LAPORAN PRAKTIKUM
SOFTWARE QUALITY ASSURANCE

PERTEMUAN KE 4

Disusun Oleh :

NAMA	**:**	**MARFIANA AYU IRAWATI**
NIM	**:**	**235611053**
JURUSAN	**:**	**SISTEM INFORMASI**
JENJANG	**:**	**S1**

Universitas Teknologi Digital Indonesia

2024

PERTEMUAN 4
OTOMATISASI PROSES BUILD DENGAN APACHE ANT
(PENGUJIAN DENGAN *TOOLS*)

I. PENDAHULUAN DAN DASAR TEORI

Dalam pengembangan perangkat lunak, proses build merupakan tahap penting yang memastikan kode sumber dapat dikompilasi, diuji, dan dihasilkan menjadi aplikasi yang dapat dijalankan. Dengan meningkatnya kompleksitas proyek perangkat lunak, manual build process sering kali menjadi tidak efisien dan rentan terhadap kesalahan. Oleh karena itu, otomatisasi proses build menjadi sangat penting. Salah satu alat yang sering digunakan untuk tujuan ini adalah Apache Ant.

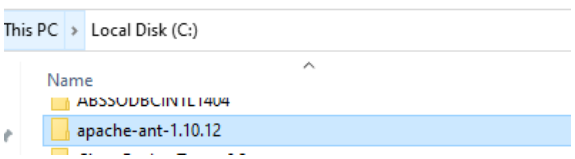
Apache Ant adalah alat otomatisasi yang ditulis dalam bahasa pemrograman Java. Alat ini memungkinkan pengembang untuk mendefinisikan proses build dalam bentuk file XML, sehingga memudahkan untuk mengatur, menjalankan, dan memelihara proses build. Selain itu, Ant mendukung berbagai tugas seperti kompilasi, pengujian, dan pengemasan, sehingga memungkinkan integrasi dengan alat pengujian lainnya.

Otomatisasi proses build dengan Apache Ant memberikan banyak keuntungan dalam pengembangan perangkat lunak, mulai dari pengurangan kesalahan manual hingga peningkatan efisiensi. Dengan integrasi alat pengujian, pengembang dapat memastikan bahwa aplikasi yang dibangun memenuhi standar kualitas yang diinginkan. Penggunaan Apache Ant dalam proses build tidak hanya menyederhanakan pengelolaan proyek tetapi juga membantu dalam mengoptimalkan siklus pengembangan perangkat lunak secara keseluruhan.

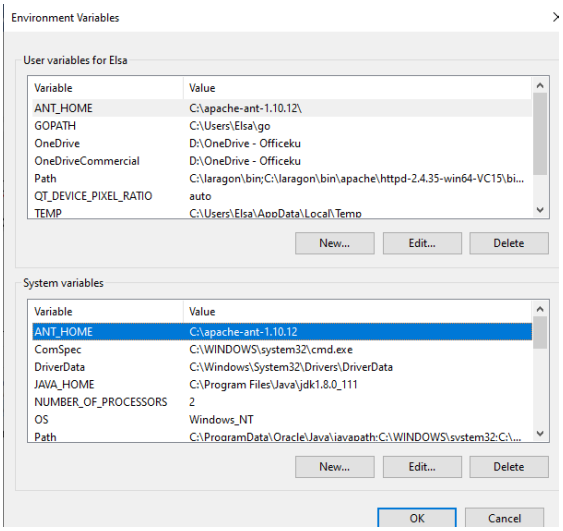
II. PEMBAHASAN

Praktik :

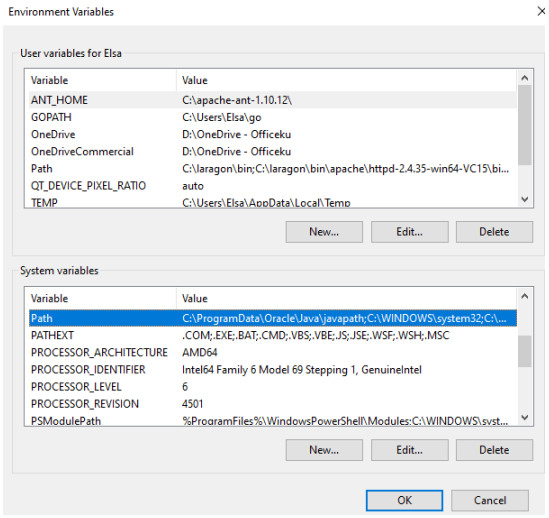
1. **Instalasi ANT, yaitu :**
   Download file master apache ant, lalu ekstrak file di drive C, seperti gambar berikut.



