**Laporan Software Quality Assurance**

**Pertemuan 9 dan 10**

****

**Oleh :**

**Nama : Novita Ranti Muntiari**

**Nim : 145610135**

**Jurusan : Sistem Informasi**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**AKAKOM**

**YOGYAKARTA**

**2014**

**Code Converage Menggunakan Cobertura**

**Pertemuan 9**

1. **Latar Belakang**

Code coverage menunjukkan seberapa lengkap suatu source code telah dicakup dengan test. Code coverage merupakan bagian dari White Box testing. White Box testing merupakan testing yang dilakukan terhadap kondisi internal dari software dan berkaitan dengan cara kerja atau source code dari software tersebut secara langsung. Cobertura adalah software tool yang digunakan untuk keperluan mengukur code coverage untuk Java. Cobertura untuk melakukan proses code coverage.

1. **Pembahasan Laporan**

***Diperlukan cobertura-2.0.3***

Di Ektrak di localdisk C dan di masukkan di Folder Utama modul9

***Struktur Direktori dan Berbagai File***

Nama Folder Utama *:* ***modul9***

|-- cobertura-2.0.3

|-- build.properties

|-- build.xml

|-- classes

|-- lib

|   |-- junit-4.10.jar

|   `-- junit-dep-4.10.jar

`-- src

 |-- tests

|   `-- BicycleTest.java

 `-- vehicles

|-- Bicycle.java

`-- Car.java

***Penjelasan Dari Struktur direktori di atas***

Eksekusi Cobertura memerlukan berbagai file .jar , .java , .properties dan xml. Dari folder utama adalah modul9, dan di dalam modul9 ada 3 folder yaitu classes, lib dan src dan 1 file build.xml dan 1 file build.properties dan ekstrak file cobertura-2.0.3 , di dalam folder src ada 2 folder yaitu tests dan vehicles dan di dalam folder tests ada 1 file java yaitu BicycleTest.java dan di dalam folder vehicles ada 2 file java yaitu Bicycle.java dan Car.java dan di dalam folder lib ada 2 file yaitu , junit-4.10.jar dan junit-dep-4.10.jar dan folder classes tidak ada isinya.

***Membuat Folder di Direktori di Local di C dengan nama modul9 dengan mengikuti perintah Struktur Direktori di atas Dan Membagi file-file yang ada ke masing-masing direktori atau folder.***

***Isi dari file build.properties***

src.dir=src

cobertura.dir=cobertura-2.0.3***//lokasi tempat ekstrak cobertura ya***

lib.dir=lib

classes.dir=classes

instrumented.dir=instrumented

reports.dir=reports

reports.xml.dir=${reports.dir}/junit-xml

reports.html.dir=${reports.dir}/junit-html

coverage.xml.dir=${reports.dir}/cobertura-xml***//file disimpan di direktori reports***

coverage.summaryxml.dir=${reports.dir}/cobertura-summary-xml

coverage.html.dir=${reports.dir}/cobertura-html ***//lokasi tempat file-file***

***Isi dari file build.xml***

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

<project name="code.coverage.with.cobertura" default="coverage" basedir="."> ***//Nama project***

<description>Project's code coverage project with Cobertura</description> ***//deskrpsi***

<property file="build.properties" />***//tambahan file build.properties***

<path id="cobertura.classpath">***//pembuka id cobertura.classpath***

<fileset dir="${cobertura.dir}">***//direktori cobertura dan pembuka fileset***

<include name="cobertura.jar" />***//nama file jar***

<include name="lib/\*\*/\*.jar" />***//lokasi file jar***

</fileset>***//penutup fileset***

</path>***//penutup path id cobertura***

<path id="junit.classpath">***//pembuka id junit.classpath***

<fileset dir="${lib.dir}">***//direktori lib dan pembuka fileset***

<include name="junit-4.10.jar" />***//nama file jar***

<include name="junit-dep-4.10.jar" />***//nama file jar***

</fileset>***//penutup fileset***

</path>***//penutup path id junit.classpath***

<taskdef classpathref="cobertura.classpath" resource="tasks.properties"/>

<target name="init">***//target nama init***

<mkdir dir="${classes.dir}" />***//membuat direktori classes***

<mkdir dir="${instrumented.dir}" />***//membuat direktori instrumented***

<mkdir dir="${reports.xml.dir}" />***//membuat direktori reports.xml***

<mkdir dir="${reports.html.dir}" />***//membuat directori reports.html***

<mkdir dir="${coverage.xml.dir}" />***//membuat direktori converage.xml***

<mkdir dir="${coverage.summaryxml.dir}" />***//membuat direktori coverage.summaryxml***

<mkdir dir="${coverage.html.dir}" />***//membuat direktori coverage.html***

</target>***//penutup target***

<target name="compile" depends="init">***//nama target adalah compile***

<javac srcdir="${src.dir}" destdir="${classes.dir}" debug="yes">***//lokasi direktori***

<classpath refid="cobertura.classpath" />***//file***

<classpath refid="junit.classpath" />***//file .junit***

</javac>***//penutup javac***

</target>***//penutup target compile***

<target name="instrument" depends="init,compile">***//nama target instrument***

<delete file="cobertura.ser"/>***//menghapus file cobertura***

<delete dir="${instrumented.dir}" />***//menghapus direktori instrumented***

<cobertura-instrument todir="${instrumented.dir}">***//direktori instrumented***

<ignore regex="org.apache.log4j.\*" />

<fileset dir="${classes.dir}">***//nama direktori classes***

<include name="\*\*/\*.class" />***//lokasi***

<exclude name="\*\*/\*Test.class" />

</fileset>***//penutup fileset***

</cobertura-instrument>***//penutup cobertura instrument***

</target>***//penutup target***

<target name="test" depends="init,compile">***//nama target test***

<junit fork="yes" dir="${basedir}" failureProperty="test.failed">***//direktori***

<classpath location="${instrumented.dir}" />***//lokasi***

<classpath location="${classes.dir}" />***//lokasi***

<classpath refid="cobertura.classpath" />***//file***

<classpath refid="junit.classpath" />***//file***

<formatter type="xml" />***//format xml***

<test name="${testcase}" todir="${reports.xml.dir}" if="testcase" />***//nama test***

<batchtest todir="${reports.xml.dir}" unless="testcase">***//direktori reports***

<fileset dir="${src.dir}">***//direktori src***

<include name="\*\*/\*Test.java" />***//nama test***

</fileset>***//penutup fileset***

</batchtest>***//penutup batchtest***

</junit>***//penutup junit***

<junitreport todir="${reports.xml.dir}">***//direktori report.xml***

<fileset dir="${reports.xml.dir}">***//direktori***

<include name="TEST-\*.xml" />***//name***

</fileset>***//penutup fileset***

<report format="frames" todir="${reports.html.dir}" />***//format report***

</junitreport>***//penutup junitreport***

</target>***//penutup target***

<target name="coverage-check">***//nama target***

<cobertura-check branchrate="34" totallinerate="100" />***//check***

</target>***//penutup target***

<target name="coverage-report">***//nama target***

<cobertura-report srcdir="${src.dir}" destdir="${coverage.xml.dir}" format="xml" />

</target>

<target name="summary-coverage-report">***//nama target***

<cobertura-report srcdir="${src.dir}" destdir="${coverage.summaryxml.dir}" format="summaryXml" />

</target>

<target name="alternate-coverage-report">***//nama target***

<cobertura-report destdir="${coverage.html.dir}">

<fileset dir="${src.dir}">***//directory src***

<include name="\*\*/\*.java"/>***//nama***

</fileset>***//penutup fileset***

</cobertura-report>***//penutup cobertura report***

</target>***//penutup target***

<target name="clean" description="Remove all files created by the build/test process."> ***//nama target***

<delete dir="${classes.dir}" />***//hapus direktori***

<delete dir="${instrumented.dir}" />***//hapus direktori***

<delete dir="${reports.dir}" />***//hapus direktori***

<delete file="cobertura.log" />***//hapus file***

<delete file="cobertura.ser" />***//hapus file***

<delete file="build.xml~" />***//hapus file***

<delete file="build.properties~" />***//hapus file***

</target>***//penutup target***

<target name="coverage" depends="compile,instrument,test,coverage-report,summary-coverage-report,alternate-coverage-report" description="Compile, instrument ourself, run the tests and generate JUnit and coverage reports."/> ***//mengcompile dan report***

</project>***//penutup project***

***Isi dari file Bicycle.java***

package vehicles; ***//nama package “vehicles”***

public class Bicycle { ***//nama class “Bicycle”***

public int cadence; ***//variabel cadence menggunakan type data integer***

public int gear; ***//variabel gear menggunakan type data integer***

public int speed; ***//variabel speed menggunakan type data integer***

public Bicycle(int startCadence, int startSpeed, int startGear) { ***//variabel menggunakan type data integer***

gear = startGear; ***//variabel gear sama dengan variabel startGear***

cadence = startCadence; ***//variabel cadence sama dengan variabel startCadence***

speed = startSpeed; ***//variabel speed sama dengan variabel startSpeed***

}

public void setCadence(int newValue) { { ***//class setGadence variabel newValue menggunakan type data integer***

cadence = newValue; ***//class cadence variabel sama dengan newValue***

}

public void setGear(int newValue) { ***//class setGear variabel newValue menggunakan type data integer***

gear = newValue; ***class gear variabel sama dengan newValue***

}

public void setSpeed(int newValue) { ***//class setSpeed variabel newValue menggunakan type data integer***

speed = newValue; ***class speed variabel sama dengan newValue***

}

public int getGear() {

return gear; ***//cetak hasil gear***

}

public int getCadence() {

return cadence; ***//cetak hasil cadence***

}

public int getSpeed() {

return speed; ***//cetak hasil speed***

}

public void applyBrake(int decrement) {

speed -= decrement; ***//speed berkurang***

}

public void speedUp(int increment) {

speed += increment; ***//speed bertambah***

}

}

***Isi dari file Car.java***

package vehicles; ***//nama package “vehicles”***

public class Car{ ***//nama class “Car”***

public int cadence; ***//variabel cadence menggunakan type data integer***

public int gear; ***//variabel gear menggunakan type data integer***

public int speed; ***//variabel speed menggunakan type data integer***

public Car(int startCadence, int startSpeed, int startGear) { ***//variabel menggunakan type data integer***

gear = startGear; ***//variabel gear sama dengan variabel startGear***

cadence = startCadence; ***//variabel cadence sama dengan variabel startCadence***

speed = startSpeed; ***//variabel speed sama dengan variabel startSpeed***

}

public void setCadence(int newValue) { { ***//class setGadence variabel newValue menggunakan type data integer***

cadence = newValue; ***//class cadence variabel sama dengan newValue***

}

public void setGear(int newValue) { ***//class setGear variabel newValue menggunakan type data integer***

gear = newValue; ***class gear variabel sama dengan newValue***

}

public void setSpeed(int newValue) { ***//class setSpeed variabel newValue menggunakan type data integer***

speed = newValue; ***class speed variabel sama dengan newValue***

}

public int getGear() {

return gear; ***//cetak hasil gear***

}

public int getCadence() {

return cadence; ***//cetak hasil cadence***

}

public int getSpeed() {

return speed; ***//cetak hasil speed***

}

public void applyBrake(int decrement) {

speed -= decrement; ***//speed berkurang***

}

public void speedUp(int increment) {

speed += increment; ***//speed bertambah***

}

public void park() {

speed = 0;

cadence = 0;

gear = 0;

***//dimulai dari nol***

}

}

***Isi dari file BicycleMain.java***

package tests; ***//nama package “tests”***

import vehicles.\*; ***//masukkan “vehicles”***

import org.junit.\*; ***//masukkan “org.junit”***

import static org.junit.Assert.\*; ***//masukkan “Assert”***

public class BicycleTest {***//nama class “BicycleTest”***

 @Test

 public void test\_applyBrake() {***//Metode***

 System.out.println("Test applyBrake method ...");***//Massage***

Bicycle Bike = new Bicycle(10, 20, 1); ***//Membuat Variabel Bike = Bicycle***

 assertEquals(Bike.getGear(), 1); ***//assertEquals getGear dengan 1***

 assertEquals(Bike.getSpeed(), 20); ***// assertEquals getSpeed dengan 20***

 assertEquals(Bike.getCadence(), 10); ***// assertEquals getCadence dengan 10***

 // testing setter

 Bike.setGear(2); ***//Bike setGear ya 2***

 Bike.setSpeed(30); ***//Bike setSpeed ya 30***

 Bike.setCadence(5); ***//Bike setCadence ya 5***

 assertEquals(Bike.getGear(), 2); ***//assertEquals di Bike getGear dengan 2***

 assertEquals(Bike.getSpeed(), 30);***//assertEquals di Bike getSpeed dengan 30***

 assertEquals(Bike.getCadence(), 5); );***//assertEquals di Bike getCadence dengan 5***

 Bike.applyBrake(10); );***// Bike applyBrake dengan 10***

 assertEquals(Bike.getSpeed(), 20); ***//assertEquals di bike dengan 20***

}

 @Test

public void test\_speedUp() {***//Metode***

 System.out.println("Test speedUp method ...") ; ***//Massage***

 Bicycle Bike = new Bicycle(10, 20, 1); ***//Membuat Variabel Bike = Bicycle***

 assertEquals(Bike.getGear(), 1); ***//assertEquals getGear dengan 1***

 assertEquals(Bike.getSpeed(), 20); ***// assertEquals getSpeed dengan 20***

 assertEquals(Bike.getCadence(), 10); ***// assertEquals getCadence dengan 10***

 // testing setter

 Bike.setGear(2); ***//Bike setGear ya 2***

 Bike.setSpeed(30); ***//Bike setSpeed ya 30***

 Bike.setCadence(5); ***//Bike setCadence ya 5***

 assertEquals(Bike.getGear(), 2); ***//assertEquals di Bike getGear dengan 2***

 assertEquals(Bike.getSpeed(), 30); ***//assertEquals di Bike getSpeed dengan 30***

 assertEquals(Bike.getCadence(), 5); ***//assertEquals di Bike getCadence dengan 5***

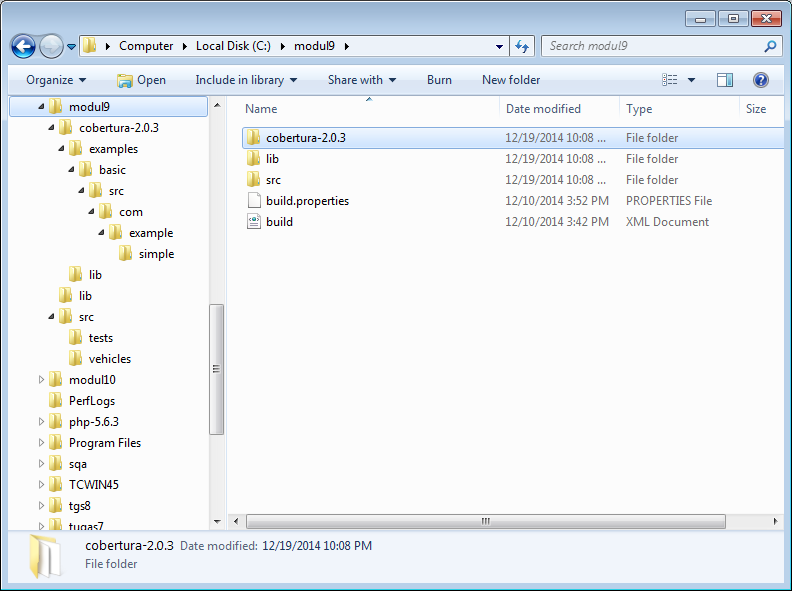
 Bike.speedUp(5);***//Speed ya 5***

 assertEquals(Bike.getSpeed(), 35); ***//assertEquals di bike dengan 35***

}

}

***Tampilan Direktori di localdisc C di dalam modul9:***

***Buka Command Prompt***

*Selanjutya Ketikan di Command Prompt :*

C:\modul9>ant compile***//mengcompile atau menjalankan program***

***Output :***

Buildfile: C:\modul9\build.xml

[taskdef] Could not load definitions from resource tasks.properties. It could

not be found.

