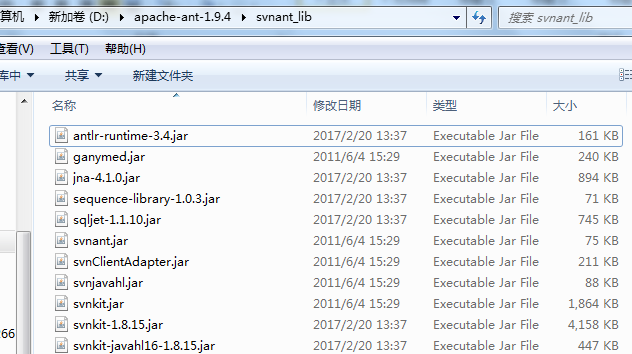
***ANT脚本部署的环境配置及使用***

1. **Java运行环境JDK, 在本例中我使用的是32位的jdk1.7.0\_25, 安装包的下载地址可参考- http://www.oracle.com/technetwork/java/javase/downloads/index.html , 安装步骤请自行百度;**
2. **安装Ant, 在本例中我使用的是apache-ant-1.9.4, 安装包的下载地址可参考- http://ant.apache.org/bindownload.cgi , 下载apache-ant-1.9.4-bin.zip, 下载完之后解压到一个位置, 之后的环境变量配置可以参考这位大神写的, http://www.cnblogs.com/hoojo/archive/2013/06/14/java\_ant\_project\_target\_task\_run.html ;**
3. **安装svn服务器, 如果你已经在用svn作为代码的版本控制那请跳过这一步,如果没有,下载地址可参考 http://subversion.apache.org/packages.html , 本例中使用的svn版本是1.7.4,安装及配置用户和权限还有代码管理请自行百度;**

**4.到你的apache-ant-1.9.4-bin的安装路径中lib文件夹里面新建一个文件夹名为svnant\_lib, 需要用到svnant的类库,下载地址可参考 http://subclipse.tigris.org/files/documents/906/49042/svnant-1.3.1.zip , 下载完之后解压, 把lib文件夹内的ganymed.jar, svnant.jar, svnClientAdapter.jar 复制到svnant\_lib文件夹内**

**5.由于svnant很久没更新, 所以svnant-1.3.1不支持1.7或以上的svn, 所以在上一步我们没有复制有关svnkit的jar, 这里要单独下载, 下载地址可参考http://www.svnkit.com/download.php , 下载org.tmatesoft.svn\_1.8.6.standalone.zip, 本例中用的是1.8.6版, 同样下载之后解压, 吧lib文件夹内的antlr-runtime-3.4.jar, jna-4.1.0.jar, sequence-library-1.0.3.jar, sqljet-1.1.10.jar,svnkit-1.8.6.jar,svnkit-javahl16-1.8.6.jar复制到svnant\_lib文件夹内,完成之后如下图所示:**



**6.因为某些项目使用到了泛型、annotation注解，使用javac有些代码是编译不通过的，所以这里使用了jdt的编译方式。需要用到:**

**jdtCompilerAdapter.jar**

**org.eclipse.jdt.compiler.tool\_1.0.1.v\_793\_R33x.jar**

**org.eclipse.jdt.core\_3.3.3.v\_793\_R33x.jar**

**org.eclipse.jdt.debug.ui\_3.2.102.v20071002\_r332.jar**

**参考的下载地址http://download.csdn.net/download/wosishui21/5947899 ,下载之后复制进去apache-ant-1.9.4-bin的安装路径中lib文件夹里面;**

**7.需要用到自己写的java类打成jar，increment.export.jar：源码如下：**

**ExportIncrementFiles：下载差异文件的类**

package com.cntest.util;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileReader;

public class ExportIncrementFiles {

/\*\*

\*

\* @param configPath

\* 差异文件路径配置目录

\* @param baseDir

\* 新版文件基本路径目标位置

\* @param destDir

\* 增量文件保存位置

\* @param oldBaseDir

\* 旧版文件基本路径目标位置

\* @param decrementDestDir

\* 减量文件保存位置

\* @throws Exception

\*

\* @author chenyou

\*/

private static void export(String configPath, String baseDir,

String destDir, String oldBaseDir, String decrementDestDir)

throws Exception {

String srcFile = baseDir +"/"+ configPath;

String desFile = destDir +"/"+ configPath;

String oldSrcFile = oldBaseDir +"/"+ configPath;

