

تعرف على ant

الهدف من هذا المقال هو تعريف الـ ant لمطوري الجافا الذين لم يستعملوا هذه الأداة من قبل. و لتبين الفائدة المرجوة من هذه الأداة سننقد مقارنة بين مشروعين أحدهما يتم إنشاؤه و ترجمته باستعمال سطر الأوامر و الآخر باستعمال الـ Ant . متابعة متمعة J

1. مقدمة

Ant عبارة عن مشروع مفتوح المصدر مقدم من Apache مكتوب بالجافا، الهدف من كل هذا هو توفير أداة تسمح بآتممة العمليات المتكررة بكثرة طيلة المدة التي تستغرقها كتابة المشروع.

Ant متوفر للتحميل على العنوان التالي:

<http://ant.apache.org>.

لتقريب الصورة و توضيحها أكثر ، يمكن مقارنة Ant بالأداة الشهيرة make الخاصة بنظام التشغيل unix . لكن Ant موجه للعمل بعض النظر عن نظام التشغيل الذي يعمل عليه.

حجز الزاوية في ant هو ملف XML يحوي مجموعة الأوامر الواجب تنفيذها، توفر الأداة مجموعة من العمليات الشائع استعمالها (على هيئة java class) ، كما أنه من الممكن كتابة أوامر أخرى لأتممتها حسب حاجة المبرمج،

شهرة ant و قوته دفعت بمختلف IDE الخاصة بالجافا بدعمه، يمكنك إيجاد ant في مختلف الـ IDE المتوفر للـ java كـ Eclipse و Netbeans

2. تثبيت ANT:

تثبيت ANT سهل جدا حيث أنه يكفي عمل التالي:

1. فك ضغط الـ ant في مجلد (مثلا D:\apache-ant-1.7.1)
2. إنشاء متغير نظام جديد اسمه ANT_HOME حيث يكون محتواه مسار المجلد السابق.
3. إضافة %ANT_HOME% إلى المتغير path

ب. إنشاء مشروع باستعمال سطر الأوامر:

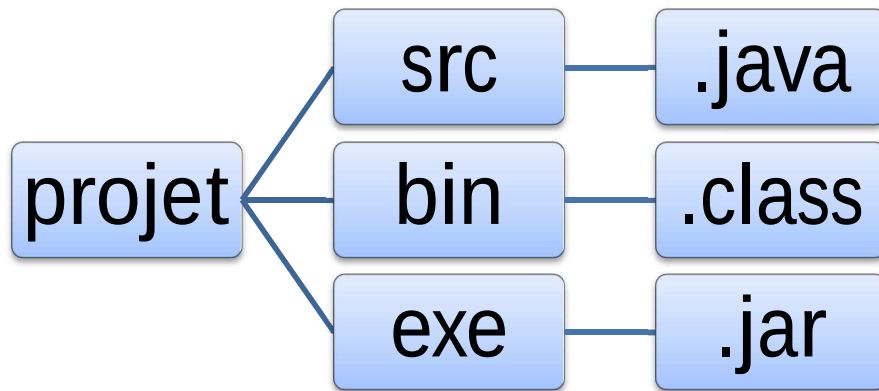
في هذا الدرس سنضع كل نوع من الملفات لوحده:

- الكود المصدري (ملفات .java) في مجلد src
- ملفات .class في مجلد bin
- ملفات .jar في مجلد exe



لإنشاء هذه المجلدات ما علينا سوى استعمال الأمر Md متبوعا باسم المجلد لإنشائه (في linux يجب استبدال الـ md بـ mkdir و الـ \ بالـ /)

سينتج لنا التشعب التالي:



ب.1. الترجمة:

ننشئ المجلد src فقط الآن باستعمال الأمر

```
Md src
```

الصنف التالي يقوم بإظهار العبارة الشهيرة Hello World في شاشة سطر الأوامر

```
package org.sdf;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world");
    }
}
```