init:

[mkdir] Created dir: C:\modul9\classes

[mkdir] Created dir: C:\modul9\instrumented

[mkdir] Created dir: C:\modul9\reports\junit-xml

[mkdir] Created dir: C:\modul9\reports\junit-html

[mkdir] Created dir: C:\modul9\reports\cobertura-xml

[mkdir] Created dir: C:\modul9\reports\cobertura-summary-xml

[mkdir] Created dir: C:\modul9\reports\cobertura-html

compile:

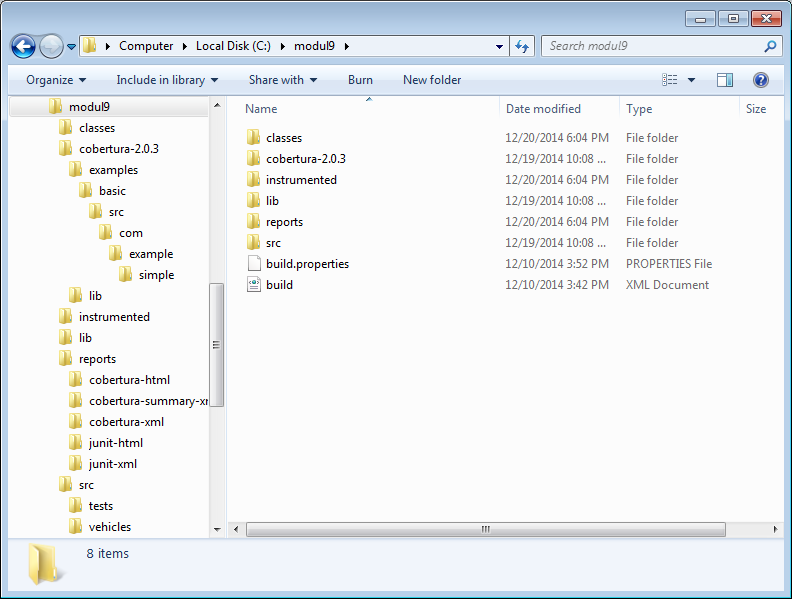
[javac] C:\modul9\build.xml:28: warning: 'includeantruntime' was not set, de

faulting to build.sysclasspath=last; set to false for repeatable builds

[javac] Compiling 3 source files to C:\modul9\classes

BUILD SUCCESSFUL

Total time: 1 second

***Tampilan Direktori setelah di compile di localdisc C di dalam modul9:***

Setelah di compile terdapat folder baru yaitu instrumented dan report yang terdapat 5 folder didalamnya.

1. **Tugas Dan Pembahasan**

Carilah contoh program dari Internet. Sebutkan URL tempat program tersebut berada. Buatlah analisis code coverage dari program tersebut dengan menggunakan Cobertura.

URL : http://sekedar-tutorial.blogspot.com/2011/05/modul-1-praktikum-pbo-class-dan-object.html

***Caranya sama dengan mengerjakan latihan di atas :***

Nama Folder Utama *:* ***tugas9***

***Struktur Direktori :***

|-- cobertura-2.0.3

|-- build.properties

|-- build.xml

|-- classes

|-- lib

|   |-- junit-4.10.jar

|   `-- junit-dep-4.10.jar

`-- src

 |-- tests

|   `-- televOOP.java

 `-- vehicles

`-- tv.java

***Penjelasan Dari Struktur direktori di atas***

Eksekusi Cobertura memerlukan berbagai file .jar , .java , .properties dan xml. Dari folder utama adalah tugas9, dan di dalam tugas9 ada 3 folder yaitu classes, lib dan src dan 1 file build.xml dan 1 file build.properties dan ekstrak file cobertura-2.0.3 , di dalam folder src ada 2 folder yaitu tests dan vehicles dan di dalam folder tests ada 1 file java yaitu televOOP.java dan di dalam folder vehicles ada 1 file java yaitu tv.java dan di dalam folder lib ada 2 file yaitu , junit-4.10.jar dan junit-dep-4.10.jar dan folder classes tidak ada isinya.

***Membuat Folder di Direktori di Local di C dengan nama tugas9 dengan mengikuti perintah Struktur Direktori di atas Dan Membagi file-file yang ada ke masing-masing direktori atau folder.***

***Isi dari file build.properties***

src.dir=src

cobertura.dir=cobertura-2.0.3***//lokasi tempat ekstrak cobertura ya***

lib.dir=lib

classes.dir=classes

instrumented.dir=instrumented

reports.dir=reports

reports.xml.dir=${reports.dir}/junit-xml

reports.html.dir=${reports.dir}/junit-html

coverage.xml.dir=${reports.dir}/cobertura-xml***//file disimpan di direktori reports***

coverage.summaryxml.dir=${reports.dir}/cobertura-summary-xml

coverage.html.dir=${reports.dir}/cobertura-html ***//lokasi tempat file-file***

***Isi dari file build.xml***

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

<project name="code.coverage.with.cobertura" default="coverage" basedir="."> ***//Nama project***

<description>Project's code coverage project with Cobertura</description> ***//deskrpsi***

<property file="build.properties" />***//tambahan file build.properties***

<path id="cobertura.classpath">***//pembuka id cobertura.classpath***

<fileset dir="${cobertura.dir}">***//direktori cobertura dan pembuka fileset***

<include name="cobertura.jar" />***//nama file jar***

<include name="lib/\*\*/\*.jar" />***//lokasi file jar***

</fileset>***//penutup fileset***

</path>***//penutup path id cobertura***

<path id="junit.classpath">***//pembuka id junit.classpath***

<fileset dir="${lib.dir}">***//direktori lib dan pembuka fileset***

<include name="junit-4.10.jar" />***//nama file jar***

<include name="junit-dep-4.10.jar" />***//nama file jar***

</fileset>***//penutup fileset***

</path>***//penutup path id junit.classpath***

<taskdef classpathref="cobertura.classpath" resource="tasks.properties"/>

<target name="init">***//target nama init***

<mkdir dir="${classes.dir}" />***//membuat direktori classes***

<mkdir dir="${instrumented.dir}" />***//membuat direktori instrumented***

<mkdir dir="${reports.xml.dir}" />***//membuat direktori reports.xml***

<mkdir dir="${reports.html.dir}" />***//membuat directori reports.html***

<mkdir dir="${coverage.xml.dir}" />***//membuat direktori converage.xml***

<mkdir dir="${coverage.summaryxml.dir}" />***//membuat direktori coverage.summaryxml***

<mkdir dir="${coverage.html.dir}" />***//membuat direktori coverage.html***

</target>***//penutup target***

<target name="compile" depends="init">***//nama target adalah compile***

<javac srcdir="${src.dir}" destdir="${classes.dir}" debug="yes">***//lokasi direktori***

<classpath refid="cobertura.classpath" />***//file***

<classpath refid="junit.classpath" />***//file .junit***

</javac>***//penutup javac***

</target>***//penutup target compile***

<target name="instrument" depends="init,compile">***//nama target instrument***

<delete file="cobertura.ser"/>***//menghapus file cobertura***

<delete dir="${instrumented.dir}" />***//menghapus direktori instrumented***

<cobertura-instrument todir="${instrumented.dir}">***//direktori instrumented***

<ignore regex="org.apache.log4j.\*" />

<fileset dir="${classes.dir}">***//nama direktori classes***

<include name="\*\*/\*.class" />***//lokasi***

<exclude name="\*\*/\*Test.class" />

</fileset>***//penutup fileset***

</cobertura-instrument>***//penutup cobertura instrument***

</target>***//penutup target***

<target name="test" depends="init,compile">***//nama target test***

<junit fork="yes" dir="${basedir}" failureProperty="test.failed">***//direktori***

<classpath location="${instrumented.dir}" />***//lokasi***

<classpath location="${classes.dir}" />***//lokasi***

<classpath refid="cobertura.classpath" />***//file***

<classpath refid="junit.classpath" />***//file***

<formatter type="xml" />***//format xml***

<test name="${testcase}" todir="${reports.xml.dir}" if="testcase" />***//nama test***

<batchtest todir="${reports.xml.dir}" unless="testcase">***//direktori reports***

<fileset dir="${src.dir}">***//direktori src***

<include name="\*\*/\*Test.java" />***//nama test***

</fileset>***//penutup fileset***

</batchtest>***//penutup batchtest***

</junit>***//penutup junit***

<junitreport todir="${reports.xml.dir}">***//direktori report.xml***

<fileset dir="${reports.xml.dir}">***//direktori***

<include name="TEST-\*.xml" />***//name***

</fileset>***//penutup fileset***

<report format="frames" todir="${reports.html.dir}" />***//format report***

</junitreport>***//penutup junitreport***

</target>***//penutup target***

<target name="coverage-check">***//nama target***

<cobertura-check branchrate="34" totallinerate="100" />***//check***

</target>***//penutup target***

<target name="coverage-report">***//nama target***

<cobertura-report srcdir="${src.dir}" destdir="${coverage.xml.dir}" format="xml" />

</target>

<target name="summary-coverage-report">***//nama target***

<cobertura-report srcdir="${src.dir}" destdir="${coverage.summaryxml.dir}" format="summaryXml" />

</target>

<target name="alternate-coverage-report">***//nama target***

<cobertura-report destdir="${coverage.html.dir}">

<fileset dir="${src.dir}">***//directory src***

<include name="\*\*/\*.java"/>***//nama***

</fileset>***//penutup fileset***

</cobertura-report>***//penutup cobertura report***

</target>***//penutup target***

<target name="clean" description="Remove all files created by the build/test process."> ***//nama target***

<delete dir="${classes.dir}" />***//hapus direktori***

<delete dir="${instrumented.dir}" />***//hapus direktori***

<delete dir="${reports.dir}" />***//hapus direktori***

<delete file="cobertura.log" />***//hapus file***

<delete file="cobertura.ser" />***//hapus file***

<delete file="build.xml~" />***//hapus file***

<delete file="build.properties~" />***//hapus file***

</target>***//penutup target***

<target name="coverage" depends="compile,instrument,test,coverage-report,summary-coverage-report,alternate-coverage-report" description="Compile, instrument ourself, run the tests and generate JUnit and coverage reports."/> ***//mengcompile dan report***

</project>***//penutup project***

***Isi dari file tv.java***

public class tv{ ***//nama class “tv”***

String merk,tipe,ukuran,tombol\_volume,tombol\_channel; ***//variabel dan type data***

public tv(String televisi){ ***//variable televisi***

System.out.println("Objek "+televisi+" dari kelas tv sudah dibuat\n");***//keterangan object***

}

public void menyalakan\_tv(String power){ ***//public menyalakan tv***

System.out.println("Televisi sudah : "+power); ***//keterangan***

}

public void ganti\_channel(){***//public ganti channel***

System.out.println("Channel Sudah diubah");***//keterangan***

}

public void membesarkan\_volume(String suara){ ***//public memperbesar volume***

System.out.println("Volume sudah di"+suara+"kan");***//keterangan***

}

}

***Isi dari file televOOP.java***

class televOOP{***//nama class “televOOP”***

public static void main(String[] args){

tv televisiku = new tv("televisiku");***//televisi***

televisiku.merk = "Samsung";***//isi merek “Samsung”***

System.out.println("Merk Televisi : "+televisiku.merk); ***//merek televisi***

televisiku.tipe = "LCD TV";***//keterangan “LCD TV”***

System.out.println("Tipe Televisi : "+televisiku.tipe); ***//type televisi***

televisiku.ukuran = "52 inchi";***//keterangan “52 inchi”***

System.out.println("Ukuran Televisi : "+televisiku.ukuran); ***//ukuran tv***

televisiku.menyalakan\_tv("ON");***//menyalakan tv***

televisiku.ganti\_channel();***//ganti channel***

televisiku.membesarkan\_volume("besar");***//setting volume(+)***

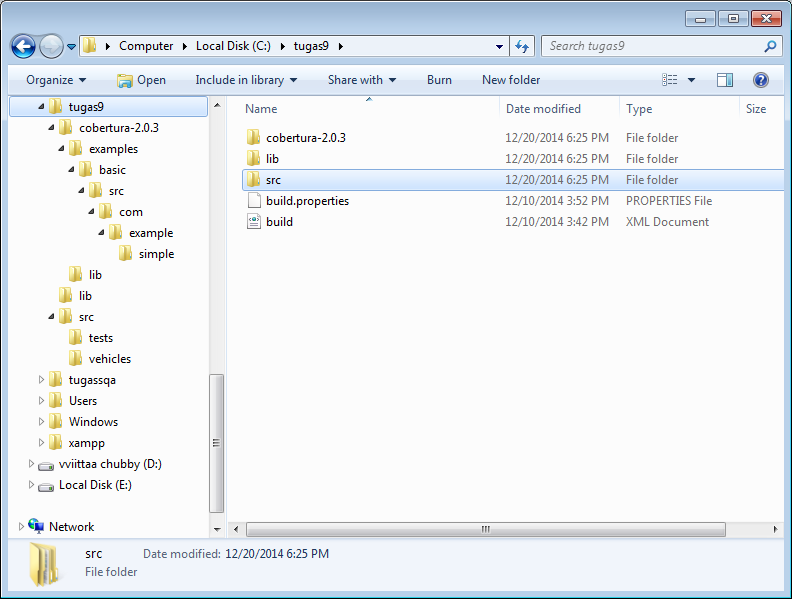
televisiku.membesarkan\_volume("kecil");***//setting volume(-)***

televisiku.menyalakan\_tv("OFF");***//televise off***

}

}

***Tampilan Direktori di localdisc C di dalam tugas9:***



***Buka Command Prompt***

*Selanjutya Ketikan di Command Prompt :*

C:\tugas9>ant compile***//mengcompile atau menjalankan program***

***Output :***

Buildfile: C:\tugas9\build.xml

[taskdef] Could not load definitions from resource tasks.properties. It could

not be found.

init:

[mkdir] Created dir: C:\tugas9\classes

[mkdir] Created dir: C:\tugas9\instrumented

[mkdir] Created dir: C:\tugas9\reports\junit-xml

[mkdir] Created dir: C:\tugas9\reports\junit-html

[mkdir] Created dir: C:\tugas9\reports\cobertura-xml

[mkdir] Created dir: C:\tugas9\reports\cobertura-summary-xml

[mkdir] Created dir: C:\tugas9\reports\cobertura-html

compile:

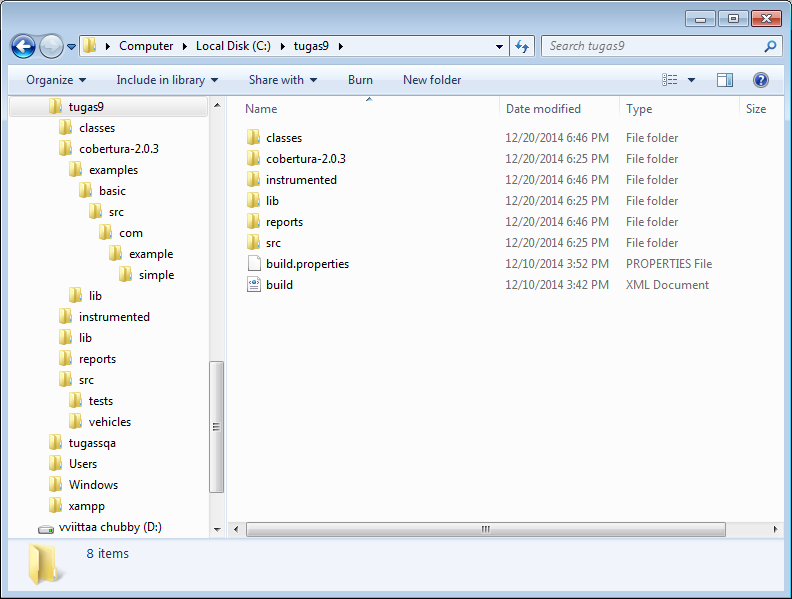
[javac] C:\tugas9\build.xml:28: warning: 'includeantruntime' was not set, de

faulting to build.sysclasspath=last; set to false for repeatable builds

[javac] Compiling 2 source files to C:\tugas9\classes

BUILD SUCCESSFUL

Total time: 1 second

***Tampilan Direktori setelah di compile di localdisc C di dalam tugas9:***

Setelah di compile terdapat folder baru yaitu instrumented dan report yang terdapat 5 folder didalamnya.