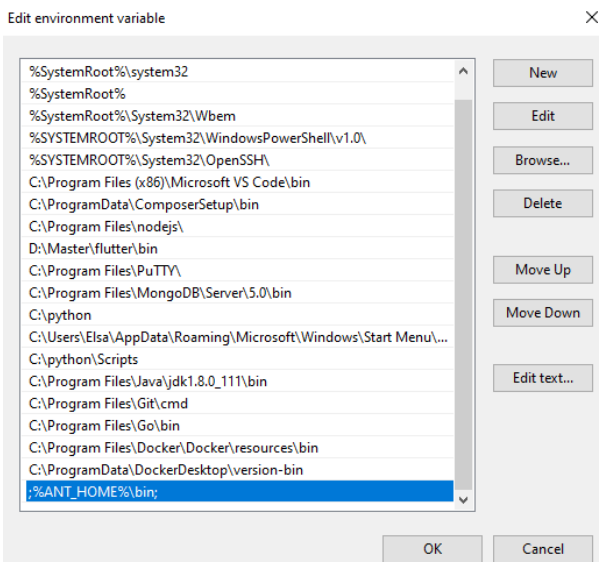
2. **Konfigurasi ANT_HOME dan PATH di Environment**
   - Buka Advanced system settings, lalu tambahkan system variable ANT_HOME



   - Double klik Path



   - Tambahkan “ ;%ANT_HOME%\bin; “ di path



   - Klik ok
   - Proses setting konfigurasi selesai

3. **Testing**
   - Buka cmd, arahkan path ke folder **C:\apache-ant-1.10.12\bin**

```
C:\>cd apache-ant-1.10.12

C:\apache-ant-1.10.12>
```

   - Cek versi ANT yang telah terinstal, dengan perintah : **ant -version**

```
C:\apache-ant-1.10.12\bin>ant -version
Apache Ant(TM) version 1.10.12 compiled on October 13 2021
```

   - Tandanya sudah sukses dan berhasil terinstall ANT

Soal Latihan :

Gantilah HelloWorld.java dengan program untuk menampilkan hasil perkalian 20*150 dan nama kelas Perkalian.

1. Mengganti Nama program HelloWorld.Java menjadi Perkalian.java
2. Ubah script menjadi seperti ini di file Perkalian.java

bin > src > simpleant > Perkalian.java

```java
package simpleant;


public class Perkalian{
    public static void main(String[] args){
        int kali=20*150;
        System.out.println("Hasil Perkalian dari 20 dan 150 adalah " +kali);
    }
}
```

3. Buka file build.xml, lalu ubah kodenya menjadi seperti gambar di bawah ini

bin > build.xml

```xml
<project>
<target name="clean">
<delete dir="build"/>
</target>
<target name="compile">
<mkdir dir="build/classes"/>
<javac srcdir="src" destdir="build/classes"/>
</target>
<target name="jar" >
<mkdir dir="build/jar"/>
<jar destfile="build/jar/Perkalian.jar" basedir="build/classes">
<manifest>
<attribute name="Main-Class" value="simpleant.Perkalian"/>
</manifest>
</jar>
</target>
<target name="run">
<java jar="build/jar/Perkalian.jar" fork="true"/>
</target>
</project>
```

Pembahasan :

Agar file build.xml ketika di compile tidak terjadi eror maka kita perlu mengubah nama kelas **HelloWord** menjadi nama kelas dengan kelas **Perkalian.**

4. Melakukan compile program build, jar dan run

1) Perintah **ant clean**

```
C:\apache-ant-1.10.12\bin>ant clean
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

clean:
   [delete] Deleting directory C:\apache-ant-1.10.12\bin\build

BUILD SUCCESSFUL
Total time: 0 seconds
```

Pembahasan :

ant clean berguna mengembalikan project tersebut ke strukur file semula

2) Perintah **ant compile**

```
C:\apache-ant-1.10.12\bin>ant compile
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

compile:
    [mkdir] Created dir: C:\apache-ant-1.10.12\bin\build\classes
    [javac] C:\apache-ant-1.10.12\bin\build.xml:13: warning: 'includeantruntime' w
as not set, defaulting to build.sysclasspath=last; set to false for repeatable bui
lds
    [javac] Compiling 1 source file to C:\apache-ant-1.10.12\bin\build\classes

BUILD SUCCESSFUL
Total time: 2 seconds
```

Pembahasan :

perintah "compile" digunakan untuk mengompile seluruh source code

3) Perintah **ant jar**

```
C:\apache-ant-1.10.12\bin>ant jar
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

jar:
    [mkdir] Created dir: C:\apache-ant-1.10.12\bin\build\jar
      [jar] Building jar: C:\apache-ant-1.10.12\bin\build\jar\Perkalian.jar

BUILD SUCCESSFUL
Total time: 1 second
```

Pembasan :

ant jar di gunakan untuk menyatukan file class ke dalam file jar

4) Perintah **ant run (hasil program)**

```
C:\apache-ant-1.10.12\bin>ant run
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

run:
     [java] Hasil Perkalian dari 20 dan 150 adalah 3000

BUILD SUCCESSFUL
Total time: 1 second
```

Pembahasan :

perintah "run" digunakan untuk menjalankan file .jar yang telah dibuat sebelumnya.

