

<http://membres-liglab.imag.fr/donsez/cours>

Apache Ant

Didier DONSEZ

Université Joseph Fourier

PolyTech'Grenoble LIG/ADELE

`Didier.Donsez@imag.fr`,

`Didier.Donsez@ieee.org`



Licence

- Cette présentation est couverte par le contrat Creative Commons By NC ND
 - <http://creativecommons.org/licenses/by-nc-nd/2.0/fr/>

Au sommaire

- Motivation
- Installation
- Structure d'un projet ANT
- Principales tâches
- Développement d'une tâche
- Bibliographie et Webographie

ANT

- Séquenceur de travaux
 - Graphe de dépendance de « cibles »
- Remplaçant de make

- Un des projets Jakarta/Apache
- Ecrit en Java
- Licence Open Source
- Intégré à de nombreux IDE java
 - Jbuilder, NetBeans Forte for Java, VisualAge for Java, Eclipse, Jedit, ...

Points forts versus Points faibles

■ Points forts

- Portable (path, séparateurs, commandes Unix et Win32)
- Rapide (une seule VM utilisée)
- Très nombreuses tâches déjà implémentées
- Largement répandu et intégré aux IDE (Eclipse, ...)
- Syntaxe rigoureuse d'XML (document bien formé)

■ Points faibles

- Verbeux (XML oblige)
- Pas de DTD !
- Pas de lancement de « tâches » interactives !!
- Dépendance de tâches (non temporelles)
- Java-Centric tout de même !!
 - Bien que <http://nant.sourceforge.net/>

Installation

- Télécharger la distribution binaire
 - <http://ant.apache.org>
- Dézipper dans un répertoire
- Positionner les variables d'environnement
 - `set JAVA_HOME=c:\j2sdk1.x.y`
 - `set ANT_HOME=c:\apache-ant-1.z.w`
 - `set PATH=%JAVA_HOME%\bin;%ANT_HOME%\bin`
- Ajouter des bibliothèques pour les taches dépendantes
 - Dans `%ANT_HOME%\lib`
 - Exemple: `netcomponents.jar` (www.savarese.org/oro/downloads) pour les taches `<ftp>` and `<telnet>`

Évites d'être ennuyer par des VM « parasites »

Build.xml

- Définit l'enchaînement à suivre pour la construction d'un projet
- Un projet comporte des cibles (target)
 - Correspondant à des activités telles que la compilation, l'installation, l'exécution, ...
- Chaque cible est composée de tâches (task)
 - exécutées en séquence (voir en //) quand la cible est exécutée
 - dépendance avec d'autres cibles
 - Ces cibles sont exécutées au préalable
- Tâche ANT
 - Correspondent à des commandes usuelles (javac, jar, copy, ...)
 - Ecrites en Java et portable quelque soit le SE
 - Built-in (fournis avec la distribution ANT) / Propriétaire

Structure build.xml

<project>

<?xml version="1.0" encoding="UTF-8"?>

<project basedir="." default="compile" name=" Typical ANT Build ">

<!-- [définitions de property] -->

<!--[définitions de path et patternset] -->

<!--[définitions de target] -->

</project>

Structure build.xml

<property>

```
<?xml version="1.0" encoding="UTF-8"?>
<project basedir="." default="compile" name=" Typical ANT Build ">
<!-- [définitions de property ] -->
  <property name="src.dir" value="./src"/>
  <property name="build.dir" value="./build"/>
  <property name="doc.dir" value="./doc"/>
  <property name="apidoc.dir" value="{doc.dir}/api"/>
  <property file="project.properties"/>
  <property environment="env"/>
  <property name="lib.dir" value="{env.LIBDIR}"/>
<!-- [définitions de path et patternset ] -->
<!-- [définitions de target ] -->
</project>
```

Structure build.xml

<path>

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project basedir="." default="compile" name=" Typical ANT Build ">
```

```
<!-- [définitions de property ] -->
```

```
<!-- [définitions de path et patternset ] -->
```

```
  <path id="project.classpath">
```

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

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

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