1. **Kesimpulan**

Menggunakan Cobertura menjalankan butuh file .jar , build.xml dan build.properties dan program .java yang berisi perintah dari program tersebut dan menghasilkan dan membantu untuk menjalankan program tersebut .

1. **Listing Terlampir**

**Refactoring**

**Pertemuan 10**

1. **Latar Belakang**

Refactoring adalah proses yang dilakukan oleh developer untuk memperbaiki kualitas internal source code tanpa mengubah behaviour atau fungsionalitas dari program aplikasi yang dibangun. Refactoring ini merupakan suatu bentuk dari program transformation. Yang dimana refactoring memudahkan kita dalam mengerjakan suatu itu lebih cepat dengan contohya merubah nama, menghapus data, memindahkan data ke package lain dengan move dan menduplikat file

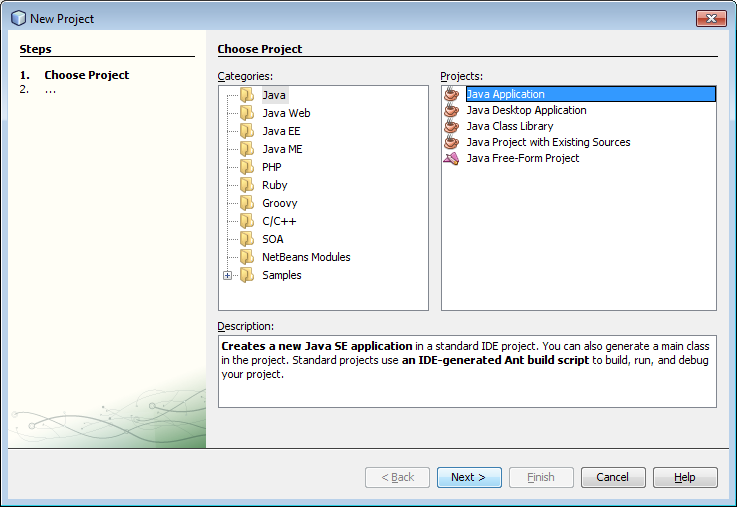
1. **Pembahasan Laporan**

***Diperlukan aplikasi yaitu NetBeans***

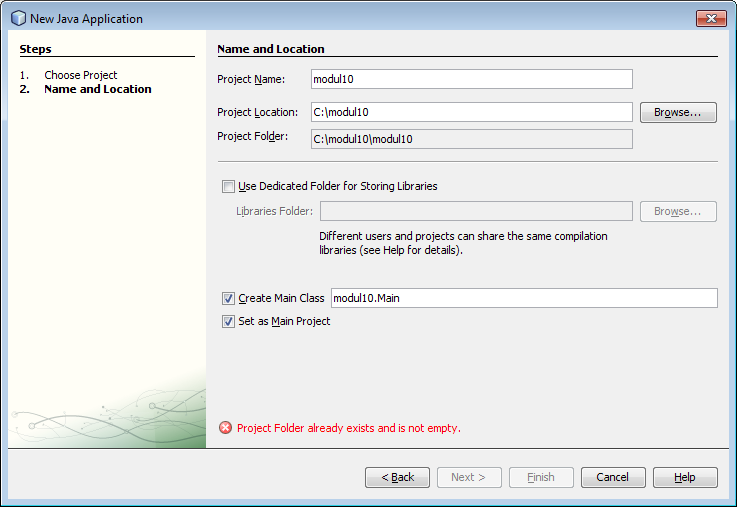
Langkah :

Membuat Project Baru , klik File lalu pilih New Project

*Akan muncul tampilan seperti di bawah ini :*



Pilih java – Java Application lalu tekan Next

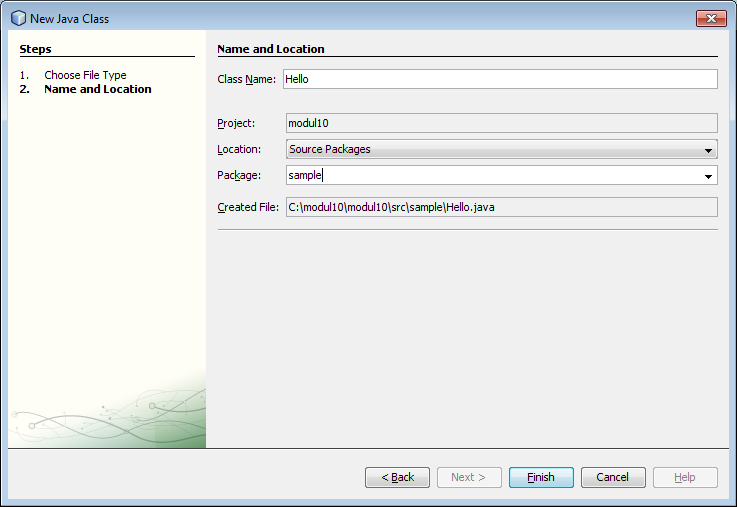


Project name adalah modul10 dan Project Location di simpan di directori C dengan nama folder modul10 lalu Finish.

***Membuat class dan Packages***

Klik kanan pada project **modul10** , pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



Class name diisikan Hello akan otomatis ke save dengan Hello.java dan packagenya diisikan sample lalu klik Finish.

Di dalam class Hello.java berisikan di bawah ini :

package sample; ***// nama package***

public class Hello { ***//nama class***

public static void main(String args[]){ ***//badan program***

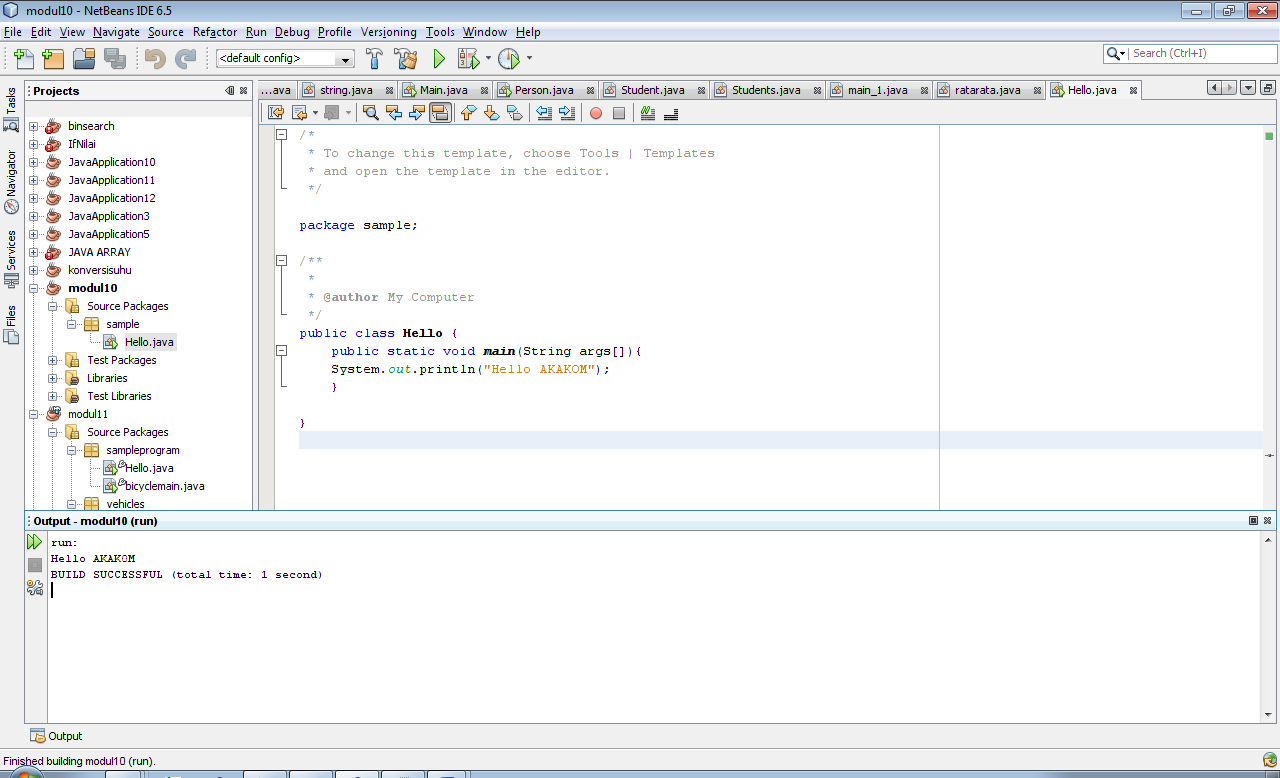
System.out.println("Hello AKAKOM"); ***//isi dari output***

}

}***//penutup program***

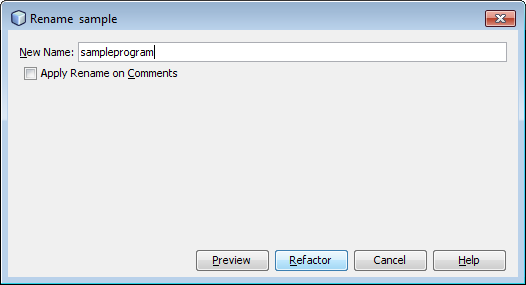
Setelah itu Run Program dengan cara klik kanan pada class Hello.java lalu pilih Run File atau Shift+F6

**Hasil Output Hello.java**



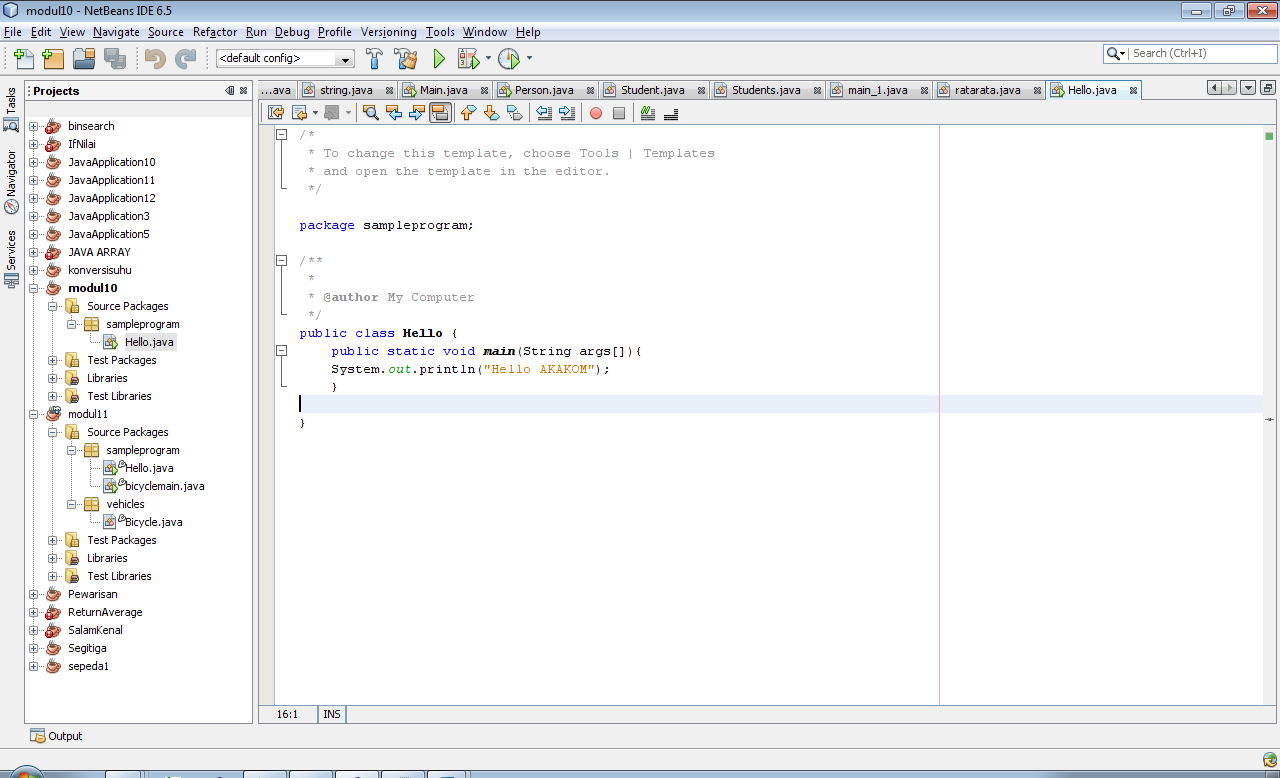
Selanjutya klik kanan pada package ***sample*** pilih Refaktor lalu pilih ***rename***

*Akan muncul tampilan seperti di bawah ini :*



Ubah New Name yang awalnya ***sample*** menjadi ***sampleprogram*** lalu klik ***Refactor***

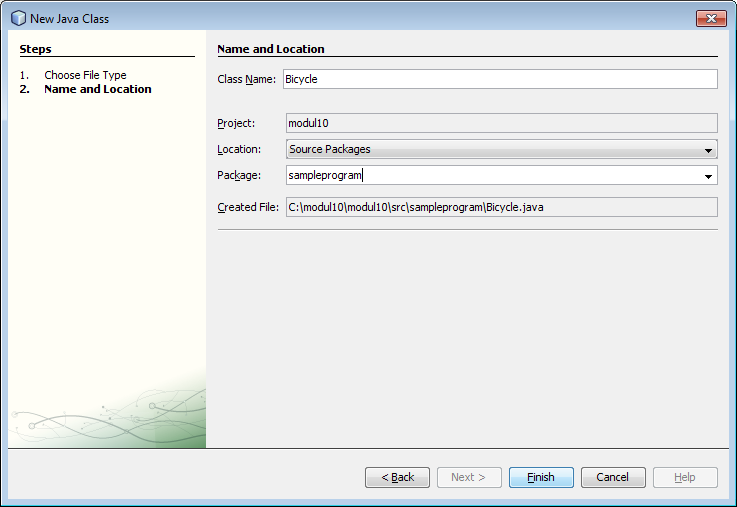
Nama Package yang awalnya sample berubah jadi sampleprogram dan secara otomatis kalau menggunakan Refactor nama package yang ada di script hello.java pasti ikut berubah.

**Tampilan program menjadi :**

Setelah selesai selanjutya kita menambahkan 2 file java dengan nama Bicycle.java dan BicycleMain.java dan 2 file java tersebut berada pada package sampleprogram.

Klik kanan pada ***package sampleprogram*** pilih ***new*** lalu pilih ***java class***

*Akan muncul tampilan seperti di bawah ini :*



Class name diisikan Bicycle akan otomatis ke save dengan Bicycle.java dan packagenya diisikan sampleprogram lalu klik Finish.

