

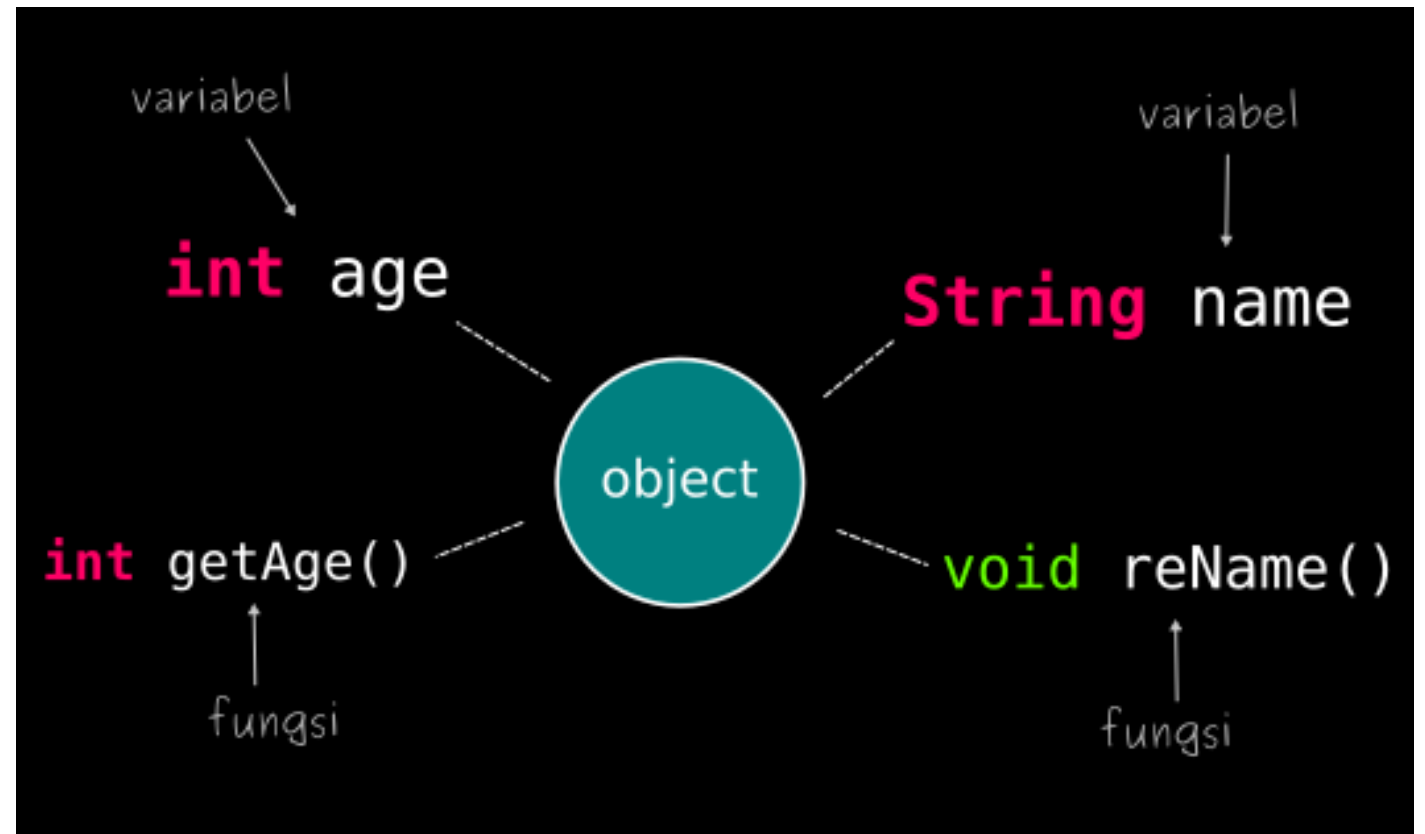
OOP (*OBJECT ORIENTED PROGRAMMING*)



■ APA ITU OOP?

- OOP (*Object Oriented Programming*) atau dalam bahasa indonesia dikenal dengan pemrograman berorientasikan objek (PBO) merupakan sebuah paradigma atau teknik pemrograman yang **berorientasikan Objek**.
- Pada OOP, Fungsi dan variabel **dibungkus** dalam sebuah **objek** atau *class* yang dapat saling berinteraksi, sehingga membentuk sebuah program.