String decrementDesFile = decrementDestDir +"/"+ configPath;

int lastIndex = desFile.lastIndexOf("/");

String desPath = desFile.substring(0, lastIndex);

int lastIndexDecrement = decrementDesFile.lastIndexOf("/");

String decrementDesPath = decrementDesFile.substring(0,

lastIndexDecrement);

File srcF = new File(srcFile);

if (srcF.exists()) {

// 如果不存在这样的源文件，就把该文件放到减量文件目录那里，这个用来解决版本之间有删除文件的情况。

File desF = new File(desFile);

File desP = new File(desPath);

if (!desP.exists()) {

desP.mkdirs();

}

System.out.println(srcFile);

// 判断最终路径是文件夹还是文件，是文件的话复制文件，文件夹的话创建文件夹

if (srcF.isFile()) {

FileInputStream fis = new FileInputStream(srcF);

FileOutputStream fos = new FileOutputStream(desF);

byte[] buf = new byte[1024];

int len = 0;

while ((len = fis.read(buf)) != -1) {

fos.write(buf, 0, len);

}

fos.flush();

fos.close();

fis.close();

} else if (srcF.isDirectory()) {

desF.mkdir();

}

} else {

File oldSrcF = new File(oldSrcFile);

File desF = new File(decrementDesFile);

File desP = new File(decrementDesPath);

if (!desP.exists()) {

desP.mkdirs();

}

System.out.println(oldSrcFile);

// 判断最终路径是文件夹还是文件，是文件的话复制文件，文件夹的话创建文件夹

if (oldSrcF.isFile()) {

FileInputStream fis = new FileInputStream(oldSrcF);

FileOutputStream fos = new FileOutputStream(desF);

byte[] buf = new byte[1024];

int len = 0;

while ((len = fis.read(buf)) != -1) {

fos.write(buf, 0, len);

}

fos.flush();

fos.close();

fis.close();

} else if (oldSrcF.isDirectory()) {

desF.mkdir();

}

}

}

/\*\*

\*

\* @param args

\* 参数0 差异文件路径，参数1 新版源文件路径，参数2 增量文件保存目标路径，参数3

\* 旧版源文件路径，参数4减量文件保存目标路径

\*/

public static void main(String[] args) {

if (args.length > 0) {

if (args.length == 1 && "help".equals(args[0])) {

System.out

.println("args[0] is Export Increment Files content path");

System.out

.println("args[1] is Export Increment Files target path");

System.out

.println("args[2] is Increment Files Export loaction");

System.out

.println("args[3] is Export Old Source Files target path");

System.out

.println("args[4] is Decrement Files Export loaction");

} else {

String configPath = args[0];

String baseDir = args[1];

String destDir = args[2];

String oldBaseDir = args[3];

String decrementDestDir = args[4];

try {

BufferedReader br = new BufferedReader(new FileReader(

configPath));

String s = null;

while ((s = br.readLine()) != null) {

s = s.trim();// 去掉路径前面的空格

String str = destDir + s;

if (!destDir.equals(str)) {// 过滤空行

export(s, baseDir, destDir, oldBaseDir,

decrementDestDir);

}

}

br.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

}

}

**Unzip.java war包解压工具**

package com.cntest.util;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.util.ArrayList;

import java.util.zip.ZipEntry;

import java.util.zip.ZipInputStream;

/\*\*

\* 解压

\* @author chenyou

\*/

public class Unzip {

@SuppressWarnings( { "unchecked", "static-access" })

public static void main(String[] args) {

if (args.length > 0) {

if (args.length == 1 && "help".equals(args[0])) {

System.out

.println("args[0] is ZIP File content path!");

System.out

.println("args[1] is zip decompression path!");

}else{

String zipPath = args[0];

String todir = args[1];

Unzip z = new Unzip();

ArrayList<String> a = z.Ectract(args[0],args[1]); // 返回解压缩出来的文件列表

for(String s : a){

System.out.println(s);

}

}

}

}

/\*\*

\* 解压缩

\* @param sZipPathFile 要解压的文件

\* @param sDestPath 解压到某文件夹

\* @return

\*/

@SuppressWarnings("unchecked")

public static ArrayList Ectract(String sZipPathFile, String sDestPath) {

ArrayList<String> allFileName = new ArrayList<String>();

try {

// 先指定压缩档的位置和档名，建立FileInputStream对象

FileInputStream fins = new FileInputStream(sZipPathFile);