Di dalam class Bicycle.java berisikan di bawah ini :

package sampleprogram;***//nama package “sampleprogram”***

public class Bicycle { ***//nama class “Bicycle”***

public int cadence; ***//variabel cadence menggunakan type data integer***

public int gear; ***//variabel gear menggunakan type data integer***

public int speed; ***//variabel speed menggunakan type data integer***

public Bicycle(int startCadence, int startSpeed, int startGear) { ***//variabel menggunakan type data integer***

gear = startGear; ***//variabel gear sama dengan variabel startGear***

cadence = startCadence; ***//variabel cadence sama dengan variabel startCadence***

speed = startSpeed; ***//variabel speed sama dengan variabel startSpeed***

}

public void setCadence(int newValue) { { ***//class setGadence variabel newValue menggunakan type data integer***

cadence = newValue; ***//class cadence variabel sama dengan newValue***

}

public void setGear(int newValue) { ***//class setGear variabel newValue menggunakan type data integer***

gear = newValue; ***class gear variabel sama dengan newValue***

}

public void setSpeed(int newValue) { ***//class setSpeed variabel newValue menggunakan type data integer***

speed = newValue; ***class speed variabel sama dengan newValue***

}

public int getGear() {

return gear; ***//cetak hasil gear***

}

public int getCadence() {

return cadence; ***//cetak hasil cadence***

}

public int getSpeed() {

return speed; ***//cetak hasil speed***

}

public void applyBrake(int decrement) {

speed -= decrement; ***//speed berkurang***

}

public void speedUp(int increment) {

speed += increment; ***//speed bertambah***

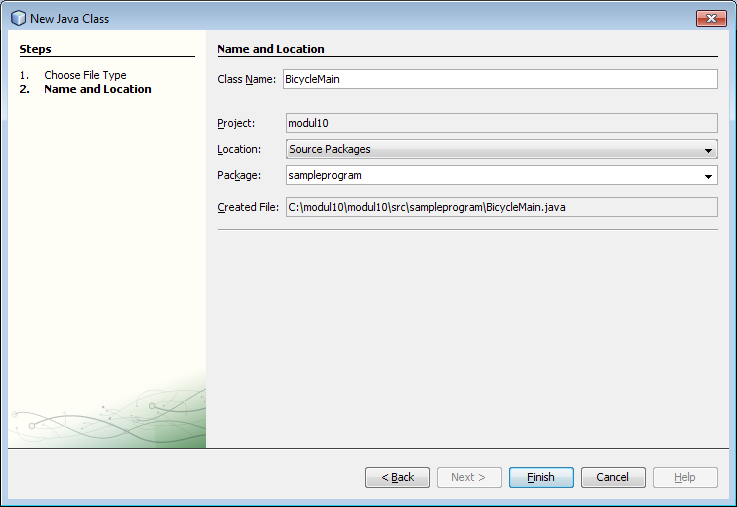
}

}

Setelah buat class Bicycle.java buat class BicycleMain.java

Klik kanan pada ***package sampleprogram*** pilih ***new*** lalu pilih ***java class***

*Akan muncul tampilan seperti di bawah ini :*



Class name diisikan BicycleMain akan otomatis ke save dengan BicycleMain.java dan packagenya diisikan sampleprogram lalu klik Finish.

Di dalam class BicycleMain.java berisikan di bawah ini :

package sampleprogram;; ***//nama package “sampleprogram”***

public class BicycleMain {***//nama class “BicycleMain”***

public static void main (String args[]) {

Bicycle Bike = new Bicycle(10, 20, 1); ***//Membuat Variabel Bike = Bicycle***

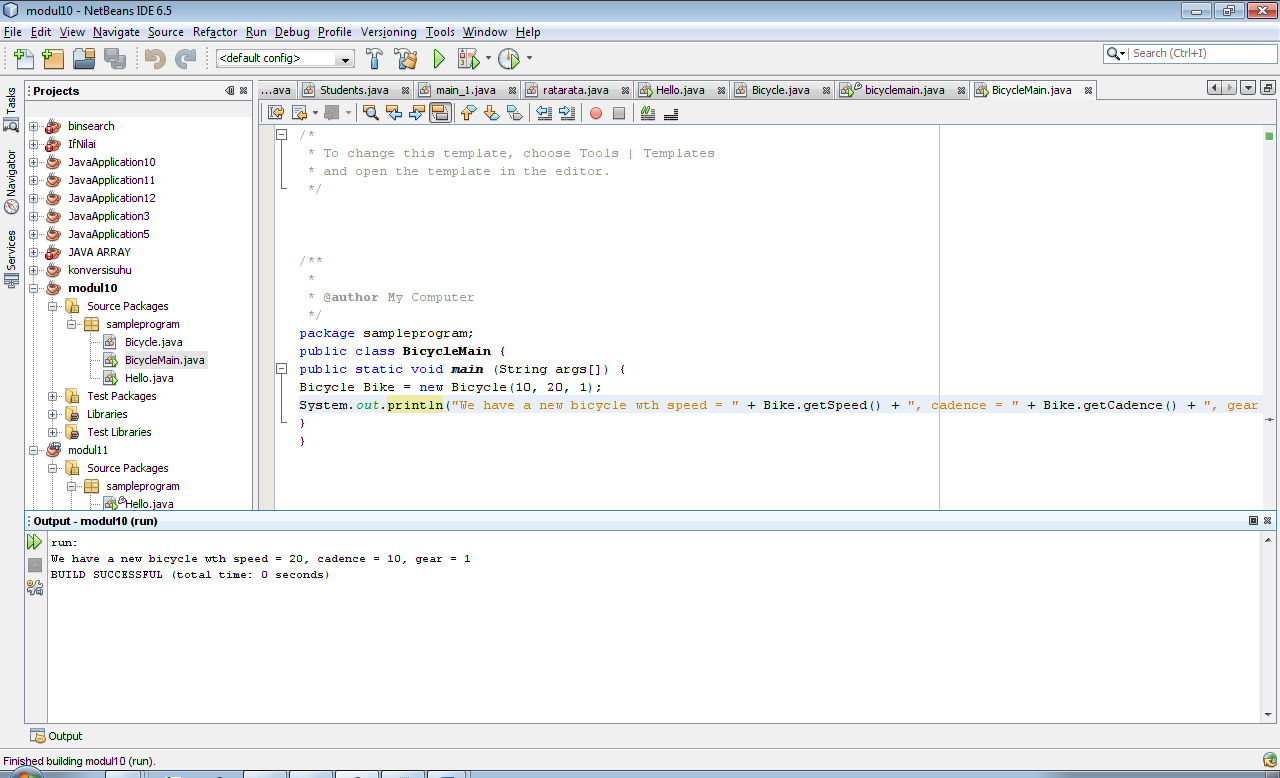
System.out.println("We have a new bicycle wth speed = " + Bike.getSpeed() + ", cadence = " + Bike.getCadence() + ", gear = " + Bike.getGear());

} ***//output speed,cadence dan gear***

}

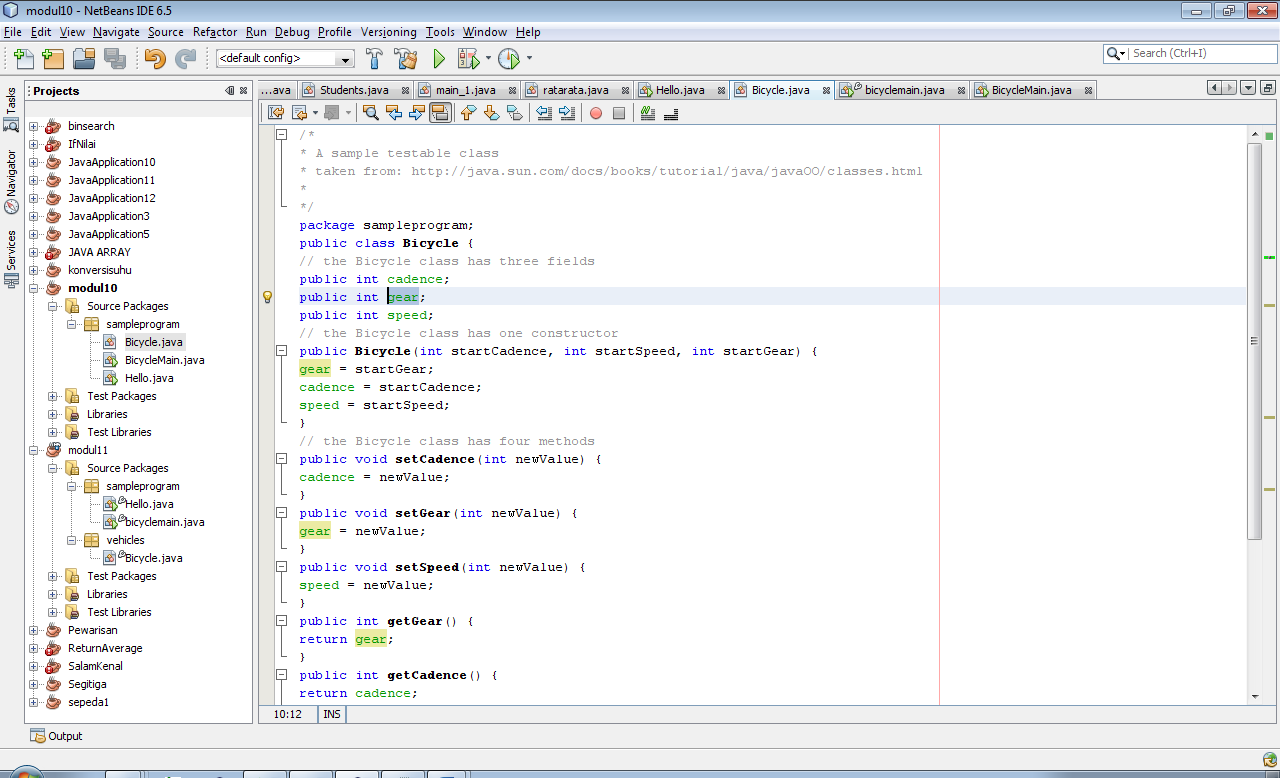
Setelah itu Run Program dengan cara klik kanan pada class BicycleMain.java lalu pilih Run File atau Shift+F6

**Hasil Output BicycleMain.java**

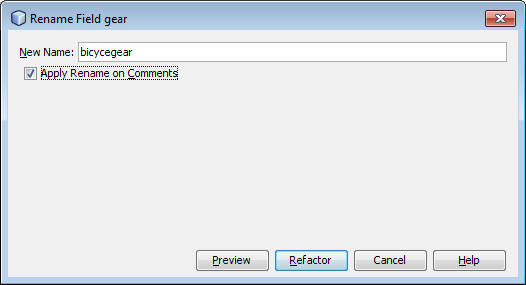


***Merefactor script java di file java Bicycle.java yaitu gear***

Klik kanan pada script java ***gear*** lalu pilih ***refaktor*** lalu pilih ***rename.*** Seperti tampilan di bawah ini :



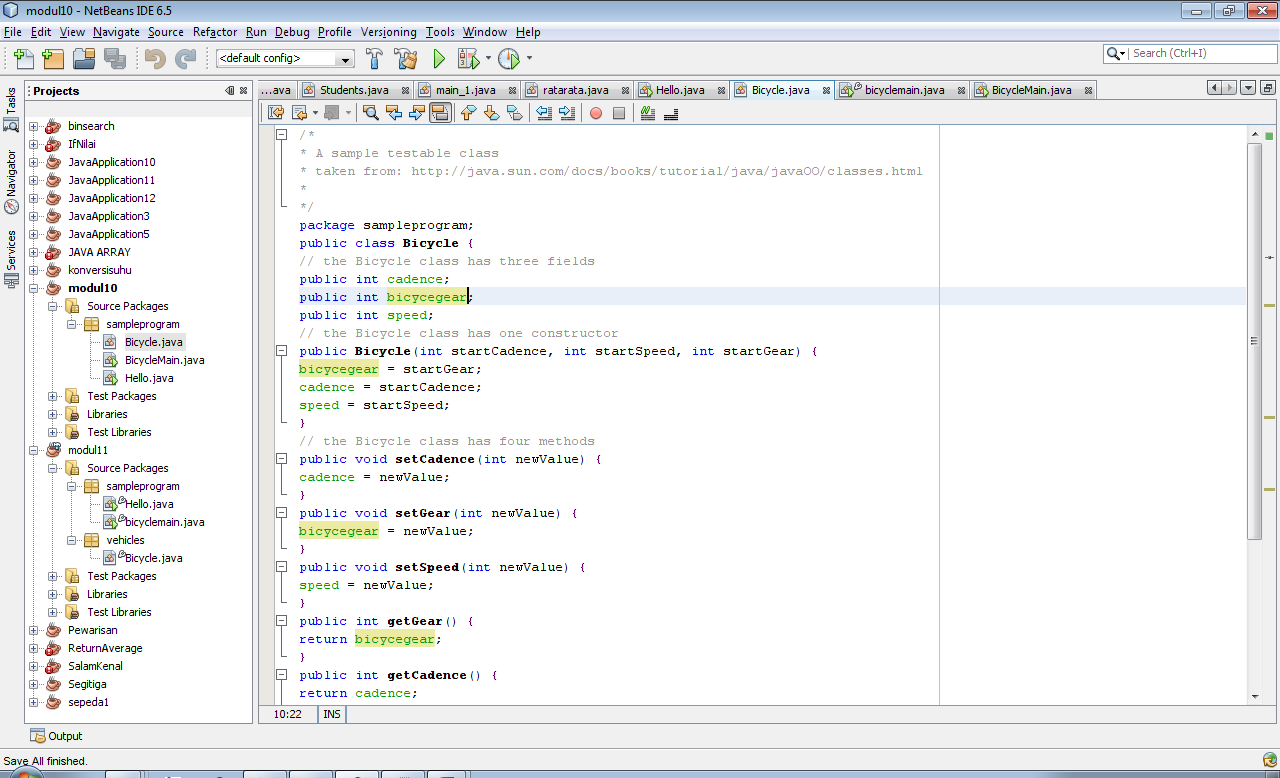
*Akan muncul tampilan seperti di bawah ini :*



New Name ya diisi ***bicyclegear*** yang awalnya hanya ***gear*** lalu centang “***Apply Rename On Comments***“dan klik ***Refactor***

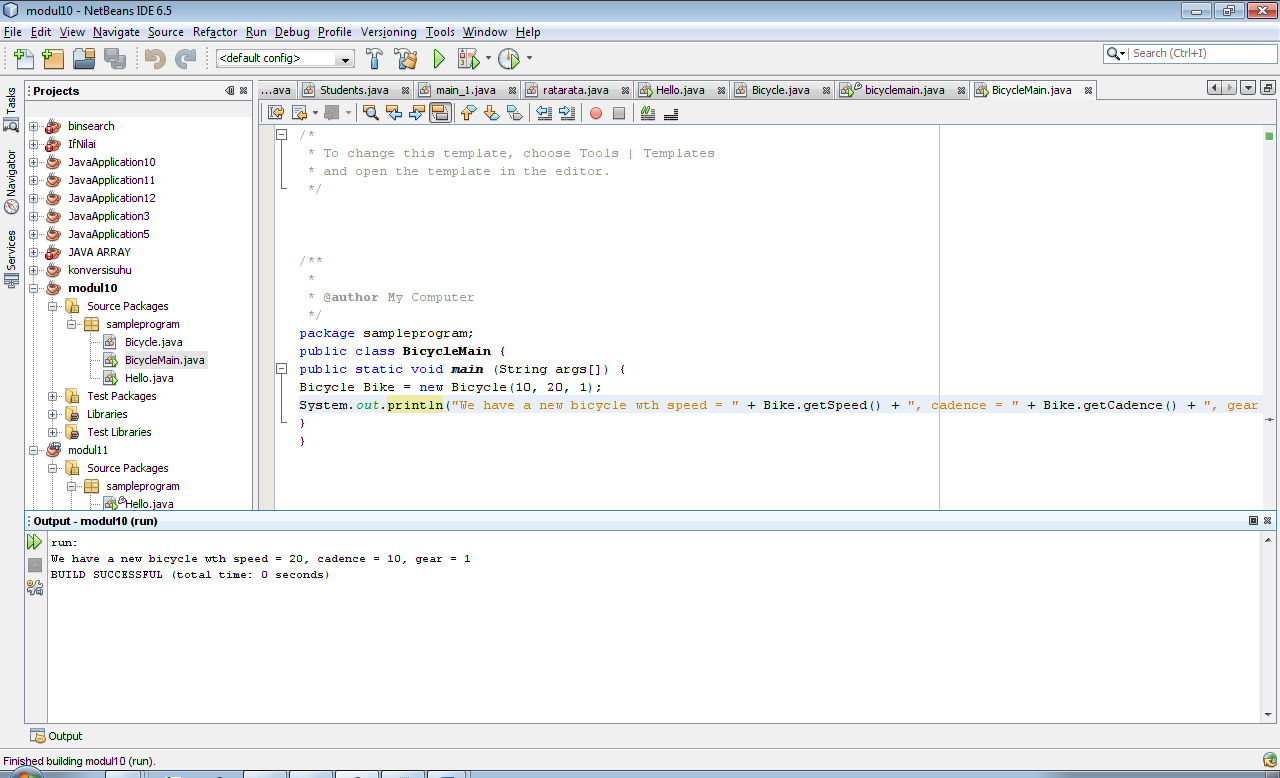
Akan terjadi perubahan pada file ***Bicycle.java*** yaitu script ***gear*** berubah semua script menjadi ***bicyclegear*** .