كما هو ظاهر من اسم الـ package يجب وضع الملف HelloWorld.java الذي يحوي هذا الصنف في التشعب src/org/sdf



الآن سنقوم بترجمة البرنامج و من ثم تشغيله

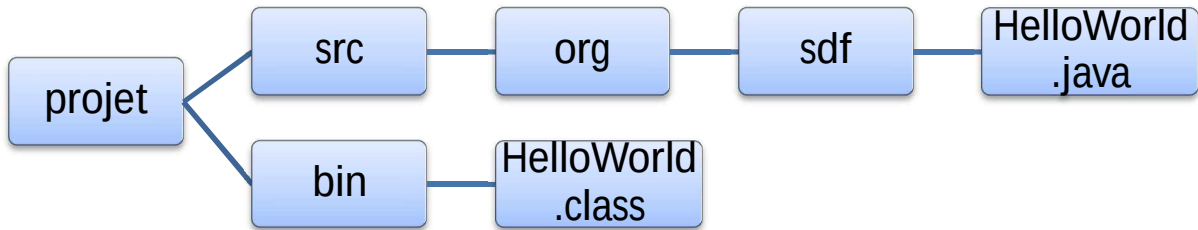
```
md bin
javac -sourcepath src -d bin\ src\org\sdf\HelloWorld.java
java -cp bin org.sdf.HelloWorld
```



ستظهر لك العبارة الشهيرة:

Hello World

لاحظ أنه صار لنا الآن التشعب التالي:



ب.2. إنشاء ملف الـ jar :

إنشاء ملف الـ jar ليس بالأمر الصعب و إنما يمر بعدة مراحل و هي كالتالي:

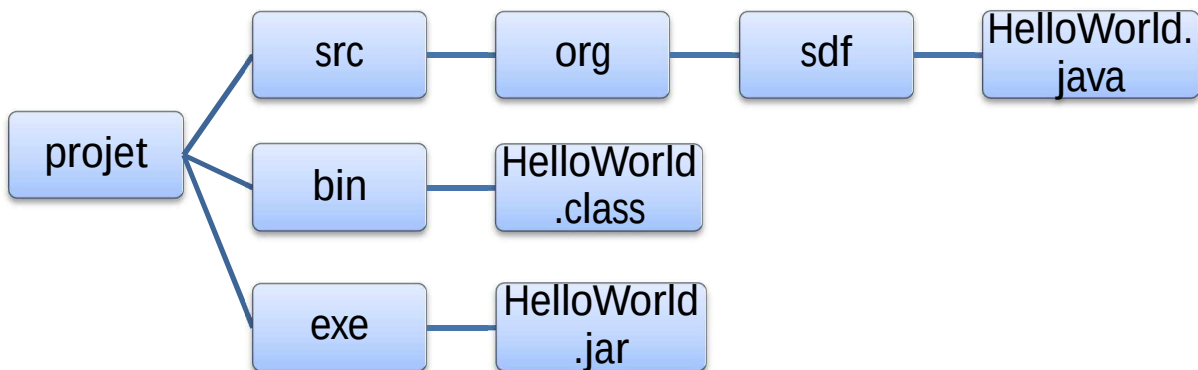
1. إنشاء ملف manifest يحتوي على اسم الصنف الرئيسي في المشروع main-class
2. إنشاء المجلد exe
3. و من ثم القيام بتوليد الـ jar

يمكن القيام بكل هذا عن طريق سطر الأوامر و ذلك كالتالي:

```

echo Main-Class: org.sdf.HelloWorld>Manifest
md exe
jar cfm exe\HelloWorld.jar Manifest -C bin .
  