Soal Tugas :

Buatlah proyek baru yang terdiri atas 2 file source code Java. File pertama adalah definisi class, file kedua adalah file utama yang di dalamnya akan membuat instance dari class yang telah didefinisikan pada file pertama. Buatlah file build.xml untuk keperluan seperti pada praktik di atas dan tunjukkan bahwa program yang anda buat tersebut berhasil dikompilasi dan dijalankan.

1. Membuat kelas induk BangunDatar.java, untuk menghitung luas bangun datar

bin > src > simpleant > BangunDatar.java

```java
package simpleant;


public class BangunDatar{
    float luas(){
        System.out.println("Menghitung
        luas bangun datar");
        return 0;
    }
}
```

2. Membuat program turunan dari BangunDatar.java, yaitu program menghitung luas segitiga

bin > src > simpleant > Segitiga.java

```java
package simpleant;


public class Segitiga extends BangunDatar{
    float alas, tinggi;
    float luas(){
        float luas = (alas * tinggi)/2;
        System.out.println("Luas Segitiga : " + luas);
        return luas;
    }
    public static void main(String[] args){
        //membuat objek bangun datar
        BangunDatar bangundatar = new BangunDatar();

        //membuat objek segitiga
        Segitiga segitiga = new Segitiga();
        segitiga.alas = 4;
        segitiga.tinggi = 10;

        //memanggil method luas
        bangundatar.luas();
        segitiga.luas();
    }
}
```

3. Ubah file di build.html, menjadi segitiga.java, seperti berikut ini

bin > build.xml

```
<project>
<target name="clean">
<delete dir="build"/>
</target>
<target name="compile">
<mkdir dir="build/classes"/>
<javac srcdir="src" destdir="build/classes"/>
</target>
<target name="jar" >
<mkdir dir="build/jar"/>
<jar destfile="build/jar/Segitiga.jar" basedir="build/classes">
<manifest>
<attribute name="Main-Class" value="simpleant.Segitiga"/>
</manifest>
</jar>
</target>
<target name="run">
<java jar="build/jar/Segitiga.jar" fork="true"/>
</target>
</project>

```

4. Melakukan compile program build, jar dan run
   1) Perintah **ant clean**

```
C:\apache-ant-1.10.12\bin>ant clean
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

clean:
   [delete] Deleting directory C:\apache-ant-1.10.12\bin\build

BUILD SUCCESSFUL
Total time: 0 seconds
```

Pembahasan :

ant clean berguna mengembalikan project tersebut ke strukur file semula

   2) Perintah **ant compile**

```
C:\apache-ant-1.10.12\bin>ant compile
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

compile:
    [mkdir] Created dir: C:\apache-ant-1.10.12\bin\build\classes
    [javac] C:\apache-ant-1.10.12\bin\build.xml:7: warning: 'includeantruntime' was not set, defaulting to build.sys
classpath=last; set to false for repeatable builds
    [javac] Compiling 3 source files to C:\apache-ant-1.10.12\bin\build\classes

BUILD SUCCESSFUL
Total time: 2 seconds
```

Pembahasan :

perintah "compile" digunakan untuk mengompile seluruh source code

   3) Perintah **ant jar**

```
C:\apache-ant-1.10.12\bin>ant jar
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

jar:
    [mkdir] Created dir: C:\apache-ant-1.10.12\bin\build\jar
      [jar] Building jar: C:\apache-ant-1.10.12\bin\build\jar\Segitiga.jar

BUILD SUCCESSFUL
Total time: 1 second
```

Pembasan :

ant jar di gunakan untuk menyatukan file class ke dalam file jar

4) Perintah **ant run (hasil program)**

```
C:\apache-ant-1.10.12\bin>ant run
Buildfile: C:\apache-ant-1.10.12\bin\build.xml

run:
     [java] Menghitung luas bangun datar
     [java] Luas Segitiga : 20.0

BUILD SUCCESSFUL
Total time: 1 second
```

Pembahasan :

perintah "run" digunakan untuk menjalankan file .jar yang telah dibuat sebelumnya.