```
    </fileset>
```

```
  </path>
```

```
<!-- [définitions de target ] -->
```

```
</project>
```

Structure build.xml

<target>

```
<?xml version="1.0" encoding="UTF-8"?>
<project basedir="." default="compile" name=" Typical ANT Build ">
<!-- [définitions de property ] -->
<!-- [définitions de path et patternset ] -->
<!-- [définitions de target ] -->
  <target name="init">
    <mkdir dir="${build.dir}"/>
  </target>

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

Lancement

- ant [options] [target [target2 [target3] ...]]
- Options:
 - help print this message
 - projecthelp print project help information
 - buildfile <file> use given build file (-file, -f)
 - find [<file>] search for build.xml, or file, towards the root of the filesystem
 - D<property>=<value> use value for given property
 - propertyfile <file> load all properties from file (with -D taking precedence)
 - version print the version information and exit
 - quiet be extra quiet (-q)
 - verbose be extra verbose
 - debug print debugging information
 - emacs produce logging information without adornments
 - logfile <file> write logging output to given file (-l)
 - logger <classname> the class that is to perform logging
 - listener <classname> add an instance of classname as a project listener
 - inputhandler <class> the class that will handle input requests

Exemples de lancement

- ant
- ant clean
- ant -f build4jonas322.xml
- ant -Dpassword=toto
- ant -listener org.apache.tools.ant.XmlLogger

Exemple de sortie

```
Z:\dev\proj1>ant compile
```

```
Buildfile: build.xml
```

```
init:
```

```
[mkdir] Created dir: Z:\dev\proj1\build
```

```
compile:
```

```
[javac] Compiling 2 source files to Z:\dev\proj1\build
```

```
[javac] Z:\dev\proj1\src\hello\Hello.java:44: cannot resolve symbol
```

```
[javac] symbol : method length ()
```

```
[javac] location: class java.lang.String[]
```

```
[javac]          int len = argv.length();
```

```
[javac]          ^
```

```
[javac] 1 error
```

```
BUILD FAILED
```

```
file:Z:/dev/proj1/build.xml:49: Compile failed;
```

```
see the compiler error output for details.
```

```
Total time: 6 seconds
```

Ordre d'exécution des cibles

- Exécution une et une seule fois de A puis B puis C puis D (si ant D)

```
<target name="A"/>
```

```
<target name="B" depends="A"/>
```

```
<target name="C" depends="B"/>
```

```
<target name="D" depends="C,B,A"/>
```

- Condition d'exécution if/unless

```
<target name="build.windows" if="os.is.windows"/>
```

```
<target name="build.no.windows" unless="os.is.windows"/>
```

Les tâches

- Built-in
- Optional
 - Requièrè des bibliothèques Java supplémentaires ou des outils externes (csc, vss, ...)
 - Peuvent être disponible que sur une plate-forme particulière
- Propriétaire
 - Ajoute pour les besoins du projet (jcrmic, apdutool ...)

Quelques tâches (i)

- Chargement de propriétés
 - LoadFile, LoadProperties, XmlProperty, EchoProperties
- Interaction utilisateur
 - Echo, Input, Sound, Splash
- Structures de contrôle
 - Condition, Available, Parallel, Sequential, Waitfor, Sleep, Fail, Ant, Antcall, MacroDef, Script (*JavaScript, PerlScript, VBScript, Jython, ...*)
 - *Pas de boucle !*
- Compilation/Génération
 - Javac, Javah, Rmic, Style, EjbJar, Jspc, ANTLR, JavaCC, ...
 - .NET (csc, vbc, jsharp, llasm, lldasm, WsdIToDotnet,), ...
 - Patch
- Source code management
 - CVS, VSS, ClearCase, Perforce

Quelques tâches (ii)

- Test
 - Junit, Cactus (formally J2EEUnit)
- Documentation
 - Javadoc, JUnit reporting, Mail / MimeMail, Stylebook,
- Package (Emballage)
 - Jar, Manifest, SignJar, GenKey, War, Ear, ...
 - Zip, BZip2, War, Tar, Cab, Rpm ...
- Exécution
 - Java, Exec, SQL
- Déploiement
 - Copy, Move, Chmod, Mkdir, Attrib
 - Unwar, unzip, unjar, untar, ...
 - Checksum
 - FTP, Telnet, Get, Scp, Sshexec

Quelques tâches (iii)