- Dengan demikian, tidak akan ada lagi kode yang “berantakan.”
- Semua akan kita bungkus dalam objek.
- Tapi sebelum membuat objek, kita harus membuat class dulu...



CLASS DAN OBJECT

Class sebenarnya bertugas untuk mengumpulkan prosedur/fungsi dan variabel dalam satu tempat.

Contoh:

Kita ingin membuat game sederhana, di dalamnya ada kucing dan makanan.

Prosedural	OOP
<pre>var catMood = "happy"; var catEnergy = 80; var makanan = "Fish"; catMove() catPlay() catJump() catEat()</pre>	<pre>class Kucing { var mood; var energi; lari() loncat() makan() } class Makanan { var nama; var rasa; hide() }</pre>

- Dengan class, kita bisa menentukan.. mana variabel dan prosedur untuk kucing dan makanan.
- Class ini nanti yang akan kita pakai untuk membuat objek.

Class adalah rancangan atau *blue print* dari sebuah objek.

- Sedangkan objek adalah sebuah variabel yang merupakan *instance* atau perwujudan dari *Class*.
- *Instance* bisa diartikan sebagai wujud dari *class*.
- *Class* berisi definisi variabel dan fungsi yang menggambarkan sebuah objek.
- **Variabel** disebut **atribut** atau **properti**;
- **Fungsi** disebut **method**.

CONTOH *CLASS*:

```
class NamaClass {  
    String atribut1;  
    String atribut2;  
  
    void namaMethod(){ ... }  
    void namaMethodLain(){ ... }  
}
```

- Lalu, Kita biasanya membuat objek (*instance*) seperti ini:

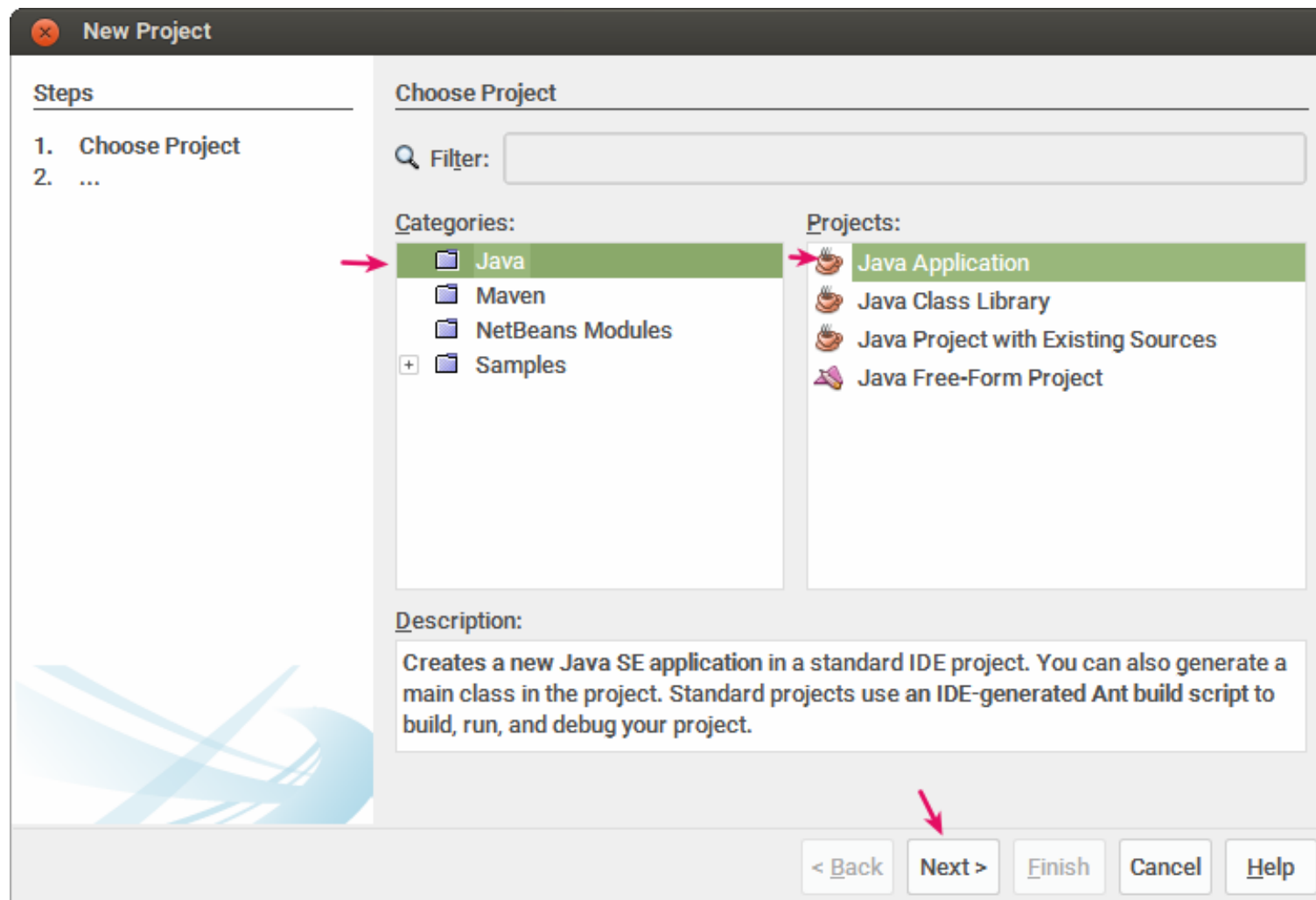
```
NamaClass namaObj = new NamaClass();
```

