EBook Gratis

APRENDIZAJE

ant

Free unaffiliated eBook created from
Stack Overflow contributors.

#ant

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: ant

It is an unofficial and free ant ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official ant.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con hormigas

Observaciones

Esta sección proporciona una descripción general de lo que es Ant, y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema importante dentro de Ant y vincular a los temas relacionados. Dado que la Documentación para Ant es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Versiones mínimas de Java

Varias versiones de Ant requieren diferentes versiones del tiempo de ejecución de Java (JRE) para poder ejecutarse.

Versión de hormiga	Versión mínima de Java
1.1 hasta 1.5.4	1.1
1.6.x lanzamientos	1.2
1.7.x lanzamientos	1.3
Lanzamientos 1.8.x	1.4
1.9.x lanzamientos	1.5
1.10.x lanzamientos	1.8

Versiones

Versión	Fecha de lanzamiento
1.4.1	2001-10-11
1.5.0	2002-07-10
1.5.1	2002-10-03
1.5.2	2003-03-03
1.5.3	2003-04-09
1.5.4	2003-08-12

Versión	Fecha de lanzamiento
1.6.0	2003-12-18
1.6.1	2004-02-12
1.6.2	2004-07-16
1.6.3	2005-04-28
1.6.4	2005-05-19
1.6.5	2005-06-02
1.7.0	2006-12-13
1.7.1	2008-07-09
1.8.0	2010-02-02
1.8.1	2010-04-30
1.8.2	2010-12-20
1.8.3	2012-03-13
1.8.4	2012-05-23
1.9.0	2013-03-10
1.9.1	2013-05-22
1.9.2	2013-07-12
1.9.3	2013-12-29
1.9.4	2014-04-30
1.9.5	2015-06-03
1.9.6	2015-07-02
1.9.7	2016-04-12
1.9.8	2016-12-31
1.9.9	2017-02-06
1.10.0	2016-12-31
1.10.1	2017-02-06

Examples

Hola Mundo

Agregue lo siguiente a un archivo llamado `build.xml` en su directorio de proyecto:

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="HelloWorld" default="main">
    <target name="main" description="this is target main">
        <echo message="Hello World" />
    </target>
</project>
```

Desde un indicador de comandos en una computadora que ejecuta Windows, la ejecución de `ant main` se mostrará similar a la siguiente:

```
$ ant main
Buildfile: C:\Users\<me>\Projects\HelloWorld\build.xml

main:
     [echo] Hello World

BUILD SUCCESSFUL
```

Además, el usuario ahora puede ejecutar el comando `ant` como el nombre de destino `default` agregado al proyecto. Cuando se ejecuta el comando `ant` , busca el objetivo `default` del proyecto y lo ejecuta.

```
$ ant
Buildfile: C:\Users\<me>\Projects\HelloWorld\build.xml

main:
     [echo] Hello World

BUILD SUCCESSFUL
```

Si el script de compilación está escrito por otra persona y al usuario final le gusta ver qué destino puede ejecutar, ejecute el comando que mostrará todos los destinos que tienen descripciones.

```
$ ant -p
```

Bootstrap Apache Ivy

Agregue el siguiente objetivo en su `build.xml`

```
<!-- Bootstrap ivy -->
<target name="ivy.bootstrap" description="Download Apache Ivy">

    <!-- Define the version to use -->
    <property name="ivy.version">2.4.0</property>
```

```
    <!-- Create directory if not exists -->
    <mkdir dir="${user.home}/.ant/lib" quiet="true" />

    <!-- Download it -->
    <echo message="Downloading Apache Ivy..." />
    <get dest="${user.home}/.ant/lib/ivy.jar"
src="https://repo1.maven.org/maven2/org/apache/ivy/ivy/${ivy.version}/ivy-${ivy.version}.jar"
/>
</target>
```

Después de ejecutar la tarea `ant ivy.bootstrap` , ahora podrá resolver las dependencias utilizando apache ivy.