**Tampilanya :**



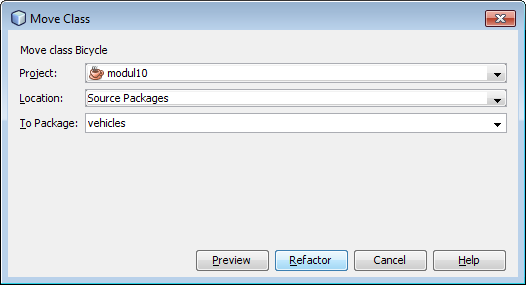
Setelah itu Run Program lagi dengan cara klik kanan pada class BicycleMain.java lalu pilih Run File atau Shift+F6

**Hasil Output BicycleMain.java**



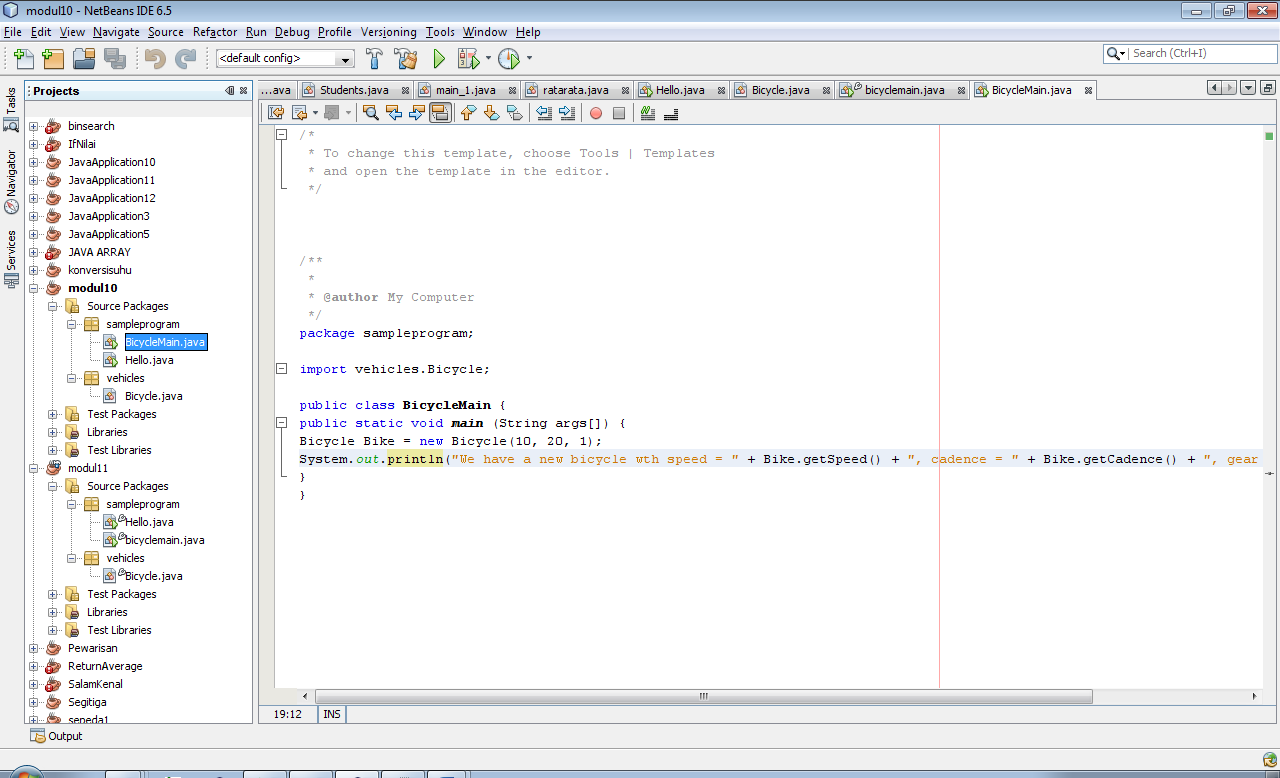
***Refaktore file java Bicycle.java pindahkan ke package vehicles***

Klik kanan pada file java ***Bicycle.java*** lalu pilih ***refactor*** lalu pilih ***move***

*Akan muncul tampilan seperti di bawah ini :*

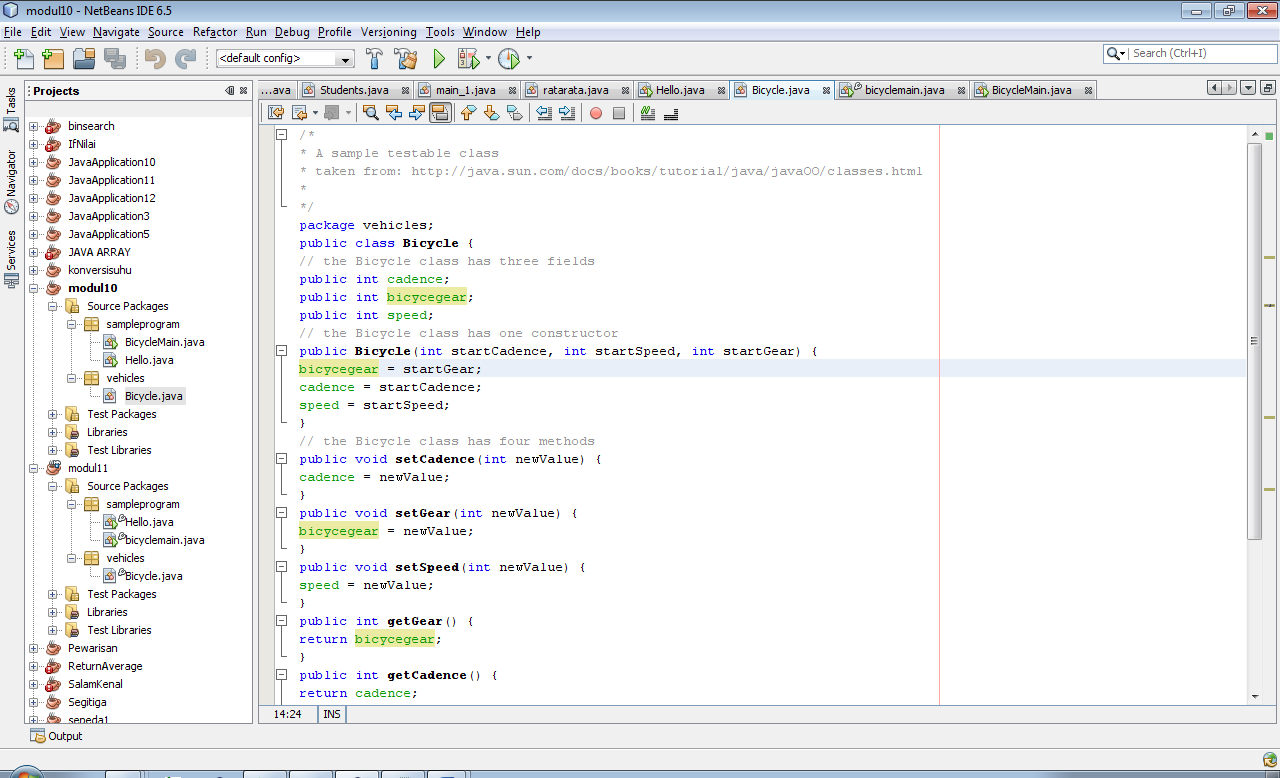
To Package yang awal ya ***sampleprogram*** di isi ***vehicles*** lalu klik ***Refaktor***

***Hasilya :***



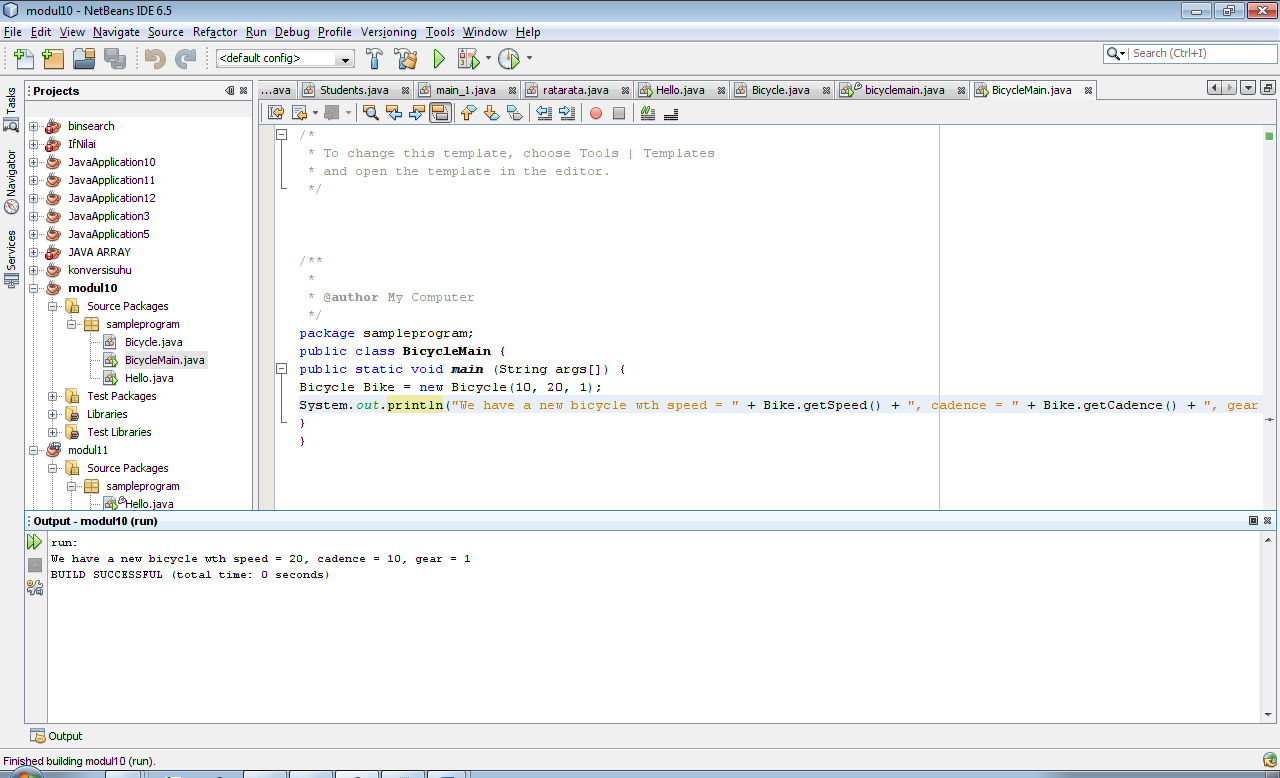
File java ***Bicycle.java*** berada di package ***vehicles***

Dan pada file ***Bicycle.java*** ada perubahan script secara otomatis setelah melakukan cara seperti di atas, yaitu nama package ya secara otomatis akan berubah , seperti di bawah ini :



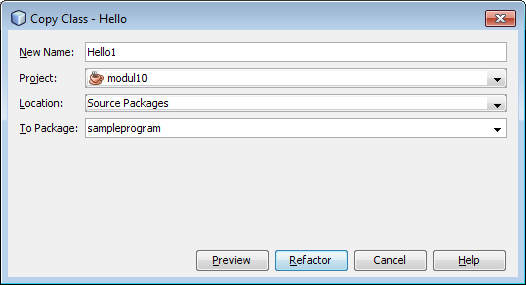
Setelah itu Run Program lagi dengan cara klik kanan pada class BicycleMain.java lalu pilih Run File atau Shift+F6

**Hasil Output BicycleMain.java**



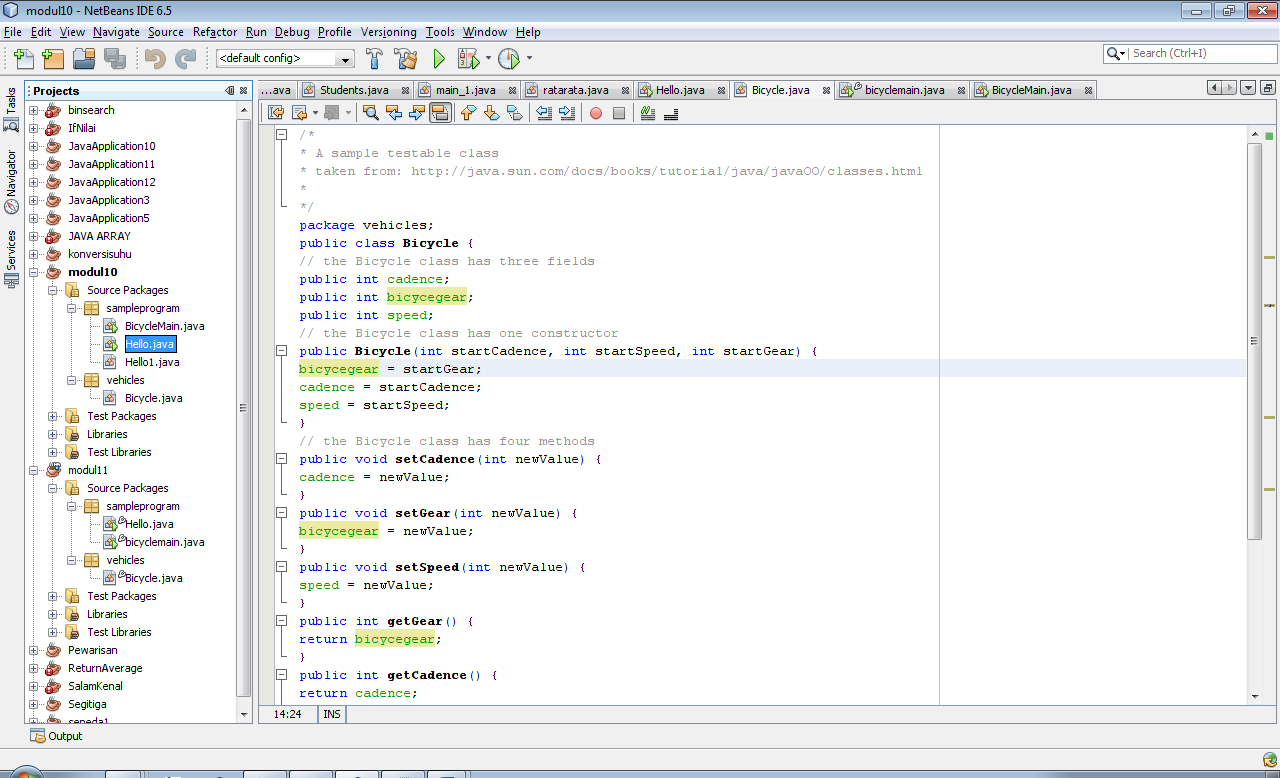
***Refactor file java hello.java menduplikat menggunakan Copy***

Klik kanan pada file java ***hello.java*** lalu pilih ***refactor*** lalu pilih ***Copy***

*Akan muncul tampilan seperti di bawah ini :*

New Name yang awal ya ***Hello*** diisi secara otomatis ***Hello1*** To Package tetap di **sampleprogram** lalu klik ***Refaktor***

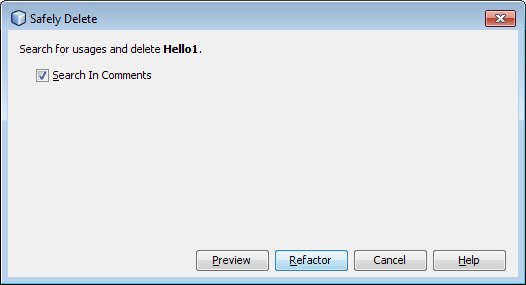
***Hasilya :***



Ada class ***Hello1.java*** di package ***sampleprogram***

***Refactor file java hello1.java menghapus menggunakan Safe Delete***

Klik kanan pada file java ***hello1.java*** lalu pilih ***refactor*** lalu pilih ***Safe Delete***

*Akan muncul tampilan seperti di bawah ini :*

Contreng “***Search In Comments***” lalu klik ***Refaktor***

Menghapus class ***Hello1.java*** di package ***sampleprogram***

1. **Tugas Dan Pembahasan**

Carilah Script java di Internet dan eksekusi sama dengan latihan di atas !