- Kata kunci new berfungsi untuk membuat objek baru dari class tertentu.
- Setelah membuat objek, kita bisa mengakses atribut dan method dari objek tersebut.

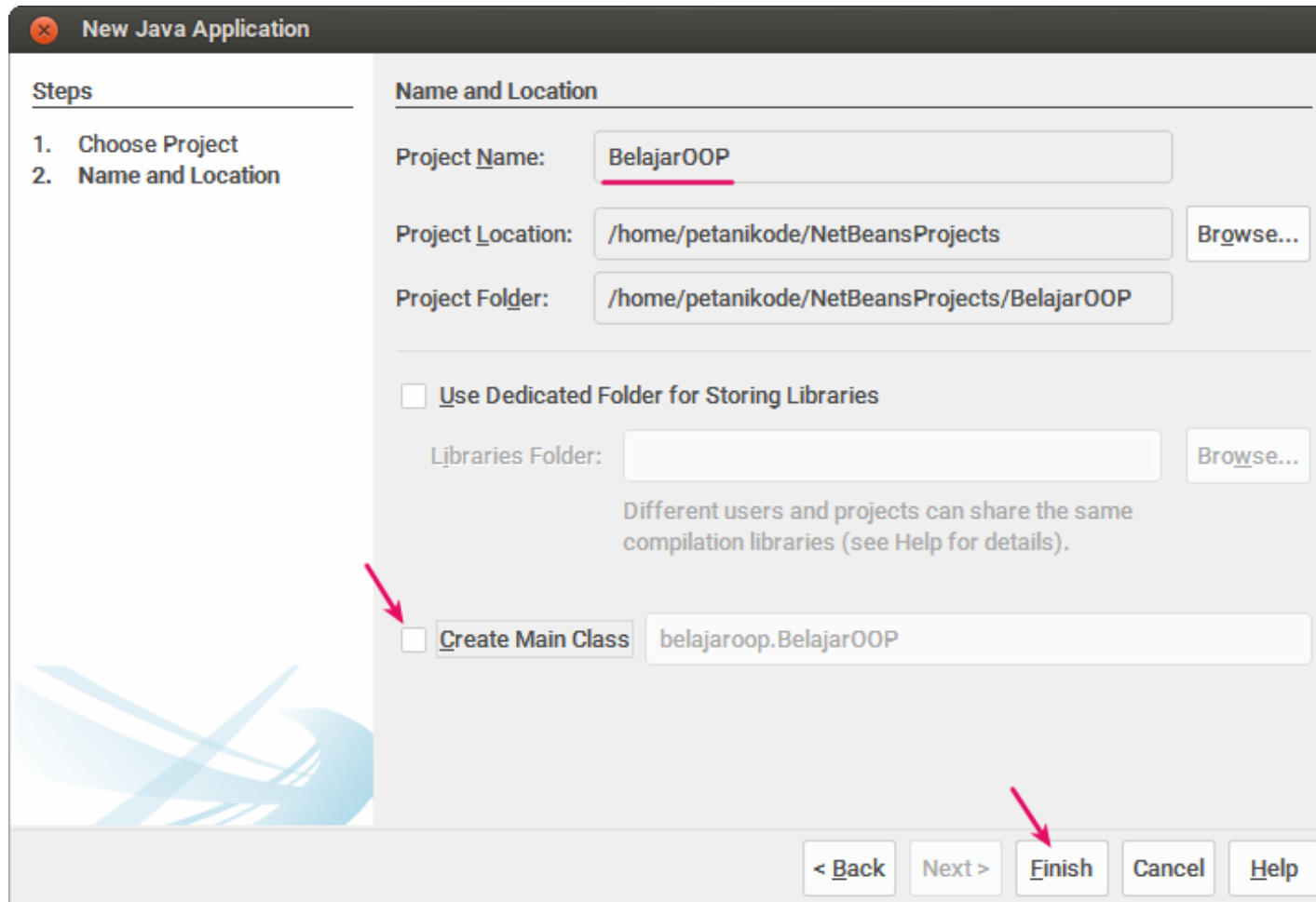


CONTOH:

- **Contoh Program OOP**
- Kita akan menggunakan Netbeans, silahkan buat project baru.



ISI NAMA PROJECT-NYA DENGAN “BELAJAROOP” DAN JANGAN CENTANG “CREATE MAIN CLASS”.