```

بعد تنفيذ هذه الأوامر يصبح التشعب على النحو التالي:



ب.3. تنفيذ البرنامج:

لتنفيذ البرنامج نكتب الأمر التالي:

```
java -jar exe\HelloWorld.jar
```

ج. إنشاء المشروع باستخدام ANT

سنقوم الآن بإنشاء ملف build.xml الخاص بمشروعنا و الذي سيحوي الأهداف الأربعة التالية:

run و jar ، compile ، Clean

ج.1. الهدف clean

الغاية من هذا الهدف هو القيام بعملية "تنظيف" للمشروع و حذف كل الملفات المؤقتة و الزائدة التي نتجت بعد ترجمة المشروع.

```
<target name="clean">
    <delete dir="bin"/>
    <delete dir="exe"/>
</target>
```

كما تلاحظون فإنه تتم مناداة المهمة delete مرتين، مرة لحذف ملفات الـ class و مرة لحذف ملف الـ Jar

ج.2. الهدف compile

سنستعمل هذا الهدف لإنشاء المجلد bin و من ثم ترجمة الملفات الموجودة في المجلد src و وضع الملفات الناتجة في المجلد bin

```
<target name="compile">
    <mkdir dir="bin"/>
    <javac srcdir="src" destdir="bin"/>
</target>
```

إنشاء المجلد يتم باستخدام المهمة mkdir أما الترجمة فتتم باستخدام المهمة javac

ج.3. الهدف jar

سنستعمل هذا الهدف لإنشاء مجلد exe لنضع فيه ملف الـ jar الخاص بمشروعنا فيه

```
<target name="jar">
    <mkdir dir="exe"/>
    <jar destfile="exe/HelloWorld.jar" basedir="bin">
        <manifest>
            <attribute name="Main-Class"
value="org.sdf.HelloWorld"/>
        </manifest>
    </jar>
</target>
```



إنشاء المجلد يتم بواسطة المهمة `mkdir` أما إنشاء الـ `jar` فيتم باستعمال المهمة `jar`

ج.4. الهدف `run`:

نستعمل هذا الهدف لتشغيل ملف الـ `jar` الذي تم إنشاؤه في الخطوة السابقة

```
<target name="run">
  <java jar="exe/HelloWorld.jar" fork="true"/>
</target>
```

تستعمل المهمة `java` لتشغيل ملف الـ `jar` أما الـ `fork="true"` فالغرض منها تشغيل المشروع في JVM جديدة

ج.5. ترتيب المهمات:

من الواضح جدا أن بعض المهمات السابقة يجب أن تنفذ قبل أخرى أو أن بعض المهمات تحتاج إلى نتائج مهمة أخرى، فمثلا لا يمكن تشغيل ملف `jar` إذا لم تكن لدينا ملفات `class`. و لا يمكن أن تكون هناك ملفات `class`. إذا لم نقم بترجمة المشروع ...

يوفر `ant` وسيلة لتبيين الترابط الموجود بين المهمات و ذلك عن طريق الـ `attribute` المسمى `depends`

المثال التالي يوضح كيف أنه يجب تنفيذ المهمة `compile` قبل المهمة `jar`

```
<target name="jar" depends="compile">
  <mkdir dir="exe"/>
  <jar destfile="exe/HelloWorld.jar" basedir="bin">
    <manifest>
      <attribute name="Main-Class"
value="org.sdf.HelloWorld"/>
    </manifest>
  </jar>
</target>
```

و الآن بعد أن بيننا كل مهمة لوحدها فإن مجموع هذه المهمات يصبح على الشكل التالي:

```
<project default="run">
  <target name="clean">
    <delete dir="bin"/>
    <delete dir="exe"/>
  </target>

  <target name="compile" depends="clean">
    <mkdir dir="bin"/>
    <javac srcdir="src" destdir="bin"/>
  </target>
</project>
```



```

</target>

<target name="jar" depends="compile">
    <mkdir dir="exe"/>
    <jar destfile="exe/HelloWorld.jar" basedir="bin">
        <manifest>
            <attribute name="Main-Class"
value="org.sdf.HelloWorld"/>
        </manifest>
    </jar>
</target>

<target name="run" depends="jar">
    <java jar="exe/HelloWorld.jar" fork="true"/>
</target>

</project>