```
<target name="ivy.resolve" description="Resolve all artifacts.">

    <!-- Define lib driectory -->
    <property name="dir.lib">lib</property>

    <!-- Create directory if not exists -->
    <mkdir dir="${dir.lib}" />

    <!-- Configure -->
    <property name="ivy.dep.file" value="ivy.xml" />
    <ivy:settings file="ivysettings.xml" />

    <!-- Retrieve to a defined pattern -->
    <echo message="Resolving dependencies..." />
    <ivy:retrieve pattern="${dir.lib}/[artifact](-[classifier]).[ext]" />
</target>
```

Define tus recursos en `ivy.xml`

```
<ivy-module version="2.0">
        <info organisation="org.apache" module="java-build-tools"/>
        <dependencies>
                <dependency org="junit" name="junit" rev="4.11" />
                <dependency org="org.apache.commons" name="commons-compress" rev="1.9" />
        </dependencies>
</ivy-module>
```

Y cualquier repositorio personalizado en `ivysettings.xml`

```
<ivysettings>
    <settings defaultResolver="chain"/>
    <resolvers>
        <chain name="chain">
            <ibiblio name="central" m2compatible="true"/>
            <ibiblio name="github" m2compatible="true" root="http://github.com/"/>
        </chain>
    </resolvers>
</ivysettings>
```

Descarga tus dependencias ejecutando `ant ivy.resolve` .

Imprimir información del entorno antes de construir

Lo siguiente es útil tener en los registros de compilación que identifican la máquina de compilación y algunos parámetros; simplemente haga que su tarea `main` dependa de esta tarea para imprimirla antes de cada compilación.

```
<!-- Print Environment Info -->
<target name="environment">

        <!-- Get the current timestamp -->
        <tstamp>
                <format property="TODAY_UK" pattern="yyyy-MM-dd HH:mm:ss:sss zzz"
locale="cn,CN" />
        </tstamp>

        <!-- Get the hostname of the system -->
        <exec executable="hostname" outputproperty="os.hostname" />

        <!-- Print a bunch of information -->
        <echo message="" />
        <echo message="  Build Information" />
        <echo message="" />
        <echo message="  OS Information" />
        <echo message="" />
        <echo message="    User       : ${user.name}" />
        <echo message="    Hostname   : ${os.hostname}" />
        <echo message="" />
        <echo message="    Name       : ${os.name}" />
        <echo message="    Version    : ${os.version}" />
        <echo message="    Build      : ${os.arch}" />
        <echo message="" />
        <echo message="" />
        <echo message="  Java Information" />
        <echo message="" />
        <echo message="    Version    : ${ant.java.version} / ${java.version}" />
        <echo message="    Java Home  : ${java.home}" />
        <echo message="" />
        <echo message="" />
        <echo message="  Current Time : ${TODAY_UK}" />
        <echo message="" />
</target>
```

Esto dará lugar a la siguiente salida,

```
environment:
     [echo]
     [echo]   Build Information
     [echo]
     [echo]   OS Information
     [echo]
     [echo]     User       : <User Name>
     [echo]     Hostname   : <Host Name>
     [echo]
     [echo]     Name       : Windows 8.1
     [echo]     Version    : 6.3
     [echo]     Build      : amd64
     [echo]
     [echo]
     [echo]   Java Information
     [echo]
     [echo]     Version    : 1.8 / 1.8.0_45
```

```
     [echo]     Java Home  : C:\Program Files\Java\jdk1.8.0_45\jre
     [echo]
     [echo]
     [echo]   Current Time : 2016-04-18 00:40:11:011 EDT
```

Ejecutar JUnit

Lo siguiente ejecutará JUnit en las pruebas que coincidan con `test/**/*Test.java` . Esto necesita el junit.jar para estar en la carpeta `lib` .

```
<project name="Project" default="junit" basedir=".">
    <path id="classpath">
        <fileset dir="lib" includes="**/*.jar"/>
        <pathelement path="build"/>
    </path>

    <target name="compile">
        <javac srcdir="test" destdir="build" classpathref="classpath"/>
    </target>

    <target name="junit" depends="compile">
        <junit fork="true" logfailedtests="false">
            <classpath refid="classpath"/>
            <batchtest>
                <fileset dir="test" includes="**/*Test.java"/>
                <formatter type="plain" usefile="false"/>
            </batchtest>
        </junit>
    </target>
</project>
```

Crear paquete jar

Lo siguiente creará `dist/output.jar` desde el código fuente en `src` y las bibliotecas en `lib` , y utilizará `src/Main.java` como la clase principal.