// 将fins传入ZipInputStream中

ZipInputStream zins = new ZipInputStream(fins);

ZipEntry ze = null;

byte[] ch = new byte[256];

while ((ze = zins.getNextEntry()) != null) {

File zfile = new File(sDestPath + ze.getName());

//File zfile = new File(sDestPath +File.separator+ ze.getName());

File fpath = new File(zfile.getParentFile().getPath());

if (ze.isDirectory()) {

if (!zfile.exists())

zfile.mkdirs();

zins.closeEntry();

} else {

if (!fpath.exists())

fpath.mkdirs();

FileOutputStream fouts = new FileOutputStream(zfile);

int i;

allFileName.add(zfile.getAbsolutePath());

while ((i = zins.read(ch)) != -1)

fouts.write(ch, 0, i);

zins.closeEntry();

fouts.close();

}

}

fins.close();

zins.close();

} catch (Exception e) {

System.err.println("Extract error:" + e.getMessage());

}

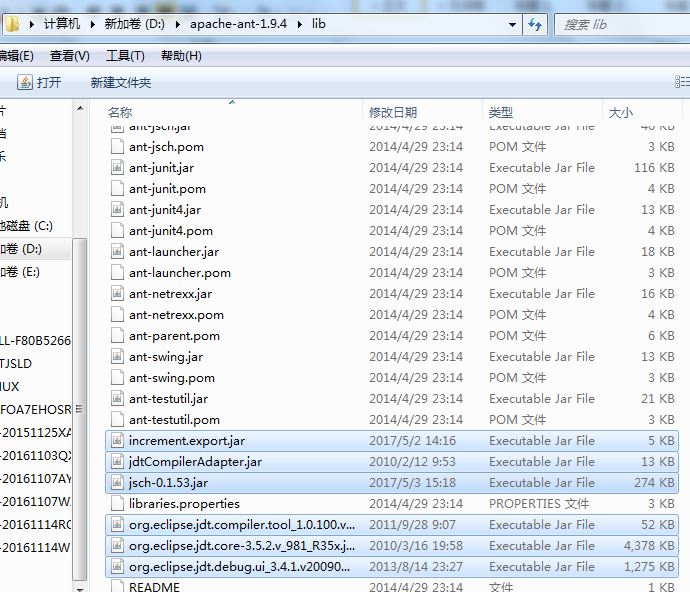
return allFileName;

}

}

**8、下载一个远程访问的一个jar包：jsch-0.1.53.jar放在lib下**

**完成6、7、8步骤后，lib目录如图所示：**



**9、将increment.export.jar配置到全局变量中去：**

**CLASSPATH ：D:\apache-ant-1.9.4\lib\increment.export.jar;.;%JAVA\_HOME%\lib\dt.jar;**

**10、脚本使用：**

1. ***编写build.xml(部署的主文件)***
2. ***编写build.properties(配置文件，使用前，详细查看配置文件的配置信息及说明)***

脚本运行：

在cmd的dos窗口输入： 注意大小写

Source /etc/profile 刷新资源

ant –f D:/build.xml inc -Dsvn.old\_revision=475 -Dsvn.new\_revision=665 war包 pentaho 一次性全部更新

123:

ant -f /home/chenyou/build.xml IncWar -Dsvn.old\_revision=665 -Dsvn.new\_revision=669

ant –f D:/build.xml IncWar -Dsvn.old\_revision=475 -Dsvn.new\_revision=665 只对war包更新

ant –f/home/chenyou/build.xml pentaho 只对pentaho更新

ant –f /home/chenyou/build.xml proUpdate 只对存储过程更新（只更新124的cntest\_fxpt到126的cntest\_fxpt\_wh）

特别注意：当

Dsvn.new\_revision，Dsvn.old\_revision 不输入时，会默认去读配置文件build.properties的值

**11、有待优化：**

项目中的jar包需要手动维护：例如，新加的jar包必须手动加到war内，SVN上没有jar包的管理

/home/chenyou/projectJAR/lib 123此目录也要手动加war

**12、关于pentaho的脚本上传和下载脚本：sync\_auto.sh**

此文件要放到pentaho环境下执行（pentaho环境的安装和配置略）

脚本路径采用的硬编码方式：

/home/pentaho/biserver-ce-5.3.0.0-213/biserver-ce/sync\_auto.sh pentaho必须安装在此路径