- Web
 - Image

Quelques tâches (iv)

- La liste à jour
 - <http://ant.apache.org/manual/tasksoverview.html>
- Il en existe des centaines d'autres
- en dehors de la distribution Apache
 - Recensées
 - <http://ant.apache.org/external.html>
 - AspectJ, J2ME, ProGuard, CheckStyle, Jmetra, Java2HTML, Doxygen ...
 - <http://ant-contrib.sourceforge.net>
 - Ou non
 - [repeat](#), [apdutool](#), [jcrmic](#), [javancs](#), [javasrc](#), [webdav...](#)
 - [AXIS tasks](#) ([wsdl2java](#), [java2wsdl](#), [admin](#))
 - [APACHE jmeter](#)
 - [Remote ANT](#) <http://sourceforge.net/projects/remoteant/>
 - exécute un projet sur un site distant (par SOAP)

Quelques tâches (v)

- <http://ant-contrib.sourceforge.net>
 - Logic Tasks
 - AntCallBack AntFetch Assert Foreach For If Outoutdate RunTarget Switch hrow TimestampSelector Trycatch
 - Network Tasks
 - HTTP Post, AntServer / RemoteAnt
 - Performance Monitoring and Tasks
 - Performance Monitor Stopwatch
 - Platform Tasks
 - Osfamily Shellsript
 - Property Tasks
 - Math Propertycopy PropertySelector Pathtofileset PropertyRegex SortList URLEncode Variable
 - Process Tasks
 - Forget Limit
 - Other Tasks
 - antclipse CompileWithWalls IniFile VerifyDesign

Exemples d'exécution Java

```
<java classname="hello.Hello"/>  
<arg line="didier jean-marie ioannis"/>  
</java>
```

```
<java classname="hello.Hello" fork="true" maxmemory="128m">  
  <sysproperty key="DEBUG" value="true"/>  
  <arg value="didier"/>  
  <arg value="jean-marie"/>  
  <arg value="ioannis"/>  
  <jvmarg value="-Xrunhprof:cpu=samples,file=log.txt,depth=3"/>  
  <classpath>  
    <pathelement location="hello.jar"/>  
    <pathelement path="{java.class.path}"/>  
  </classpath>  
</java>
```

Exemples d'initialisation avec vérification des propriétés

```
<target name="init">
  <available property="server.ok" classname="com.mycomp.server.HTTPDServer"
    classpath="{server.classpath}"/>
  <available property="setup.done" file="{server.conf.dir}/conf/server.xml"/>
</target>

<target name="check server" unless="server.ok">
  <fail message="{line.separator}Configure the server classpath."/>
</target>

<target name="check setup" unless="setup.ok">
  <fail message="{line.separator}Setup your server.xml configuration file."/>
</target>

<target name="run" depends="init, check server, check setup">
  ...
</target>
```

Exemple de structure de contrôle

```
<target name="confirm.deletion">
  <input message="All data is going to be deleted (y/n)?"
    validargs="y,n" addproperty="do.delete" />
  <condition property="do.abort">
    <equals arg1="n" arg2="{do.delete}" />
  </condition>
  <fail if="do.abort">Build aborted by user.</fail>
</target>
```


Exemple de InputHandler

■ InputHandler

■ Abstrait les entrées et sorties

- `import org.apache.tools.ant.input.*;`
- ...
- `project.setInputHandler(new PropertyFileInputHandler());`

■ Exemple

■ DefaultInputHandler

- `System.out` et `System.in`

■ PropertyFileInputHandler

- `prompt` as a key to look up a value in the property file.
- The name of the file is read from the `ant.input.properties` system property.