New Java Application

Steps

1. Choose Project
2. Name and Location

Name and Location

Project Name: BelajarOOP

Project Location: /home/petanikode/NetBeansProjects **Browse...**

Project Folder: /home/petanikode/NetBeansProjects/BelajarOOP

☐ Use Dedicated Folder for Storing Libraries

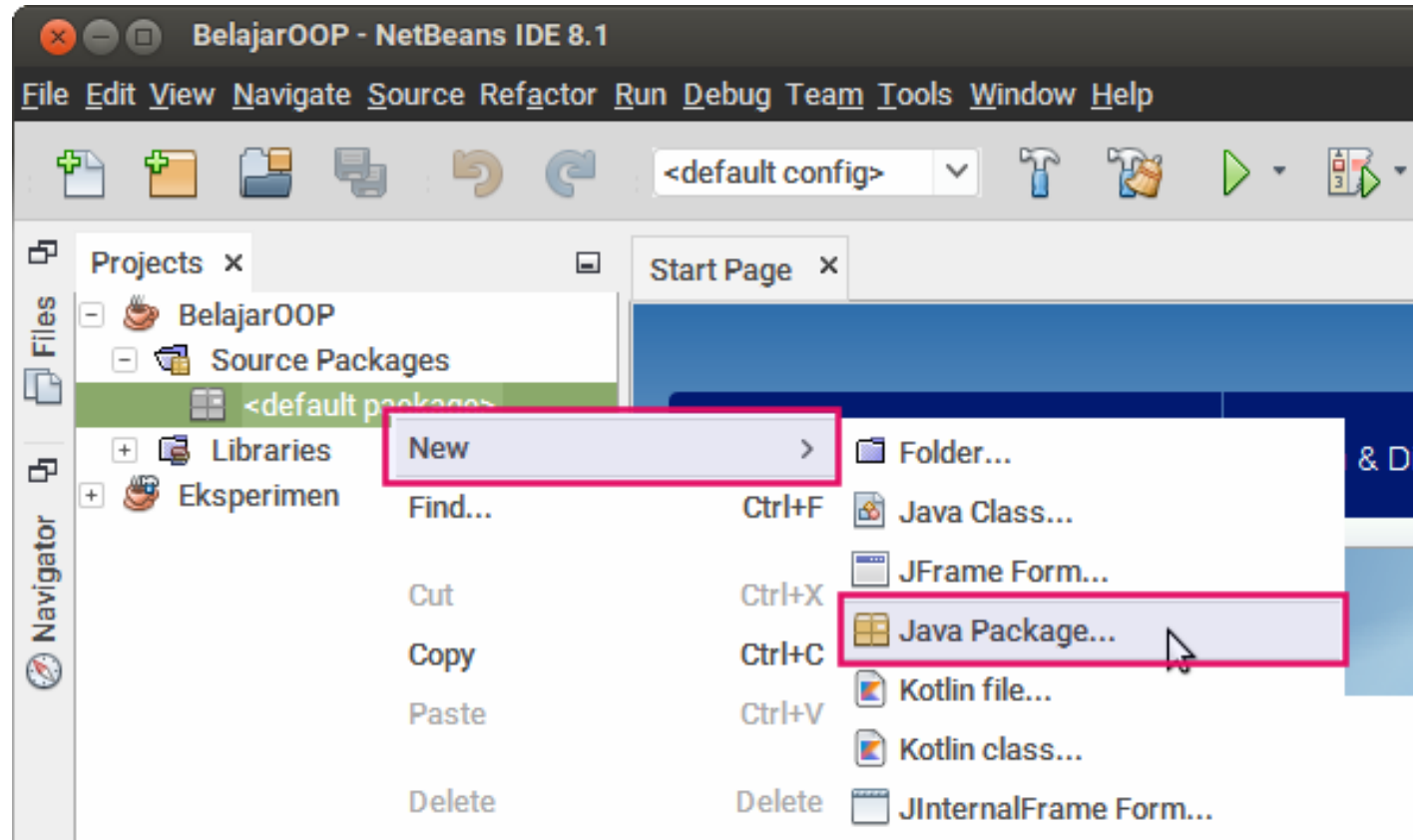
Libraries Folder: **Browse...**

Different users and projects can share the same compilation libraries (see Help for details).

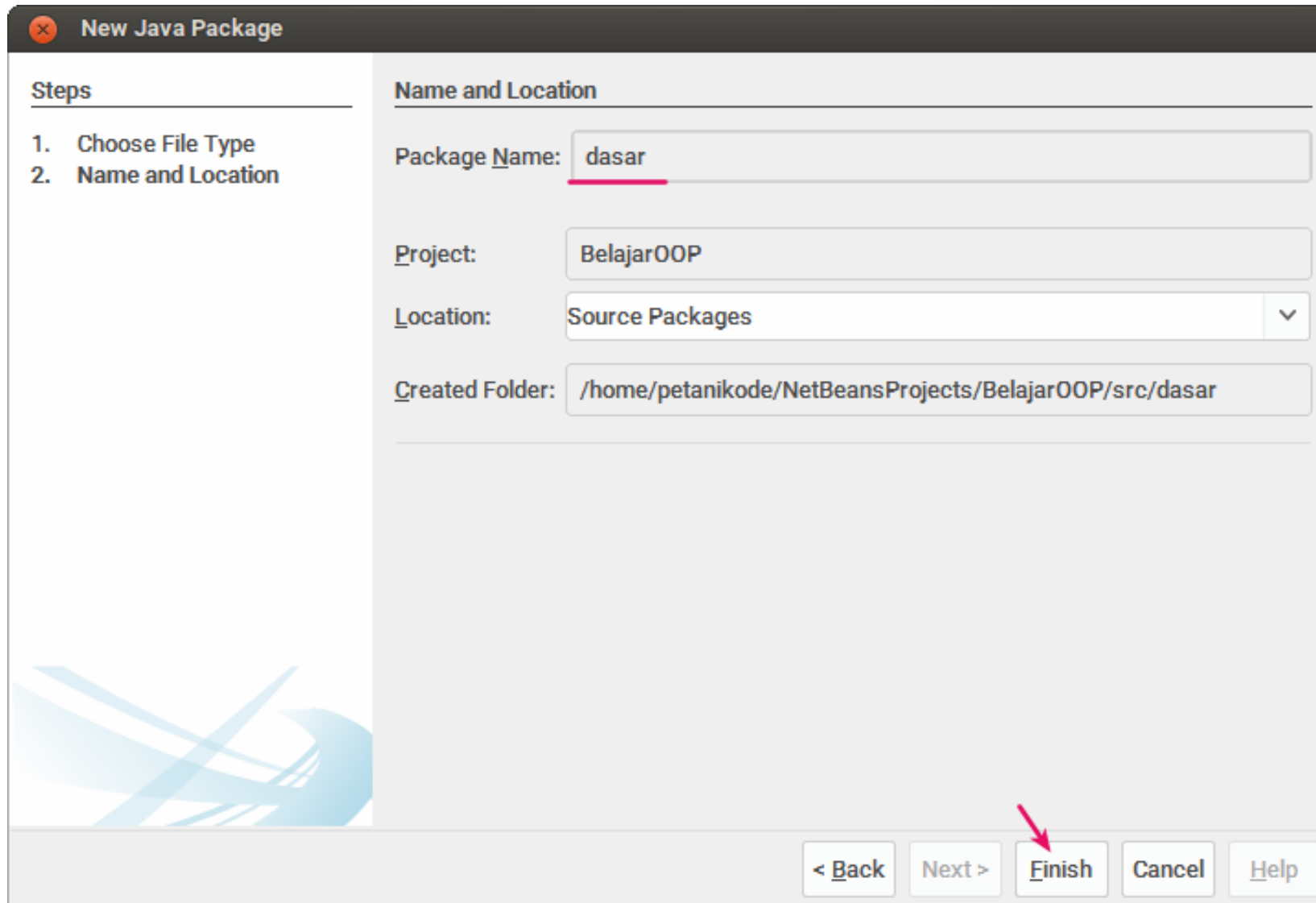
☐ Create Main Class belajaroop.BelajarOOP