```
<project name="Project" default="main" basedir=".">

    <property name="src.dir"     value="src"/>
    <property name="build.dir"   value="build"/>
    <property name="dist.dir"    value="dist"/>

    <path id="classpath">
        <fileset dir="lib" includes="**/*.jar"/>
        <pathelement path="${build.dir}"/>
    </path>

    <target name="clean">
      <delete dir="${build.dir}"/>
      <delete dir="${dist.dir}"/>
    </target>


    <target name="compile">
        <mkdir dir="${build.dir}"/>
        <javac srcdir="${src.dir}" destdir="${build.dir}" classpathref="classpath"/>
```

```
        <copy todir="${build.dir}">
            <fileset dir="${src.dir}" excludes="**/*.java"/>
        </copy>
    </target>

    <target name="jar" depends="compile">
        <mkdir dir="${dist.dir}"/>
        <jar destfile="${dist.dir}/${ant.project.name}.jar" basedir="${build.dir}">
            <fileset dir="${build.dir}"/>
            <restrict>
                <archives>
                    <zips>
                        <fileset dir="lib" includes="**/*.jar"/>
                    </zips>
                </archives>
            </restrict>
            <manifest>
                <attribute name="Main-Class" value="Main"/>
            </manifest>
        </jar>
    </target>

    <target name="main" depends="clean,jar"/>
</project>
```

Instalación o configuración

Instalar Ant es muy simple. Siga los pasos que se indican a continuación para instalar Ant en la plataforma de Windows:

1. Descarga la última versión de hormigas del sitio web de Apache
2. Descomprima el archivo en su máquina.
3. Establecer ANT_HOME en variables de entorno
4. Agregue% ANT_HOME% \ bin a su variable de entorno PATH.
5. Establezca CLASSPATH =% ANT_HOME% \ lib;% CLASSPATH%
6. Ahora abre el símbolo del sistema e ingresa el comando `ant` . Deberías ver a continuación:

```
Buildfile: build.xml does not exist!
Build failed
```

Alternativamente, utilizando Homebrew en macOS o Linuxbrew en Linux, simplemente puede ejecutar: `brew install ant`

Al usar brew no es necesario configurar las variables de entorno.

Varias distribuciones de Linux también admiten la instalación de Ant desde sus respectivos administradores de paquetes.

Para probar que Ant está instalado correctamente, navegue hasta el símbolo del sistema y

ejecute

```
ant -version
```

Este comando imprimirá la versión de Ant y también muestra que Ant se instaló correctamente.

La página de instrucciones de instalación de Ant está disponible en el sitio web de Apache Ant .

Lea Empezando con hormigas en línea: https://riptutorial.com/es/ant/topic/4223/empezando-con-hormigas

Capítulo 2: Basic File IO

Examples

Escribe un archivo usando eco

Simplemente especificando un destino de *archivo* , `echo` creará, escribirá o agregará a un archivo.

```
<echo file=example.txt" append="false">
    hello world
</echo>
```

Imprime el contenido de un archivo.

Para imprimir el contenido de un archivo, podemos usar `loadfile` para leer un archivo en una propiedad local, y luego usar `echo` para imprimir su valor.

```
<loadfile property="contents" srcFile="example.txt" />
<echo message="${contents}" />
```

Lea Basic File IO en línea: https://riptutorial.com/es/ant/topic/5932/basic-file-io

Capítulo 3: Propiedades de hormigas

Introducción

Las propiedades son pares clave-valor donde Apache Ant intenta expandir $ {clave} para valorar en tiempo de ejecución.

Las propiedades de Ant son muy útiles si tiene que hacer mucho para procesar para crear instalables o hacer implementaciones personalizadas, etc.

Por ejemplo, puede marcar $ {src.dir} como directorio de código fuente, $ {lib.dir} como biblioteca para el proyecto, $ {javadoc.dir} para javadocs, etc.

En lugar de escribir la ruta completa en cada ubicación, puede referirlos por este titular de lugar.

Examples

Cómo declarar y usar la propiedad en Ant.

Ant proporciona algunas propiedades integradas

Nombre de la propiedad	Valor
basedir	El camino absoluto de la base del proyecto.
ant.file	La ruta absoluta del archivo de compilación.
ant.version	la versión de ant
ant.project.default-target	el nombre del objetivo predeterminado del proyecto que se ejecuta actualmente
ant.project.name	nombre del proyecto
ant.java.version	Se detectó la versión de JVM.

En este ejemplo, crearemos propiedades ant personalizadas y las utilizaremos para crear un directorio temporal y copiar un archivo en él.

1. **Propiedades declaradas dentro del mismo archivo.**