Exemple de structure de contrôle

```
<target name="default">  
  <antcall target="doSomethingElse">  
    <param name="param1" value="value"/>  
  </antcall>  
</target>
```

```
<target name="doSomethingElse">  
  <echo message="param1=${param1}"/>  
</target>
```

Exemple avec SQL

<http://ant.apache.org/manual/CoreTasks/sql.html>

- Envoi de requêtes SQL à un SGBD via JDBC
- Exemple

```
<sql
  driver="org.hsqldb.jdbcDriver"
  url="jdbc:hsqldb:file:${database.dir}"
  userid="${dbadmin.username}"
  password="${dbadmin.password}"
  expandProperties="true"
  >
  <transaction>
    CREATE USER ${newuser} PASSWORD ${newpassword}
  </transaction>
  <transaction src="create.sql"/>
</sql>
```

Exemple avec JMX

(org.apache.catalina.ant.jmx)

```
<property name="jmx.serviceurl"
    value="service:jmx:rmi:///jndi/rmi://localhost:9999/server"/>
<jmx:open url="{jmx.serviceurl}" username="{jmx.username}"
    password="{jmx.password}" ref="my.jmx.server"/>
<jmx:query
    ref="my.jmx.server"
    name="*:*"
    resultproperty="mbeans"
    attributebinding="false"/>
<echo message="Number of MBeans in server {jmx.serviceurl} is {mbeans.Length}"/>
<echoproperties prefix="mbeans."/>
<jmx:set
    ref="my.jmx.server"
    name="java.lang:type=Threading" echo="true">
    <attribute name="ThreadContentionMonitoringEnabled" value="true" type="boolean"/>
    <attribute name="ThreadCpuTimeEnabled" value="false" type="boolean"/>
</jmx:set>
```

Exemple avec FTP

```

<project name="Mon Site Web Perso" default="transfert" basedir=".">
  <property name="src.dir" value="."/>
  <property name="remote.dir" value="/public_html"/>
  <target name="init">
    <tstamp><format property="TODAY" pattern="yyMMdd"/></tstamp>
    <echo file="{src.dir}/LAST_SITE_LOADING"="{TODAY}"></echo>
    <input message="Please enter username:" addproperty="userid"/>
    <input message="Please enter password:" addproperty="password"/>
  </target>
  <target name="transfert" depends="init,help">
    <ftp server="ftp.monhebergeur.fr" remotedir="{remote.dir}"
      userid="{userid}" password="{password}"
      depends="yes" binary="yes">
      <fileset dir="{src.dir}"/>
    </ftp>
    ...
    <mail from="me" tolist="you" subject="Results of transfer" files="build.log" />
  </target>
</project>

```

Exemple avec apply

- Exécute une commande du système

```
<apply executable="cc" dest="src/C" parallel="false">  
  <arg value="-c"/>  
  <arg value="-o"/>  
  <targetfile/>  
  <srcfile/>  
  <fileset dir="src/C" includes="*.c"/>  
  <mapper type="glob" from="*.c" to="*.o"/>  
</apply>
```

Exemple avec MacroDef

```

<macrodef name="call-cc">
  <attribute name="target"/>
  <attribute name="link"/>
  <attribute name="target.dir"/>
  <element name="cc-elements"/>
  <sequential>
    <mkdir dir="${obj.dir}/${target}"/>
    <mkdir dir="${target.dir}"/>
    <cc link="${link}"
      objdir="${obj.dir}/${target}"
      outfile="${target.dir}/${target}"
      <compiler refid="compiler.options"/>
      <cc-elements/>
    </cc>
  </sequential>
</macrodef>

```

```

<call-cc target="unittests" link="executable"
  target.dir="${build.bin.dir}">
  <cc-elements>
    <includepath location="${gen.dir}"/>
    <includepath location="test"/>
    <fileset dir="test/unittest"
      includes = "**/*.cpp"/>
    <fileset dir="${gen.dir}"
      includes = "*.cpp"/>
    <linker refid="linker-libs"/>
  </cc-elements>
</call-cc>

```

Exemple avec Script

```

<project name="squares"                generates
  default="main" basedir=".">
  <target name="setup"                 setup:
    <script language="javascript">
      <![CDATA[                       main:
        for (i=1; i<=10; i++) {       1
          echo=squares.createTask("echo"); 4
          main.addTask(echo);          9
          echo.setMessage(i*i);        16
        }                               25
      ]]> </script>                   36
    </target>                          49
    <target name="main"                64
      depends="setup"/>               81
  </project>                           100

BUILD SUCCESSFUL