< **B**ack Next > **F**inish Cancel **H**elp

SETELAH ITU, SILAHKAN BUAT PACKAGE BARU. KLIK KANAN PADA <DEFAULT PACKAGE>, LALU PILIH NEW->JAVA PACKAGE.



ISI NAMA PACKAGE DENGAN “DASAR”:



New Java Package

Steps

1. Choose File Type
2. Name and Location

Name and Location

Package Name: dasar

Project: BelajarOOP

Location: Source Packages ▼

Created Folder: /home/petanikode/NetBeansProjects/BelajarOOP/src/dasar

< Back Next > Finish Cancel Help

- Setelah itu, di dalam package dasar, silahkan buat class baru bernama Player.java.
- Klik kanan pada package dasar lalu pilih New->Java Class. Setelah itu, isi namanya dengan Player.java.
- Silahkan tulis kode pada Player.java seperti ini:

```
package dasar;

public class Player {

    // definisi atribut
    String name;
    int speed;
    int healthPoin;

    // definisi method run
    void run(){
        System.out.println(name + " is running...");
        System.out.println("Speed: " + speed);
    }

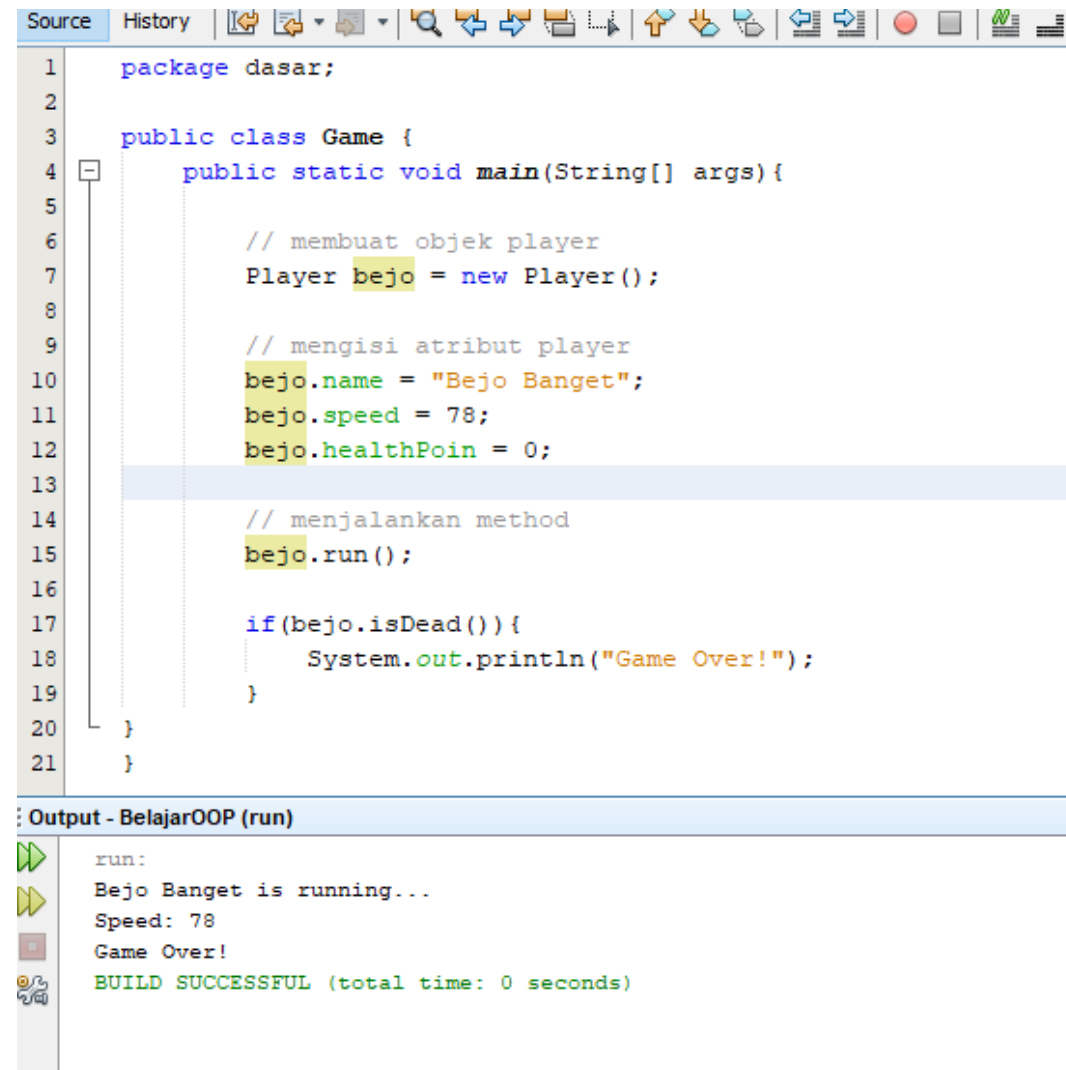
    // definisi method isDead untuk mengecek nilai kesehatan (healthPoin)
    boolean isDead(){
        if(healthPoin <= 0) return true;
        return false;
    }

}
```