```
<project name="Test Project for Ant" default="init">
    <property name="temp.dir" value="${basedir}/temp" />

    <target name="init" description="initialize">
        <mkdir dir="${temp.dir}" />
```

```
        <copy file="${basedir}/test.xml" todir="${temp.dir}/" />
    </target>
</project>
```

En Ant, $ {basedir} se referirá a la ubicación base o la ubicación donde está presente su archivo ant. Aquí declaré una propiedad nombrada como

temp.dir

que se referirá a la ubicación basedir / temp.

Por lo tanto, llamamos target init que reemplazará el marcador de posición $ {temp.dir} con su valor real y comenzará a ejecutar nuestro script. Este destino creará un directorio llamado temp en el directorio base copy copie el archivo test.xml al directorio temp.

2. **Propiedades declaradas en diferentes archivos.**

En este ejemplo, haremos referencia a las propiedades declaradas en un archivo diferente. Este es un archivo de muestra (app_version.xml) que contiene la versión de la aplicación.

```
<project name="Project Properties">
     <property name="app.version" value="1.0" />
</project>
```

Para incluir este archivo, agregaremos la tarea ant de importación para importar este archivo mientras ejecutamos los objetivos ant.

```
<import file="app_version.xml" />
```

El código de arriba se verá como

```
<project name="Test Project for Ant" default="init">
<import file="app_version.xml" />
<property name="temp.dir" value="${basedir}/temp" />

<target name="init" description="initialize">
    <mkdir dir="${temp.dir}" />
    <copy file="${basedir}/test.xml" todir="${temp.dir}/" />
    <echo message="App version is:${app.version}" />
</target>
```

Una vez que se importa el archivo, se puede acceder directamente a través del nombre de la propiedad (app.version).

Usé el archivo .xml, el mismo caso de uso también funcionará para los archivos .properties.

Lea Propiedades de hormigas en línea: https://riptutorial.com/es/ant/topic/9181/propiedades-de-hormigas

Capítulo 4: Tareas de hormigas para Git

Examples

Empezar:

```
<macrodef name = "git">
    <attribute name = "command" />
    <attribute name = "dir" default = "" />
    <element name = "args" optional = "true" />
    <sequential>
        <echo message = "git @{command}" />
        <exec executable = "git" dir = "@{dir}">
            <arg value = "@{command}" />
            <args/>
        </exec>
    </sequential>
</macrodef>
<macrodef name = "git-clone-pull">
    <attribute name = "repository" />
    <attribute name = "dest" />
    <sequential>
        <git command = "clone">
            <args>
                <arg value = "@{repository}" />
                <arg value = "@{dest}" />
            </args>
        </git>
        <git command = "pull" dir = "@{dest}" />
    </sequential>
</macrodef>
```

Clon

```
<git command="clone">
    <args>
        <arg value = "-v" />
        <arg value = "git@YOURGITURL:GITUSER/GITREPO" />
        <arg value = "repo" />
    </args>
</git>
```

Halar

```
<git command = "pull" dir = "repository_path" />
```

Añadir / Cometer / Empujar

```
<input message="Commit message" addproperty="commit-message" />
<git command="add">
    <args>
```

```
                    <arg value="." />
    </args>
</git>
<git command="commit">
    <args>
                    <arg value="-am ${commit-message}" />
    </args>
</git>
<git command="push" />
```

Lea Tareas de hormigas para Git en línea: https://riptutorial.com/es/ant/topic/7893/tareas-de-hormigas-para-git

Creditos

S. No	Capítulos	Contributors
1	Empezando con hormigas	Chad Nouis, Community, fgb, Josh Doug, Lucas Oliveira, Matt Clark, Maverick, Rao, Squidward
2	Basic File IO	Matt Clark
3	Propiedades de hormigas	Maverick
4	Tareas de hormigas para Git	Lucas Oliveira