```


Portabilité

- Utilisez les Path et PathElement
 - Éviter la forme `<classpath>z:\lib\my.jar;.\classes</classpath>`
- Pathconvert
 - Convertit les Path vers une chaîne en fonction d'un SE
 - Utile à la création de scripts (.bat, .sh) ou dans les tâches `<exec>`
 - Exemple

```
<pathconvert targetos="unix"
  property="project.classpath.unix" refid="project.classpath">
  <map from="${env.HOME}" to="/usr/local"/>
</pathconvert>
<echo file="run.sh" append="false" >#!/bin/sh
java -classpath ${project.classpath.unix} hello.HelloInteractive
</echo>
```

Complément sur les FileSets (i)

■ PatternSet

- Inclut/Exclut des fichiers d'un FileSet

```
<fileset dir="${server.src}" casesensitive="yes" >  
  <include name="**/*.java"/>  
  <exclude name="**/*Test*/>  
</fileset>
```

```
<fileset dir="${server.src}" casesensitive="yes" >  
  <patternset id="non.test.sources" >  
    <include name="**/*.java"/>  
    <exclude name="**/*Test*/>  
  </patternset>  
</fileset>
```

```
<fileset dir="${client.src}" >  
  <patternset refid="non.test.sources"/>  
</fileset>
```

Complément sur les FileSets (ii)

■ Mapper

- changement de noms lors des copies, déplacements, ...

```
<mapper type="flatten"/>
```

```
A.java -> A.java
```

```
foo/bar/B.java -> B.java
```

```
<mapper type="merge" to="archive.tar"/>
```

```
A.java -> archive.tar
```

```
foo/bar/B.java -> archive.tar
```

```
<mapper type="glob" from="*.java" to="*.java.bak"/>
```

```
A.java -> A.java.bak
```

```
foo/bar/B.java -> foo/bar/B.java.bak
```

```
<mapper type="regexp" from="^(.*)/([^/]+)/([^/]*)$" to="\1/\2-\3"/>
```

```
A.java ignored
```

```
foo/bar/B.java -> foo/bar/bar-B.java
```

Complément sur les FileSets (iii)

■ Selector

- Sélectionne les fichiers sur leurs noms, contenu , propriétés ...

```

<project default="default" basedir=".">
  <selector id="doctozip">
    <and>
      <size value="4" units="Ki" when="more"/>
      <date datetime="01/01/2003 12:00 AM" when="after"/>
      <or> <depth max="0"/> <filename name="*.html"/> <filename name="*.xml"/> </or>
      <majority>
        <contains text="génie logiciel" casesensitive="false"/>
        <contains text="Jakarta ANT" casesensitive="true"/>
      </majority>
    </and>
  </selector>
  <target name="default">
    <zip destfile="manual.zip" ><fileset dir="./docs"> <selector refid="doctozip"/> </fileset></zip>
  </target>
</project>

```

- Remarque: il est possible de définir des selectors custom

FilterChains et FilterReaders (i)

- Permet d'enchaîner des filtres sur les contenus échangés lors des tâches Copy, LoadFile, LoadProperties, Move
- Exemple

```
<copy file="${src.file}" tofile="${dest.file}">  
  <filterchain>  
    <filterchain>  
      <filterreader classname="org.apache.tools.ant.filters.HeadFilter">  
        <param name="lines" value="15"/>  
      </filterreader>  
      <filterreader classname="org.apache.tools.ant.filters.LineContains">  
        <param type="contains" value="main"/>  
      </filterreader>  
    </filterchain>  
  </filterchain>  
</copy>
```

- Est équivalent à la commande Unix
`cat < ${src.file} | head -n15 | grep main > ${dest.file}`

FilterChains et FilterReaders (ii)

- FilterReaders fournis dans la distribution
 - ClassConstants, ExpandProperties, **HeadFilter**, **LineContains**, LineContainsRegExp, PrefixLines, ReplaceTokens, StripJavaComments, StripLineBreaks, StripLineComments, TabsToSpaces, **TailFilter**, ...
- API
 - interface org.apache.tools.ant.filters.ChainableReader
 - classe org.apache.tools.ant.filters.BaseParamFilterReader
- Exercice : CPPFilterReader
 - réalise la substitution de macros et inclusion de fichiers comme le fait CPP

Listeners

- Reçoit des événements durant la construction
 - Build started / finished
 - Target started / finished
 - Task started / finished
 - Message logged
- Built-in: log4j et XML
- Exemple
 - ant -listener org.apache.tools.ant.XmlLogger
 - Il existe des « feuilles » XSLT pour formater le journal produit !