URL : <http://kendorkentut.blogspot.com/p/pemrograman-berorientiasi-objek_17.html>

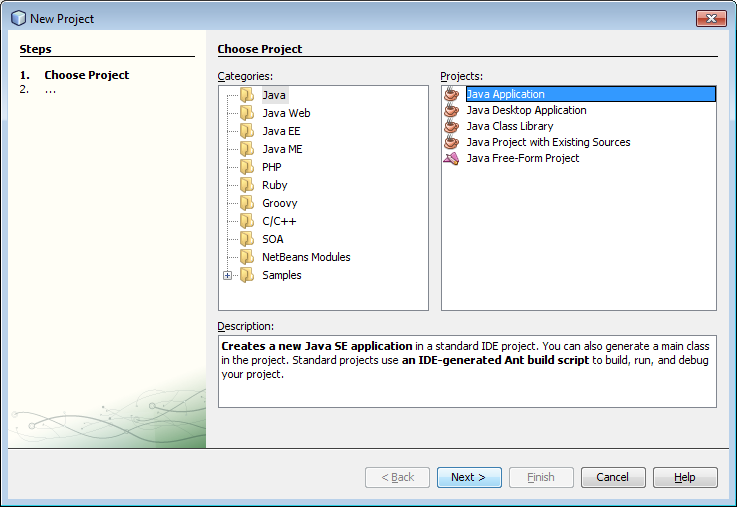
Didalam URL ini terdapat 5 file java yaitu :

1. Motor.java
2. Kendaraan.java
3. Pesawat.java
4. Mobil.java
5. Main.java

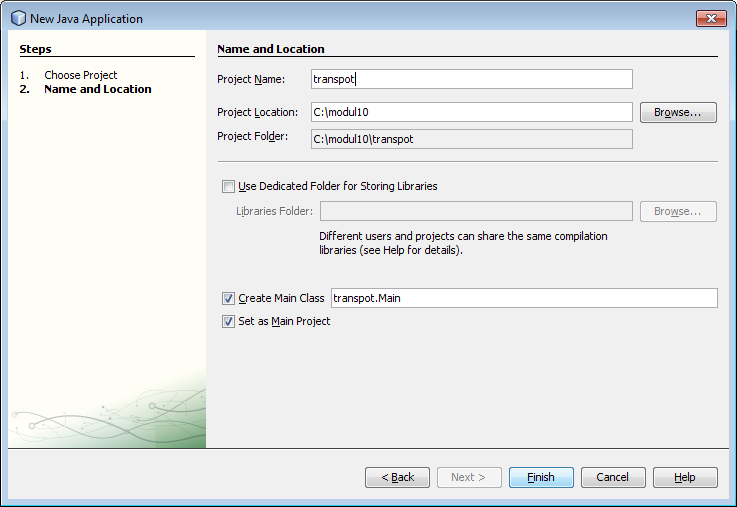
***Eksekusi :***

* Membuat Project Baru , klik ***File*** lalu pilih ***New Project***

*Akan muncul tampilan seperti di bawah ini :*



* Pilih java – Java Application lalu tekan Next

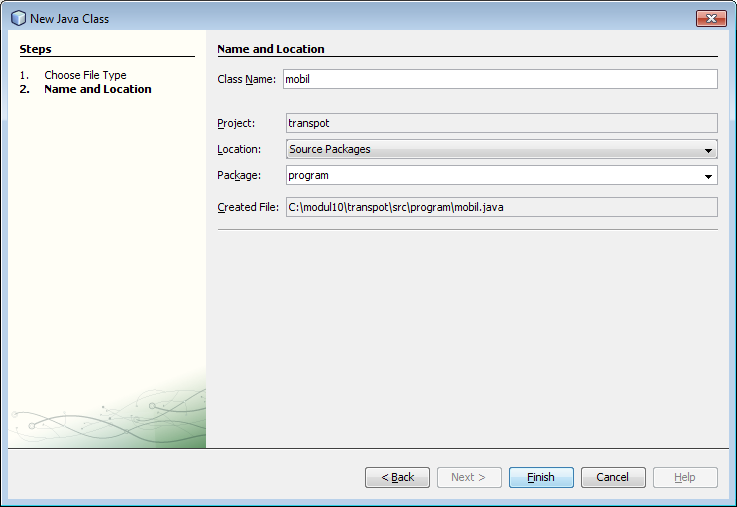


* Project name adalah ***transpot*** dan Project Location di simpan di directori C dengan nama folder modul10 lalu Finish.

***Membuat class dan Packages***

* Klik kanan pada project **transpot**, pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



* Class name diisikan mobil akan otomatis ke save dengan mobil.java dan packagenya diisikan program lalu klik Finish.
* Di dalam class mobil.java berisikan di bawah ini :

package transpot; ***//nama package “transpot”***

public class mobil extends kendaraan {***//nama class dan extends***

public mobil(int posisi) {***//variabel posisi dan type data***

super(posisi); ***//posisi***

}

@Override

public void Move() {***//move***

setPosisi(getPosisi() + 70); ***//posisi ya 70***

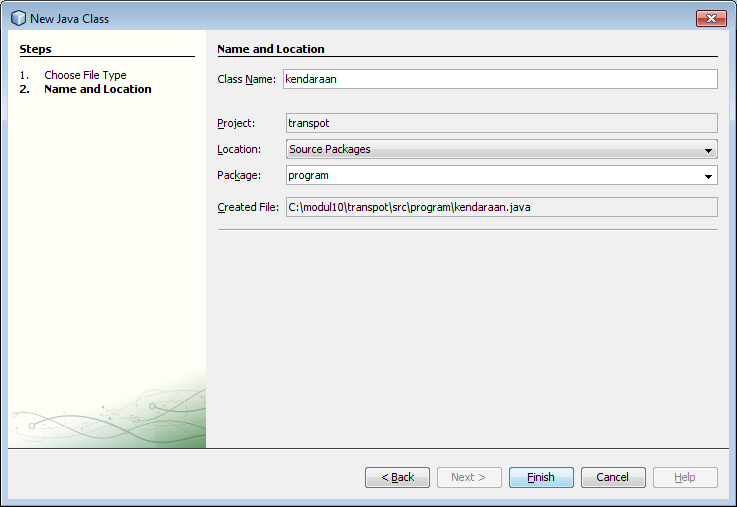
}

}***//penutup program***

***Tambah class***

* Klik kanan pada package **program**, pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



* Class name diisikan kendaraan akan otomatis ke save dengan kendaraan.java dan packagenya diisikan program lalu klik Finish.
* Di dalam class kendaraan.java berisikan di bawah ini :

package program;***//nama package “program”***

public class kendaraan {***//nama class “kendaraan”***

private int posisi;

public kendaraan(int posisi) {

this.posisi = posisi;

}***//variabel posisi***

public void Move() {

return;

}***//cetak dan pindah***

public int getPosisi() {

return posisi;

}***//cetak posisi***

public void setPosisi(int posisi) {

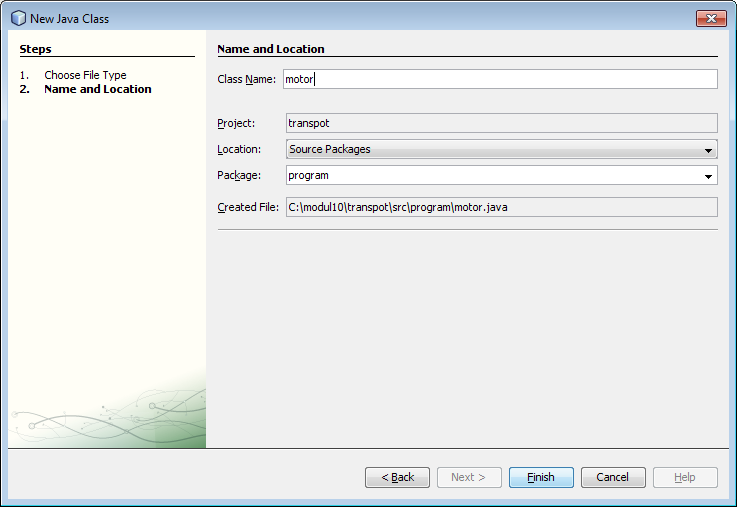
this.posisi = posisi;

}}***//setting posisi***

***Tambah class***

* Klik kanan pada package **program**, pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



* Class name diisikan motor akan otomatis ke save dengan motor.java dan packagenya diisikan program lalu klik Finish.
* Di dalam class motor.java berisikan di bawah ini :

package program; ***//nama package “program”***

public class motor extends kendaraan {***//nama class “motor”***

public motor(int posisi) {

super(posisi);

***//setting posisi***

}

@Override

public void Move() {***//move***

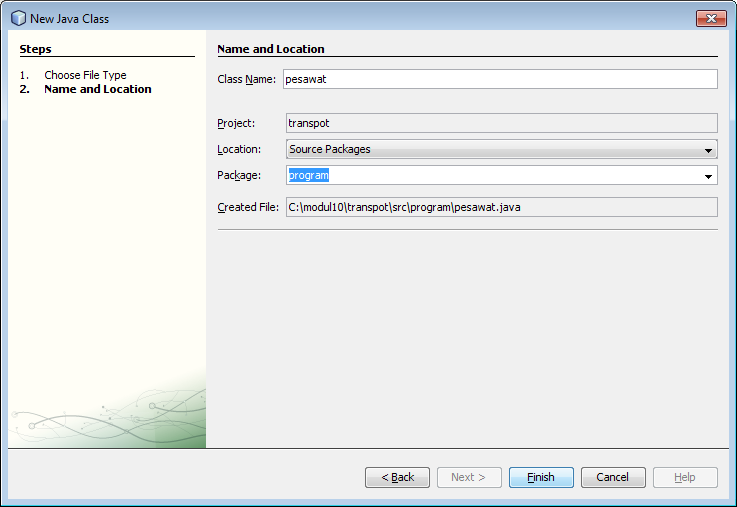
setPosisi(getPosisi() + 30); ***//posisi ya 30***

}}

***Tambah class***

* Klik kanan pada package **program**, pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



* Class name diisikan pesawat akan otomatis ke save dengan pesawat.java dan packagenya diisikan program lalu klik Finish.
* Di dalam class pesawat.java berisikan di bawah ini :

package program; ***//nama package “program”***

public class pesawat extends kendaraan {***//nama class “pesawat”***

public pesawat(int posisi) {

super(posisi);

***//setting posisi***

}

@Override

public void Move() {***//move***

setPosisi(getPosisi() + 300); ***//posisi ya 300***

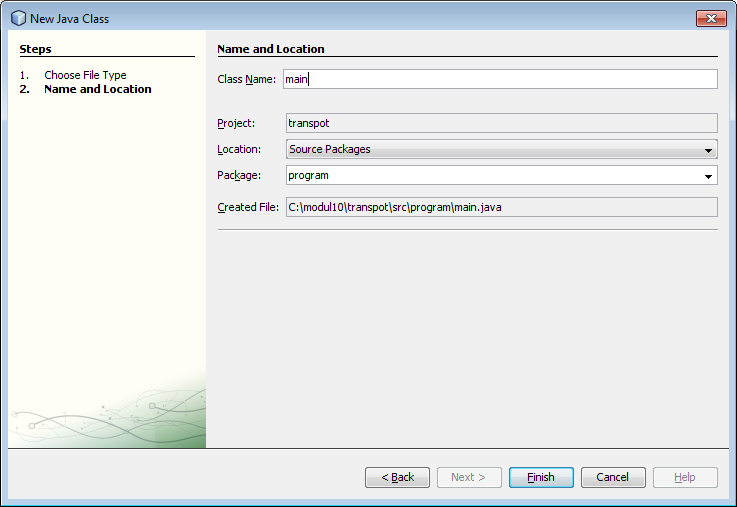
}

}

***Tambah class***

* Klik kanan pada package **program**, pilih ***New*** lalu Pilih ***Java Class***

*Akan muncul tampilan seperti di bawah ini :*



* Class name diisikan main akan otomatis ke save dengan main.java dan packagenya diisikan program lalu klik Finish.
* Di dalam class main.java berisikan di bawah ini :

package program; ***//nama package “program”***

public class main {

public static void main(String[] args) {***//nama class “pesawat”***

kendaraan kendaraan[] = new kendaraan[3];

int posisi = 10; ***//setting posisi***

kendaraan[0] = new motor(posisi); ***//posisi kendaraan motor***

kendaraan[1] = new mobil(posisi); ***//posisi kendaraan mobil***

kendaraan[2] = new pesawat(posisi); ***//posisi kendaraan pesawat***

if (kendaraan[0] instanceof motor) {

System.out.println("Kendaraan[0] merupakan instanceof Motor");

}***//setting output Kendaraan[0]***

if (kendaraan[1] instanceof motor) {

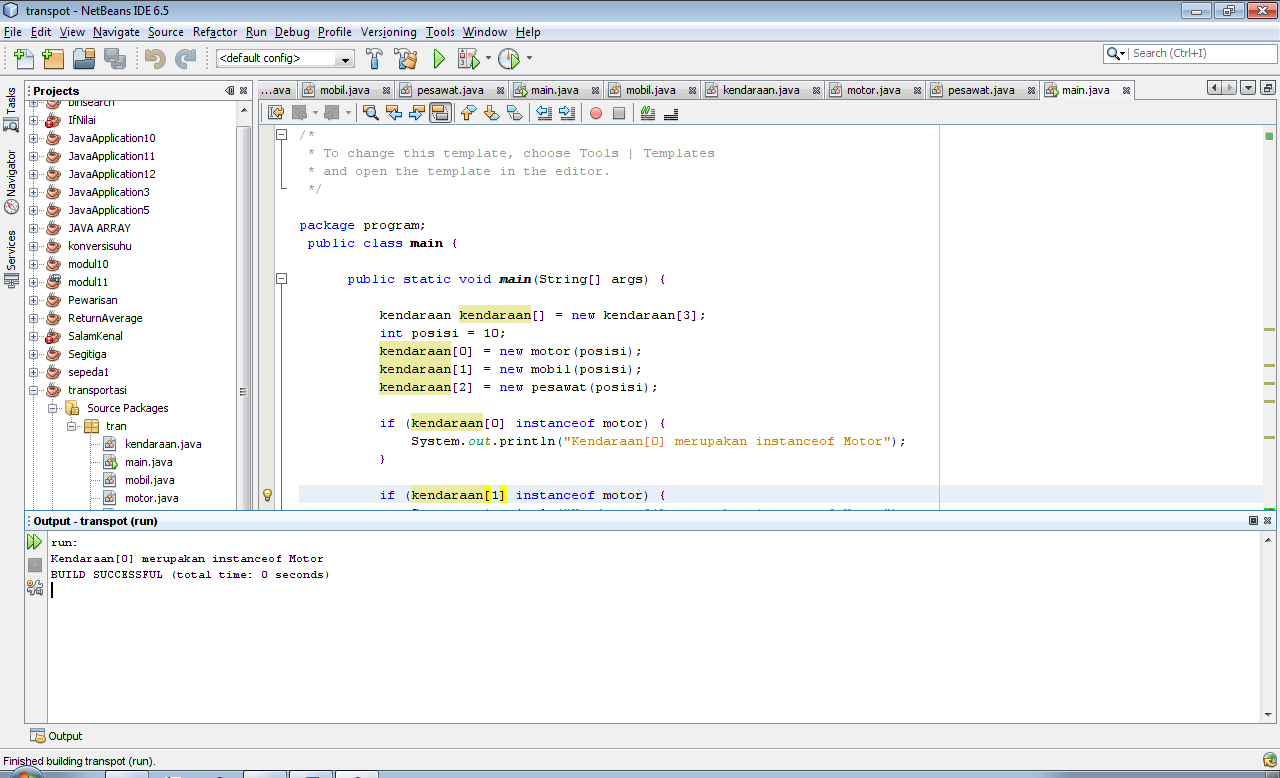
System.out.println("Kendaraan[1] merupakan instanceof Motor");

