使い方

-f でビルドファイルを指定しない場合は、 build.xml を使用する

■ターゲットを指定して実行

```
ant ターゲット名
```

■ターゲットの一覧

```
ant -p
```

■ java で実行する場合

```
java -cp ant-launcher.jar org.apache.tools.ant.launch.Launcher ターゲット名
```

基本的な build.xml

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

<project name="app"  basedir="." default="all">
  <!-- 変数 -->
  <property name="app.home" location="."/>
  <property name="app.lib"   location="${app.home}/lib"/>
  <property name="app.class" location="${app.home}/build/classes"/>
  <property name="app.src"   location="${app.home}/src"/>
  <property name="app.dist"   location="${app.home}/dist"/>

  <property name="app.jar" location="${app.home}/app.jar"/>
  <property name="app.manifest" location="${app.home}/manifest.txt"/>
  <property name="app.mainclass" value="app.Main"/>

  <property name="app.java.version" value="1.6"/>
  <property name="app.test.class" value="TestAll"/>

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

  <!-- クリーン処理 -->
  <target name="clean" >
    <delete dir="${app.class}"/>
  </target>

  <!-- リソースファイルコピー -->
  <target name="copyResource" >
    <mkdir dir="${app.class}"/>
    <copy todir="${app.class}">
      <fileset dir="${app.src}">
        <exclude name="**/*.java"/>
        <exclude name="**/*.form"/>
      </fileset>
    </copy>
  </target>

  <!-- Java のコンパイル -->
  <target name="compile" >
    <antcall target="copyResource"/>
    <mkdir dir="${app.class}"/>

     <javac includeantruntime="false" debug="true" srcdir="${app.src}" target="${app.java.version}"
source="${app.java.version}" destdir="${app.class}" encoding="UTF-8">
      <classpath refid="classpath" />
    </javac>
  </target>

  <!-- Java の実行 -->
  <target name="run" >
    <java fork="yes" classname="${app.mainclass}">
```

```
    <permissions>
       <grant class="java.security.AllPermission"/>
    </permissions>
      <classpath refid="classpath" />
    </java>
  </target>

  <!-- jar ファイルの作成 -->
  <target name="jar">
    <jar basedir="${app.class}" jarfile="${app.jar}" manifest="${app.manifest}" />
  </target>

  <!-- ドキュメント作成 -->
  <target name="javadoc" >
    <javadoc encoding="UTF-8"
      access="private" charset="UTF-8"
      docencoding="UTF-8"
      destdir="${app.dist}/javadoc">
      <sourcepath path="${app.src}" />
    </javadoc>
  </target>

  <!-- Unit-Test -->
  <target name="UnitTest" >
    <!-- JUnit によるテストを実行 -->
    <junit fork="yes" haltonfailure="yes">
      <test name="${app.test.class}" />
      <formatter type="xml" usefile="true" />
      <classpath refid="classpath" />
    </junit>

    <!--<mkdir dir="report"/>
    <junit fork="yes" haltonfailure="yes">
      <batchtest fork="yes" todir="report">
        <fileset dir="${app.src}">
          <include name="**/Test*.java"/>
          <exclude name="**/TestAll.java"/>
        </fileset>
      </batchtest>
      <formatter type="xml"/>
      <classpath refid="classpath" />
    </junit>-->

    <!--Jenkins には
    report/**/TEST-*.xml
    のように登録 -->
  </target>

  <!-- デフォルトターゲット -->
  <target name="all" depends="compile,jar">
  </target>

</project>
```

プロキシを使う

http://yoshimov.com/?page=Java%2_FAnt%A4_%CE%BD%E8_%CD%FD%A4_%C7_Proxy%A4_%F2%CD%F8%CD%D1%A4%B9%A4%EB

```
<property name="proxy.host" value="proxy.somewhere.com" />
<property name="proxy.port" value="8080" />
<setproxy proxyhost="${proxy.host}" proxyport="${proxy.port}" />
```

をテキトウな場所に入れる。

Javadoc

javadoc の ソースを指定する際に

```
<packageset dir="${dpk.src}" defaultexcludes="yes">
  <include name="**/*" />
</packageset>
```

などのようにパッケージを設定しないとうまくいかないことがある。