Création de tâches ANT personnalisées

- La classe doit
 - étendre `org.apache.tools.ant.Task`
 - méthode `execute()`
 - lève `BuildException` si besoin
 - comporter un setter pour chaque attribut (`setMessage(String msg)`)
 - manipuler les éléments imbriqués (texte, `FileSet`, ...)

- La déclaration

```
<?xml version="1.0"?>
```

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

```
  <taskdef name="myecho" classname="fr.imag.adele.ant.task.MyEchoTask"
    classpath="./myechotask.jar"/>
```

```
  <target name="main">
```

```
    <myecho message="Hello World!"/>
```

```
    <myecho>My echo task works</myecho>
```

```
  </target>
```

```
</project>
```


Création de tâches ANT personnalisées

```
package fr.imag.adele.ant.task;
import org.apache.tools.ant.BuildException;
import org.apache.tools.ant.Task;
public class MyEchoTask extends Task {
    private String msg;
    private String text;
    public void execute() throws BuildException {
        if(msg!=null) System.out.println(msg);
        if(text!=null) System.out.println(text);
    }
    public void setMessage(String msg) {
        this.msg = msg;
    }
    public void addText(String text){
        this.text=text;
    }
}
```

Ant depuis Java

- Motivation : réutilisation des tâches Ant dans des programmes Java
- Exemple

```
public class Unzip {
    public static void main (String[] args) {
        if(args.length==1) { new Unzip().unzip(args[0],".");
        } else if(args.length==2) { new Unzip().unzip(args[0],args[1]); } else { ... } }
    final class Expander extends org.apache.tools.ant.taskdefs.Expand {
        public Expander() {
            project = new Project();
            project.init();
            taskType = "unzip";
            taskName = "unzip";
            target = new Target();
        }
    }
    protected void unzip(String zipFilepath, String destinationDir) {
        Expand expand = new Expander();
        expand.setSrc(new File(zipFilepath));
        expand.setDest(new File(destinationDir));
        expand.execute();
    }
}
```

Apache Ivy <http://ant.apache.org/ivy>

- Tool for managing (recording, tracking, resolving and reporting) project dependencies.
 - not tied to any methodology or structure.
 - configurability to be adapted to a broad range of dependency management and build processes.
 - tight integration with Apache Ant
 - Ant tasks ranging from dependency resolution to dependency reporting and publication.
 - transitive dependencies management
 - configured to use a lot of different repositories

Apache Ivy <http://ant.apache.org/ivy>

■ Exemple de module

- ```
<ivy-module version="2.0">
 <info organisation="apache" module="hello-ivy"/>
 <dependencies>
 <dependency org="commons-lang" name="commons-lang" rev="2.0"/>
 <dependency org="commons-cli" name="commons-cli" rev="1.0"/>
 </dependencies>
</ivy-module>
```

# Buildr

---

- <http://incubator.apache.org/buildr/>

# Gant

<http://gant.codehaus.org/>

---

- Tool for scripting Ant tasks using Groovy instead of XML to specify a build
- Example

```
includeTargets << gant.targets.Clean
cleanPattern << ['**/*~' , '**/*.bak']
cleanDirectory << 'build'
target (stuff : 'A target to do some stuff.') {
 println ('Stuff')
 depends (clean)
 echo (message : 'A default message from Ant.')
 otherStuff ()
}
target (otherStuff : 'A target to do some other stuff') {
 println ('OtherStuff')
 echo (message : 'Another message from Ant.')
 clean ()
}
setDefaultTarget (stuff)
```
- Run the build

```
gant -p
Gant -Dfoo=bar -f build.groovy
```
- Misc

```
ant2gant, gant ant task, ivy (dependencies management), execute (shell commands and pipe)...
```