***//setting output Kendaraan[1]***

} } }

Setelah itu Run Program lagi dengan cara klik kanan pada class main.java lalu pilih Run File atau Shift+F6

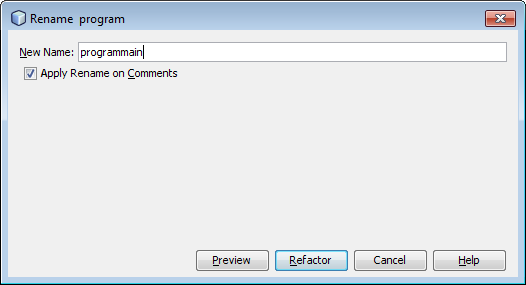
**Hasil Output main.java**



***Refaktor rename package***

Selanjutya klik kanan pada package ***program*** pilih Refaktor lalu pilih ***rename***

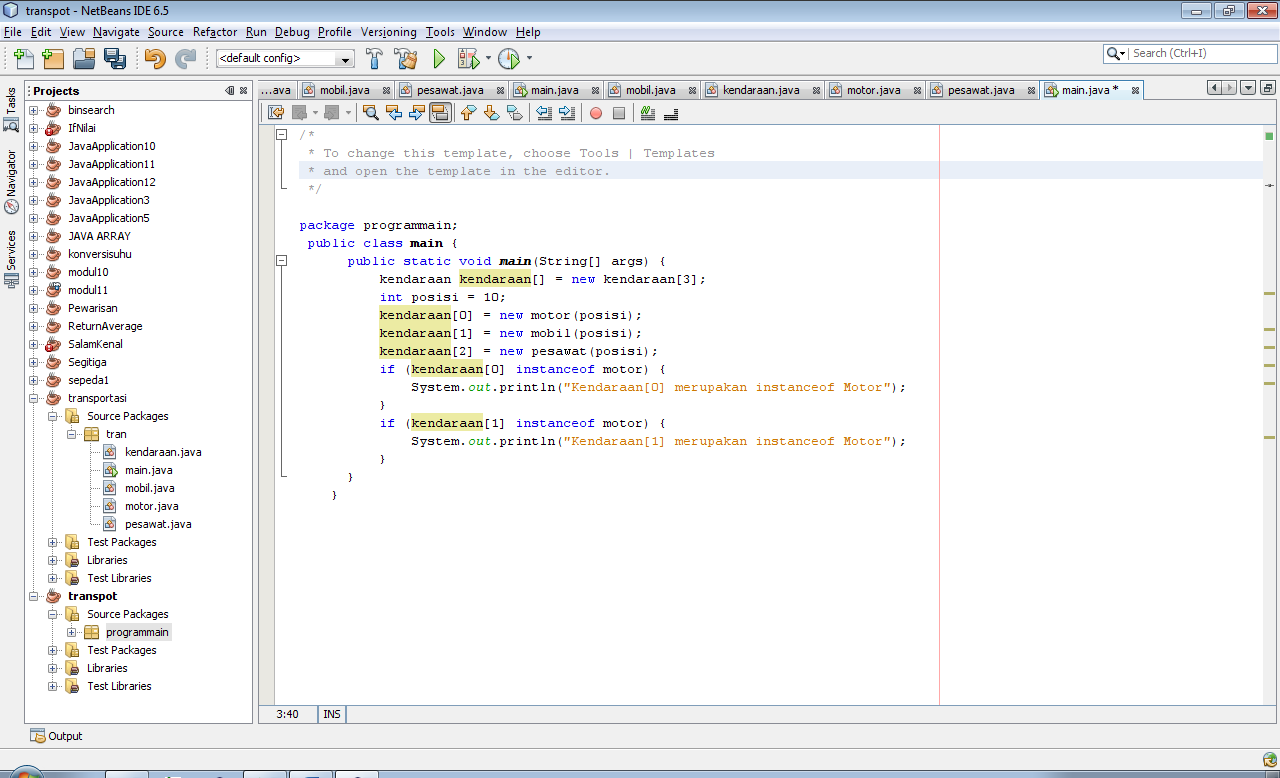
*Akan muncul tampilan seperti di bawah ini :*



Ubah New Name yang awalnya program menjadi ***programmain*** lalu klik ***Refactor***

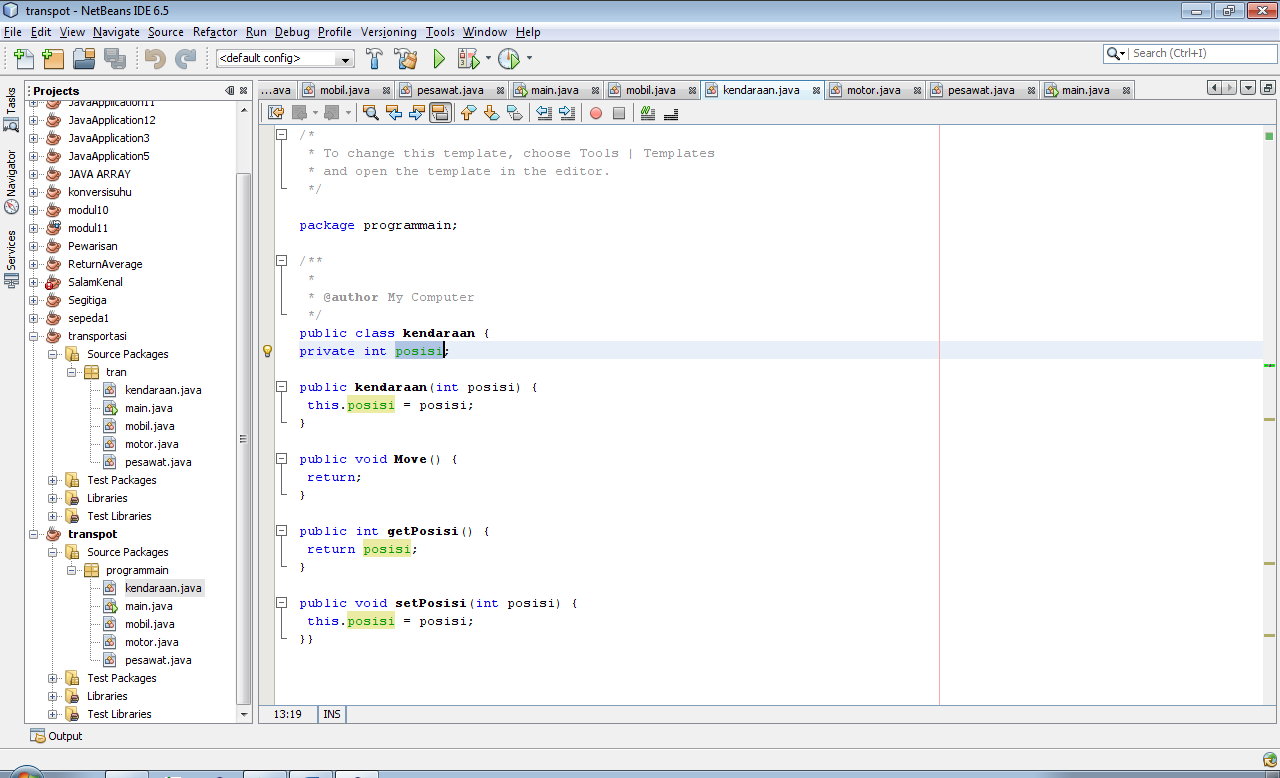
Nama Package yang awalnya program berubah jadi programmain dan secara otomatis kalau menggunakan Refactor nama package yang ada di script main.java pasti ikut berubah.

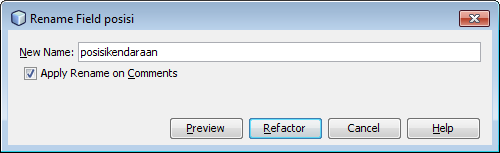
**Tampilan program menjadi :**



***Merefactor script java di file java kendaraan.java yaitu posisi***

Klik kanan pada script java ***posisi*** lalu pilih ***refaktor*** lalu pilih ***rename.*** Seperti tampilan di bawah ini :

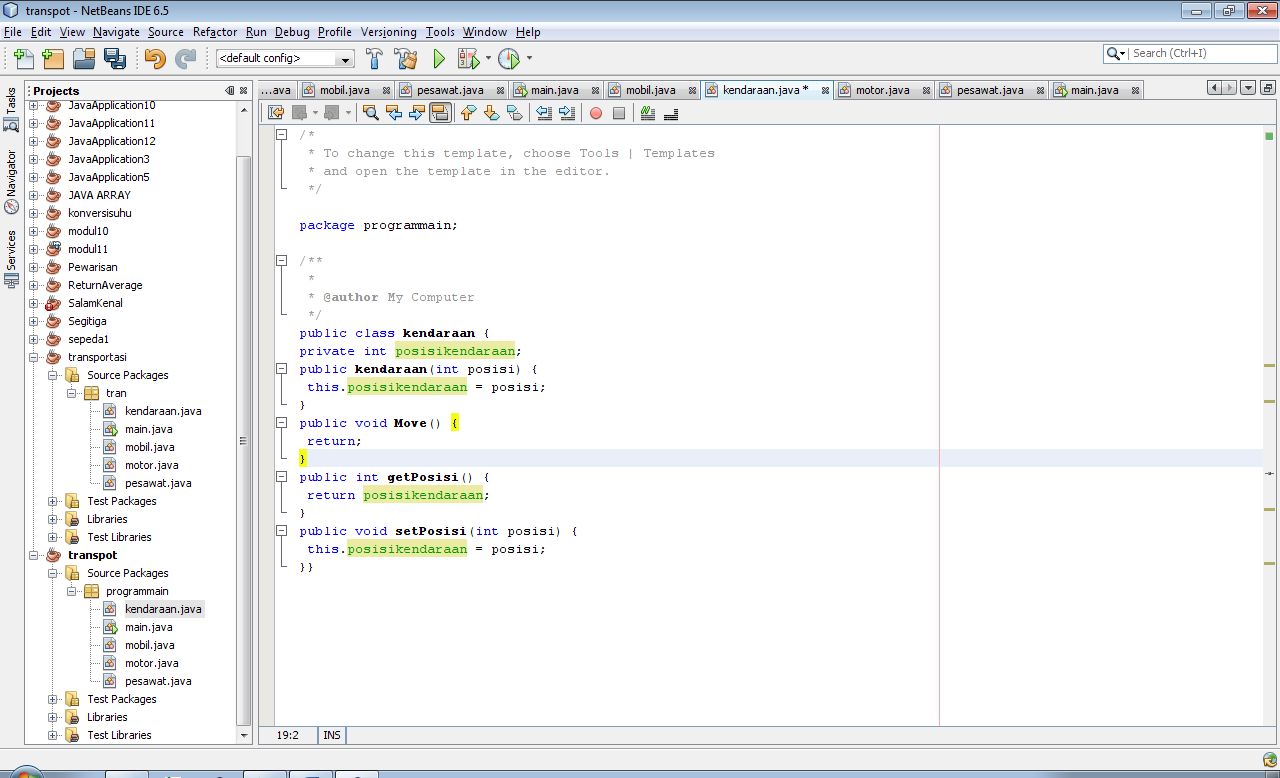


*Akan muncul tampilan seperti di bawah ini :*

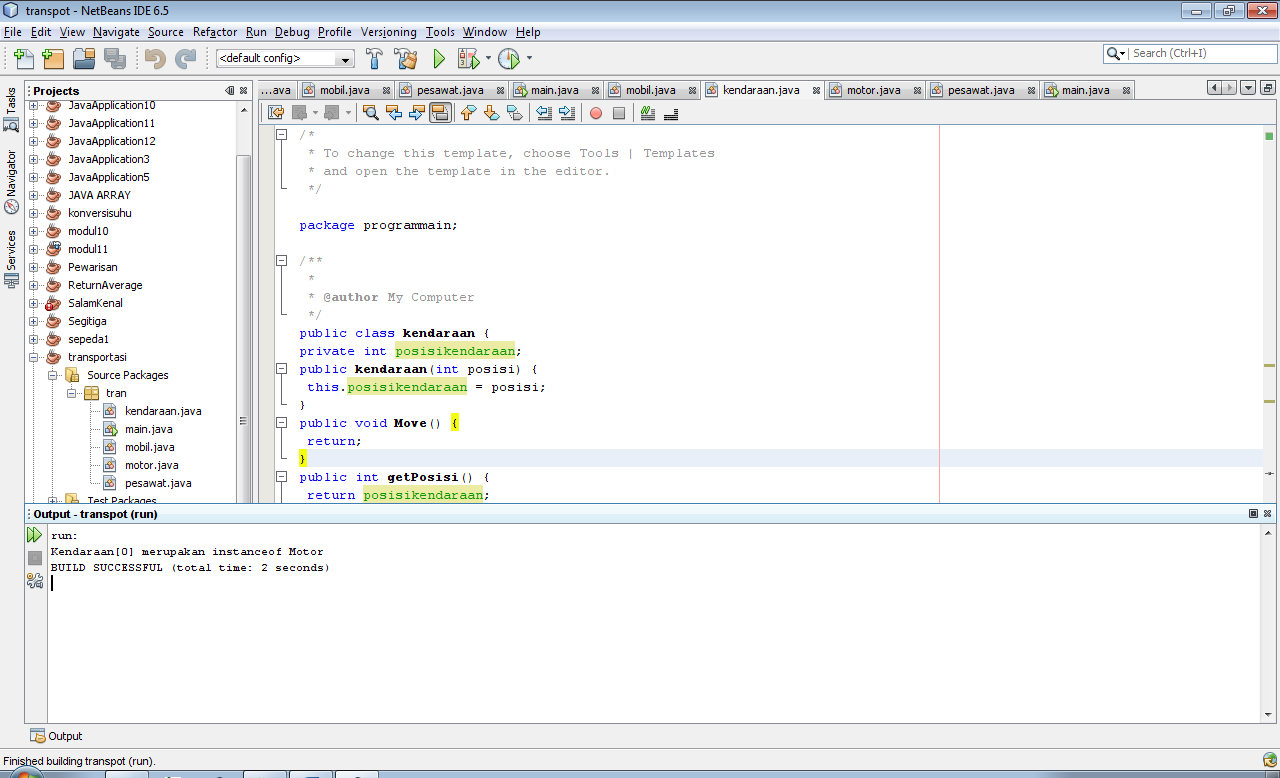
New Name ya diisi ***posisikendaraan*** yang awalnya hanya ***posisi*** lalu centang “***Apply Rename On Comments***“dan klik ***Refactor***

Akan terjadi perubahan pada file ***kendaraan.java*** yaitu script ***posisi*** berubah semua script menjadi ***posisikendaraan*** .