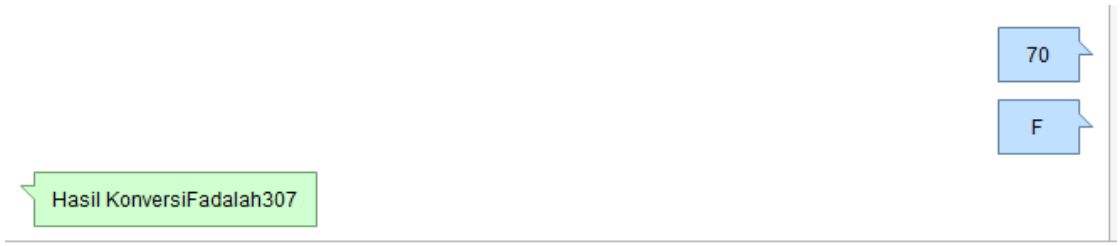
Path

1. 1-2-3-4-5-6(T)-7-16
2. 1-2-3-4-5-8(F)-9(T)-10-16
3. 1-2-3-4-5-8(F)-11(F)-12(T)-13-16
4. 1-2-3-4-5-8(F)-11(F)-14(F)-15-16

Hasil uji coba

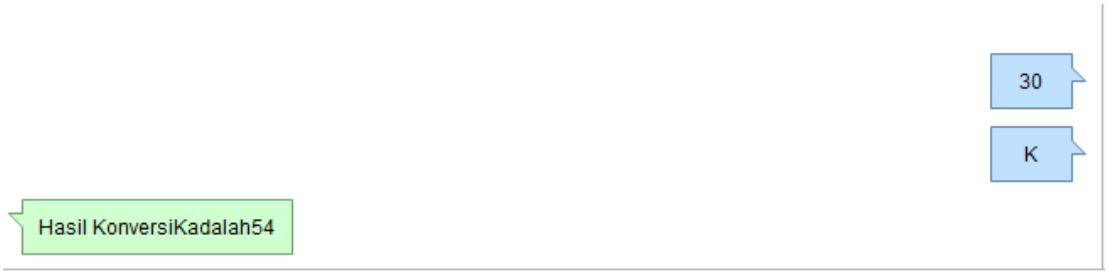
1. Suhu celsius ke Reamur (R)

2. Suhu celsius ke Fahrenhit (F)

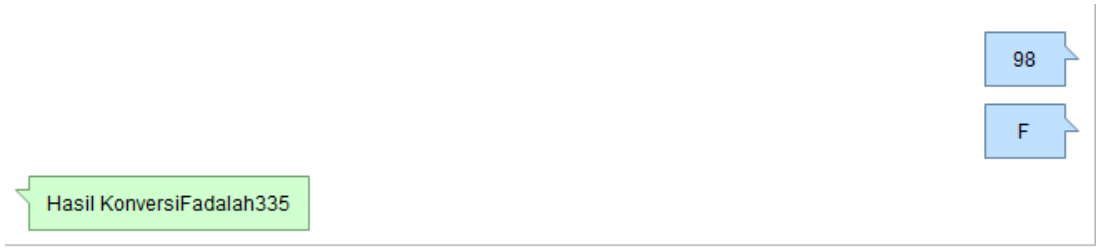


3. Suhu Celsius ke Kelvin (K)

4. Suhu Celsius ke Kelvin (K)



5. Suhu celsius ke Fahrenhit (F)



III. KESIMPULAN

Otomatisasi proses build dengan Apache Ant merupakan langkah penting dalam pengembangan perangkat lunak modern. Dengan memanfaatkan Ant, pengembang dapat mendefinisikan dan mengelola proses build secara efisien melalui file XML, yang mencakup kompilasi, pengujian, dan pengemasan aplikasi. Keuntungan utama dari otomatisasi ini meliputi:

1. Efisiensi dan Konsistensi : Proses build yang otomatis mengurangi waktu yang dibutuhkan dan meningkatkan konsistensi, mengurangi kemungkinan kesalahan manusia.
2. Integrasi Pengujian : Ant memungkinkan integrasi langsung alat pengujian seperti JUnit, sehingga pengujian dapat dilakukan secara otomatis selama fase build. Ini membantu dalam deteksi dini masalah dan menjaga kualitas kode.
3. Portabilitas dan Fleksibilitas : Ant dapat dijalankan di berbagai platform dan dapat disesuaikan untuk memenuhi kebutuhan spesifik proyek, menjadikannya alat yang fleksibel bagi tim pengembangan.
4. Peningkatan Kolaborasi : Dengan otomatisasi, seluruh tim dapat berkolaborasi lebih baik, karena proses build yang terstandarisasi meminimalkan kebingungan dan kesalahan.

Secara keseluruhan, penggunaan Apache Ant dalam otomatisasi proses build tidak hanya meningkatkan efisiensi pengembangan tetapi juga mendukung praktik pengujian yang lebih baik, sehingga menghasilkan perangkat lunak yang lebih berkualitas dan dapat diandalkan.