來生成一個資源文件,file用來指明存放的路徑，comment是對其進行的申明，entry用來設置資源文件的信息，即Key - value 的鍵值對，給資源文件主要是用來配置數據庫信息的。

<target name="build" depends="init">

<echo message="編譯原始程式中..."/>

<javac srcdir="${src.path}"

destdir="${build.path}"

debug="on"

deprecation="off"

optimize="on"

excludes="${web.path}"

memoryInitialSize="512m"

memoryMaximumSize="1024m"

fork="true"

target="1.5"

source="1.5"

nowarn="on"

encoding="UTF-8"

>

<classpath refid="class.path"/>

</javac>

<echo message="編譯完畢:${build.path}"/>

<copy toDir="${build.path}">

<fileset dir="${src.path}">

<include name="\*.properties"/>

<include name="\*.xml"/>

</fileset>

</copy>

<copy toDir="${web.path}/WEB-INF/classes">

<fileset dir="${web.path}/WEB-INF">

<include name="log4j.properties"/>

</fileset>

</copy>

<copy toDir="${build.path}/iPLM/vo">

<fileset dir="${src.path}/iPLM/vo">

<include name="\*.xml"/>

</fileset>

</copy>

<copy toDir="${build.path}/resources/image">

<fileset dir="${src.path}/resources/image">

<include name="\*\*"/>

</fileset>

</copy>

<copy toDir="${build.path}/resources/mail">

<fileset dir="${src.path}/resources/mail">

<include name="\*\*"/>

</fileset>

</copy>

<!-- generate hibernate files-->

<antcall target="hibernate"/>

</target>

Build 是對工程進行編譯，其中javac是對java類進行編譯的命令，srcdir指定了要編譯的類的路徑，destdir是編譯後的class文件放的路徑, classpath是編譯用到的路徑。下麵的幾個copy要用到的一些文件拷貝到相應的目錄下。Antcall 用來調用其他的指令，該處調用了hibernate指令。

<target name="hibernate">

<echo message="產生 Hibernate 映射檔中..."/>

<taskdef name="hibernatedoclet"

classname="xdoclet.modules.hibernate.HibernateDocletTask"

>

<classpath refid="class.path"/>

</taskdef>

<hibernatedoclet destdir="${build.path}"

excludedtags="@version,@author"

force="true">

<fileset dir="${src.path}">

<include name="iPLM/vo/\*.java"/>

</fileset>

<hibernate validatexml="true" version="3.0"/>

<hibernatecfg

poolSize="10"

dialect="${springframework.hibernate.dialect}"

driver="${springframework.sql.driver}"

jdbcUrl="${springframework.sql.url}"

userName="${springframework.sql.uid}"

password="${springframework.sql.passwd}"

version="3.0"

destDir="${build.path}"

cacheProviderClass="org.hibernate.cache.EhCacheProvider"

>

<otherProperty name="show\_sql" value="true"/>

<otherProperty name="hibernate.autoReconnect" value="true"/>

<otherProperty name="hibernate.cache.use\_second\_level\_cache" value="true"/>

</hibernatecfg>

</hibernatedoclet>

<echo message="生成完畢!"/>

</target>

該指令，是用來產生相應的hibernate映射，即和Vo類想對應的xml文件