**Tampilanya :**



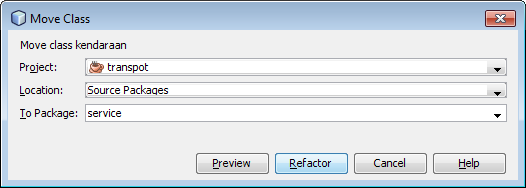
Setelah itu Run Program lagi dengan cara klik kanan pada class main.java lalu pilih Run File atau Shift+F6

**Hasil Output main.java**

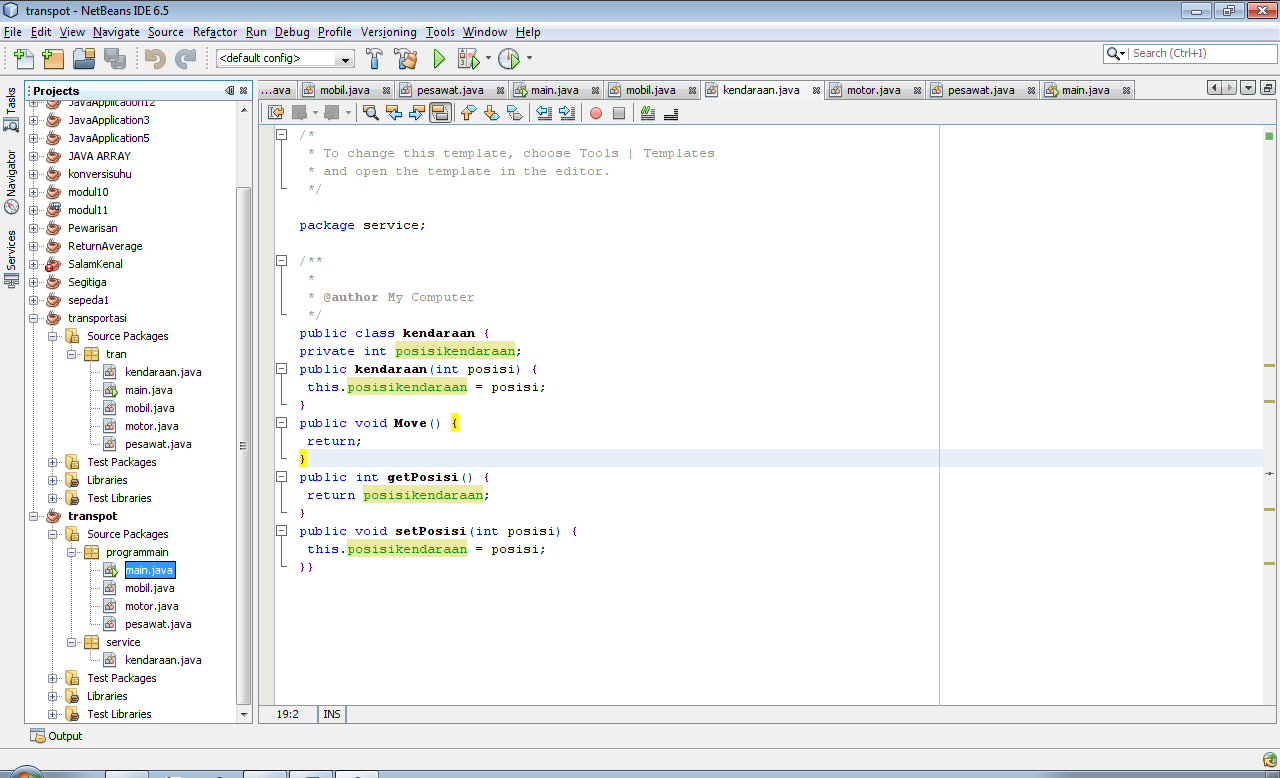
***Refaktore file java kendaraan.java pindahkan ke package service***

Klik kanan pada file java ***kendaraan.java*** lalu pilih ***refactor*** lalu pilih ***move***

*Akan muncul tampilan seperti di bawah ini :*

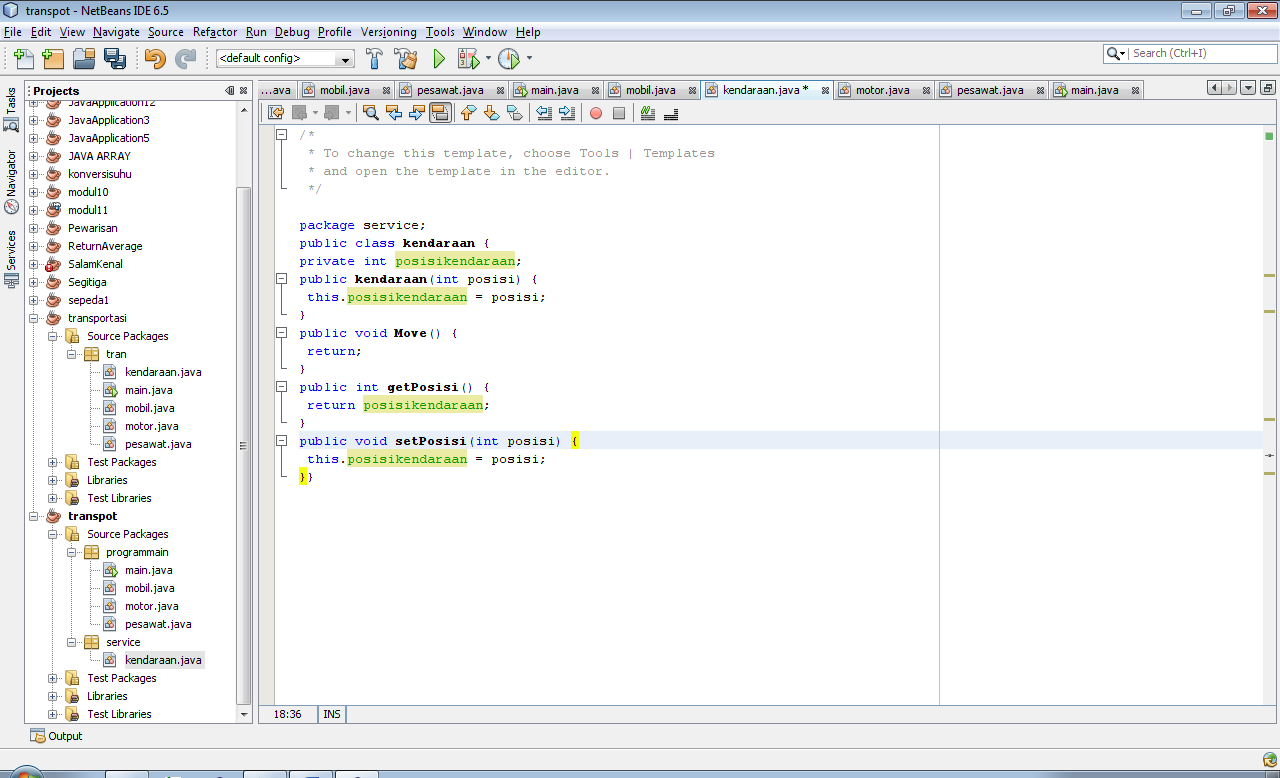


To Package yang awal ya ***programmain*** di isi ***service*** lalu klik ***Refaktor***

***Hasilya :***

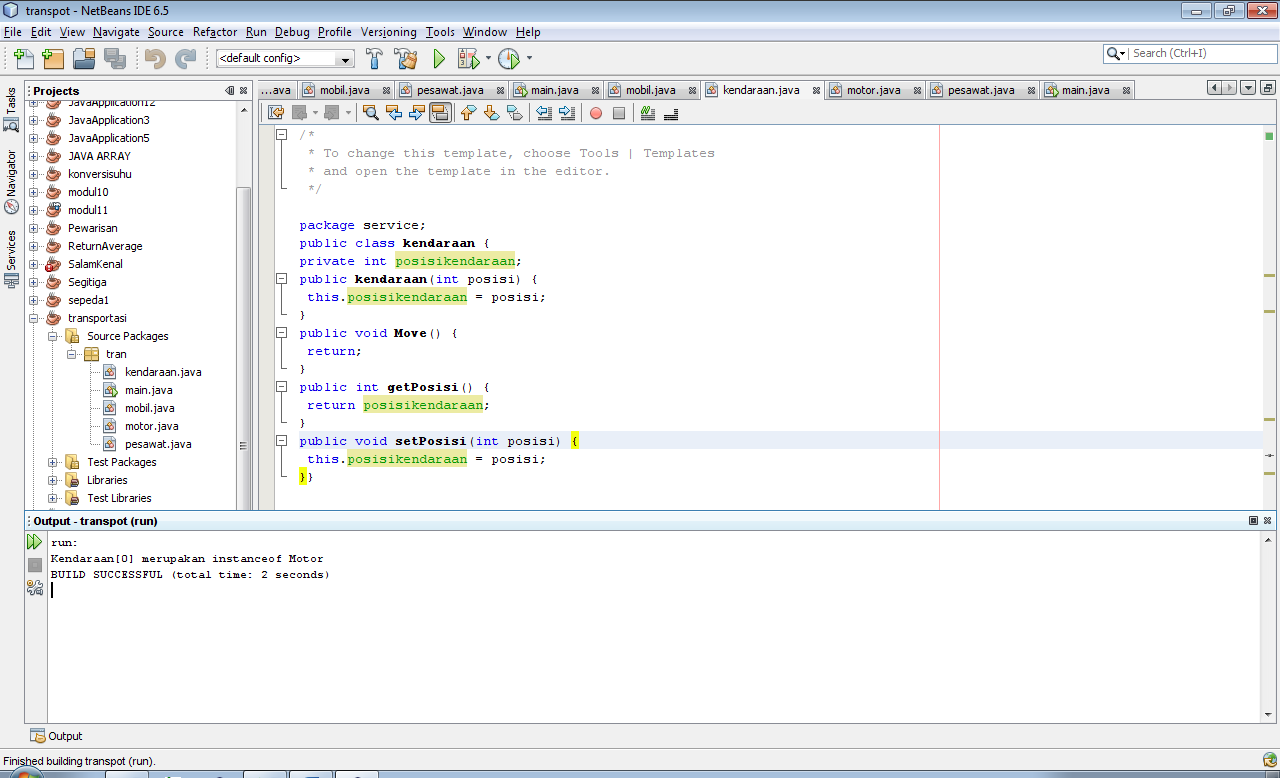
File java ***kendaraan.java*** berada di package ***service***

Dan pada file ***kendaraan.java*** ada perubahan script secara otomatis setelah melakukan cara seperti di atas, yaitu nama package ya secara otomatis akan berubah , seperti di bawah ini :



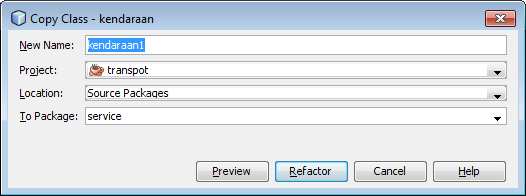
Setelah itu Run Program lagi dengan cara klik kanan pada class main.java lalu pilih Run File atau Shift+F6

**Hasil Output main.java**



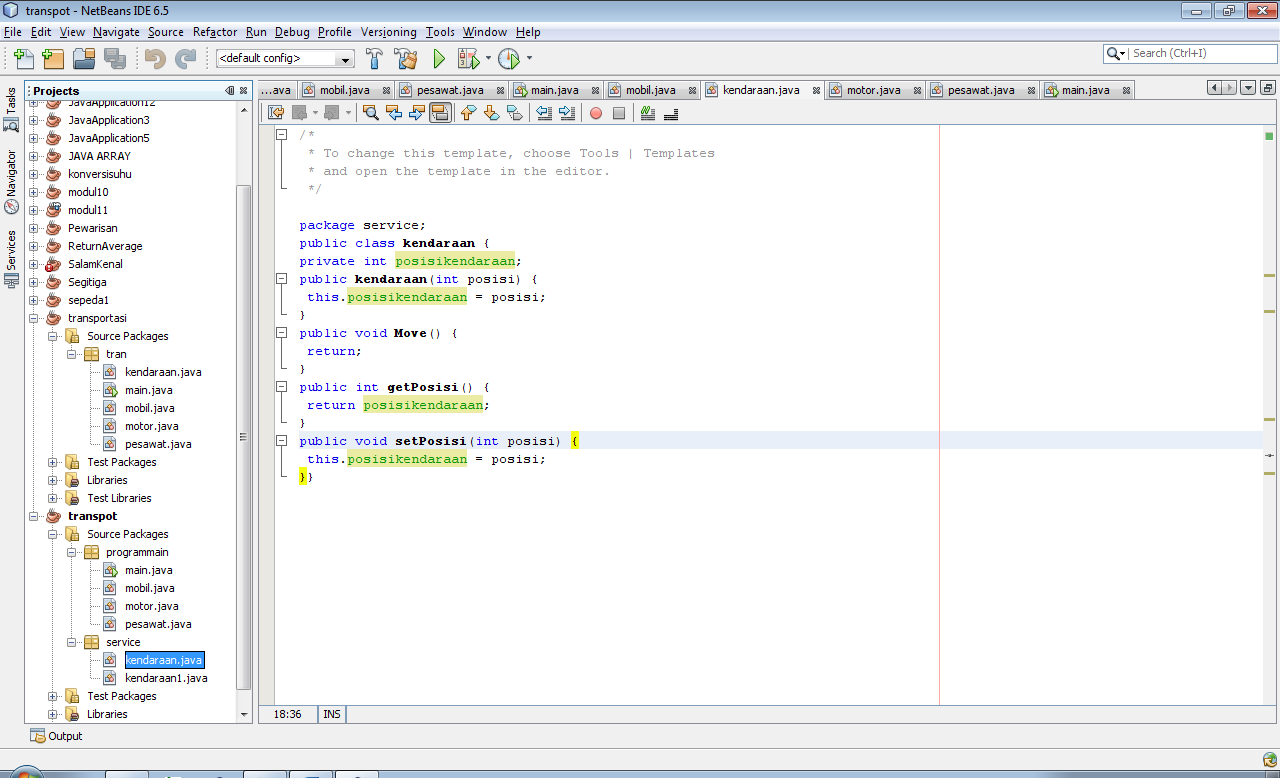
***Refactor file java kendaraan.java menduplikat menggunakan Copy***

Klik kanan pada file java ***kendaraan.java*** lalu pilih ***refactor*** lalu pilih ***Copy***

*Akan muncul tampilan seperti di bawah ini :*

New Name yang awal ya ***kendaraan*** diisi secara otomatis ***kendaraan1*** To Package tetap di **service** lalu klik ***Refaktor***

***Hasilya :***

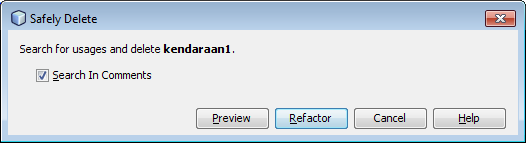


Ada class ***kendaraan1.java*** di package ***service***

***Refactor file java kendaraan1.java menghapus menggunakan Safe Delete***

Klik kanan pada file java ***kendaraan1.java*** lalu pilih ***refactor*** lalu pilih ***Safe Delete***

*Akan muncul tampilan seperti di bawah ini :*



Contreng “***Search In Comments***” lalu klik ***Refaktor***

Menghapus class ***kendaraan1.java*** di package ***service***

1. **Kesimpulan**

Menggunakan Refaktor memudahkan user atau pengguna untuk melalukan perubahan script dengan cara yang tidak satu per satu atau sekali merubah langsung terganti semua , mengcopy file , memindahkan file ke package yang baru , menduplikat file java dan menghapus file java yang di inginkan.

1. **Listing Terlampir**