```

و عليه فإنه من الممكن تشغيل المشروع عن طريق كتابة الأوامر التالية:

```

ant clean
ant compile
ant jar
ant run

```

و بما أننا قد حددنا الترابط الموجود بين كل المهمات فإنه يمكن أيضا كتابة الأمر التالي فقط

```
ant run
```

أو حتى الأمر التالي فقط

```
ant
```

بحكم أننا تمرنا لل attribut المسمى default الخاص بالمشروع القيمة run و الذي معناه أنه يجب على ant تنفيذ الهدف run إذا لم يتم تحديد أي هدف لتنفيذه

ج.6. تحسين ملف الـ build.xml:

لا شك أنكم قد لاحظتم أنه تم استعمال نفس أسماء المجلدات أكثر من مرة في نفس ملف build.xml ماذا لو أردنا تغيير أسماء المجلدات ، بلا شك يتوجب علينا إعادة تسمية المجلدات في كل الملف، من أجل هذا يوفر ant إمكانية استعمال الـ properties و إرفاق قيم بها.

كما أنه من الممكن معرفة ما هي العمليات التي يقوم بها ant لحظة وقوعها و ذلك باستعمال المهمة echo



```

<project default="run">

<property name="src.dir"      value="src"/>
<property name="bin.dir"      value="bin"/>
<property name="jar.dir"      value="exe"/>
<property name="main-class"   value="org.sdf.HelloWorld"/>

<target name="clean">
<delete dir="${bin.dir}"/>
<delete dir="${jar.dir}"/>
<echo message="clean is done"/>
</target>

<target name="compile" depends="clean">
<mkdir dir="${bin.dir}"/>
<javac srcdir="${src.dir}" destdir="${bin.dir}"/>
<echo message="compile is done"/>
</target>

<target name="jar" depends="compile">
<mkdir dir="${jar.dir}"/>
    <jar destfile="${jar.dir}/sdf.jar" basedir="${bin.dir}">
        <manifest>
            <attribute name="Main-Class" value="${main-class}"/>
        </manifest>
    </jar>
<echo message="creating JAR file"/>
</target>

<target name="run" depends="jar">
<java jar="${jar.dir}/sdf.jar" fork="true"/>
</target>

</project>

```

د. مقارنة بين ANT و سطر الأوامر:

يمكن عقد مقارنة بسيطة بين ANT و سطر الأوامر على النحو التالي:

Command ligne	build.xml
md bin javac -sourcepath src -d bin\ src\org\sdf\HelloWorld.java	<target name="compile"> <mkdir dir="bin"/> <javac srcdir="src" destdir="bin"/> </target>
echo Main-Class: org.sdf.HelloWorld>Manifest md exe jar cfm exe\HelloWorld.jar MonManifest - C bin .	<target name="jar"> <mkdir dir="exe"/> <jar destfile="exe/HelloWorld.jar" basedir="bin"> <manifest> <attribute name="Main-Class" value="org.sdf.HelloWorld"/> </manifest>



	<pre></jar> </target></pre>
java -jar exe\HelloWorld.jar	<pre><target name="run"> <java jar="exe/HelloWorld.jar" fork="true"/> </target></pre>

هـ. خاتمة:

يوفر ANT خدمات للمبرمج تجنبه تكرير نفس الأعمال و المهمات الروتينية على نفس المشروع مما يزيد من إنتاجيته، استعمال ant لملفات xml يجعل ملف build.xml واضح و سهل القراءة

هناك أدوات أخرى تسمح بأتمتة العمليات على مشاريع الجافا نذكر على سبيل المثال الأداة maven

عنوان المقال الأصلي:
Prise en main d'Ant

رابط المقال الأصلي:
[/http://skebir.developpez.com/tutoriels/java/ant](http://skebir.developpez.com/tutoriels/java/ant)

كاتب المقال الأصلي:
سليم كبير

الترجمة (بتصرف): Djug

تمت الترجمة بعون الله في 08.07.2009