# NAnt (<http://nant.sourceforge.net/>)

- Un portage de ANT (canevas + taches) sur .NET (et Mono)
  - Taches spécifiques à .NET
    - val, asminfo, attrib , available , aximp , call , cl , copy , csc , cvs , cvs-changelog , cvs-checkout , cvs-export , cvs-pass , cvs-rtag , cvs-tag , cvs-update , delay-sign , delete , description , echo , exec , fail , foreach , get , gunzip , if , ifnot , ilasm , ildasm , include , jsc , lib , license , link , loadfile , loadtasks , mail , mc , midl , mkdir , move , nant , nantschema , ndoc , nunit , nunit2 , property , rc , readregistry , regasm , regex , regsvcs , resgen , script , servicecontroller , setenv , sleep , solution , style , sysinfo , tar , tlbexp , tlbimp , touch , tstamp , unzip , uptodate , vbc , vjc , xmlpeek , xmlpoke , zip
- Intégration dans MS Visual Studio et SharpDevelop

# NAnt : exemple de tâches personnalisés

---

```
using NAnt.Core;
using NAnt.Core.Attributes;
namespace NAnt.Examples.Tasks {
 [TaskName("myecho")]
 public class MyEchoTask : Task {
 #region Private Instance Fields
 private string _message;
 #endregion Private Instance Fields
 #region Public Instance Properties
 [TaskAttribute("message", Required=true)]
 public string Msg {
 get { return _message; }
 set { _message = value; }
 }
 #endregion Public Instance Properties
 #region Override implementation of Task
 protected override void ExecuteTask() {
 Log(Level.Info, _message);
 }
 #endregion Override implementation of Task
 }
}
```



# NAnt : exemple de projet

```

<?xml version="1.0"?>
<project name="NAnt" default="run">
 <!-- Compile the test task and add it then use it. -->
 <target name="build">
 <mkdir dir="bin" />
 <csc target="library" output="bin/MyTasks.dll">
 <sources>
 <include name="MyEchoTask.cs"/>
 </sources>
 <references basedir="{nant::get-base-directory()}">
 <include name="NAnt.Core.dll"/>
 </references>
 </csc>
 </target>
 <target name="run" depends="build">
 <!-- Dynamically load the tasks in the Task assembly. -->
 <loadtasks assembly="bin/MyTasks.dll" />
 <!-- Call our new task, converts the message attribute to all caps and displays it. -->
 <myecho message="Hello World !"/>
 </target>
 <target name="clean">
 <!-- Delete the build output. -->
 <delete file="bin/MyTasks.dll" if="{file::exists('bin/MyTasks.dll')}" />
 </target>
</project>

```

# Alternatives

---

- Apache Maven
  - <http://membres-liglab.imag.fr/donsez/cours/maven.pdf>
  
- 7Bee (Build Everything Easier)
  - TODO

# Bibliographie et Webographie

---

## ■ Web

- <http://ant.apache.org>
- <http://ant-contrib.sourceforge.net/cc.html>
- <http://ant-contrib.sourceforge.net/tasks/tasks/index.html>

## ■ Complément

- <http://www-adele.imag.fr/~donsez/cours/tpant.zip>
  - Exemples et exercices
- <http://www-adele.imag.fr/~donsez/cours/coursjavaoutil.pdf>

# Bibliographie et Webographie

## ■ Ouvrages

- Erik Hatcher, Steve Loughran, "Java Development With Ant", Ed Manning Publications Company, Août 2002, 672 pages, ISBN 1930110588
  - Un des plus détaillés
- Richard Hightower, Nicholas Lesiecki, «Java Tools for Extreme Programming: Mastering Open Source Tools, including Ant, JUnit, and Cactus», Ed Wiley, ISBN: 0-471-20708-X, November 2001
- Eric M. Burke, Brian M. Coyner « Java Extreme Programming Cookbook », Ed O'Reilly, March 2003, ISBN: 0-596-00387-0, 288 pages
- Jesse E. Tilly, Eric M. Burke, Ant: The Definitive Guide, O'Reilly, <http://www.oreilly.com/catalog/anttdg/index.html?CMP=ILL-4GV796923290>
- Glenn Niemeyer and Jeremy Poteet, Extreme Programming with Ant: Building and Deploying Java Applications with JSP, EJB, XSLT, XDoclet, and Junit, Ed SAMS, October 2003
  - [http://java.sun.com/developer/Books/javaprogramming/Poteet/Ant\\_CH08.pdf](http://java.sun.com/developer/Books/javaprogramming/Poteet/Ant_CH08.pdf)