BERIKUTNYA, BUAT CLASS BARU LAGI BERNAMA GAME.JAVA DAN ISI KODENYA SEPERTI INI:

```
1 package dasar;
2
3 public class Game {
4     public static void main(String[] args){
5
6         // membuat objek player
7         Player bejo = new Player();
8
9         // mengisi atribut player
10        bejo.name = "Bejo Banget";
11        bejo.speed = 78;
12        bejo.healthPoin = 0;
13
14        // menjalankan method
15        bejo.run();
16
17        if(bejo.isDead()){
18            System.out.println("Game Over!");
19        }
20    }
21 }
```

COBA EKSEKUSI GAME.JAVA DENGAN KLIK KANAN PADA GAME.JAVA LALU PILIH RUN FILE. MAKA HASILNYA:



The screenshot displays an IDE window with a Java source file. The code defines a package 'dasar' and a public class 'Game'. Inside the 'Game' class, there is a 'main' method that creates a 'Player' object named 'bejo', sets its name to 'Bejo Banget', speed to 78, and health points to 0. It then calls 'bejo.run()' and checks if the player is dead, printing 'Game Over!' if so. The IDE's output window at the bottom shows the successful execution of the program, displaying the player's name, speed, and the 'Game Over!' message, followed by a 'BUILD SUCCESSFUL' status.

```
1 package dasar;
2
3 public class Game {
4     public static void main(String[] args){
5
6         // membuat objek player
7         Player bejo = new Player();
8
9         // mengisi atribut player
10        bejo.name = "Bejo Banget";
11        bejo.speed = 78;
12        bejo.healthPoin = 0;
13
14        // menjalankan method
15        bejo.run();
16
17        if(bejo.isDead()){
18            System.out.println("Game Over!");
19        }
20    }
21 }
```

Output - BelajarOOP (run)

run:
Bejo Banget is running...
Speed: 78
Game Over!
BUILD SUCCESSFUL (total time: 0 seconds)




COBA GANTI NILAI HEALTHPOIN MENJADI 0.

```
petani.healthPoin = 0;
```

PENJELASAN

Baik, kita mulai dari pembuatan class.

- Pertama-tama, kita membuat sebuah class dengan nama Player. Class ini mendefinisikan objek Player dalam Game dengan atribut sebagai berikut:
 - name adalah nama objek;
 - speed adalah kecepatan;
 - healthPoin adalah nilai kesehatan dari player, biasanya disingkat hp.

- 
- Lalu class Player memiliki method:
 - run() untuk menggerakkan player;
 - isDead() untuk mengecek kondisi kesehatan player. Method ini akan mengembalikan nilai true apabila nilai hp lebih kecil atau sama dengan nol (0), sebaliknya akan mengembalikan nilai false.

- Berikutnya kita membuat objek baru dari class Player pada class Game bernama bejo.

```
// membuat objek player  
Player bejo= new Player();
```

- Setelah itu mengisi atribut-atributny. Karena kalau tidak diisi akan bernilai Null dan bisa menyebabkan NullPointerException.

```
run:  
Bejo Banget is running...  
Speed: 78  
BUILD SUCCESSFUL (total time: 0 seconds)
```