Mengenal Object dan Class

Teguh Sutanto, M.Kom.,MCP.

Class

- Classes are the basic units of programming in the object-oriented paradigm
- Classes are essential, and they are basic parts of programs in object-oriented programming. They are used as templates to create objects.
- A class in Java may consist of five components:
  - Fields
  - Methods
  - Constructors
  - Static initializers
  - Instance initializers

Class (2)

- Fields and methods are also known as members of the class. Classes and interfaces can also be members of a class.
- A class can have zero or more class members. A class member of a class is also known as a nested class.
- Constructors are used to initialize objects of a class

Declaring a Class

[modifiers] class <class-name> {

// Isi(body) class

}

1. modifiers are optional; they are keywords that associate special meanings to the class declaration. A class declaration may have zero or more modifiers.
2. The keyword class is used to declare a class.
3. The class-name is a user-defined name of the class, which should be a valid Java identifier.
4. Each class has a body, which is specified inside a pair of braces ({}). The body of a class contains its different components, for example, fields, methods, etc

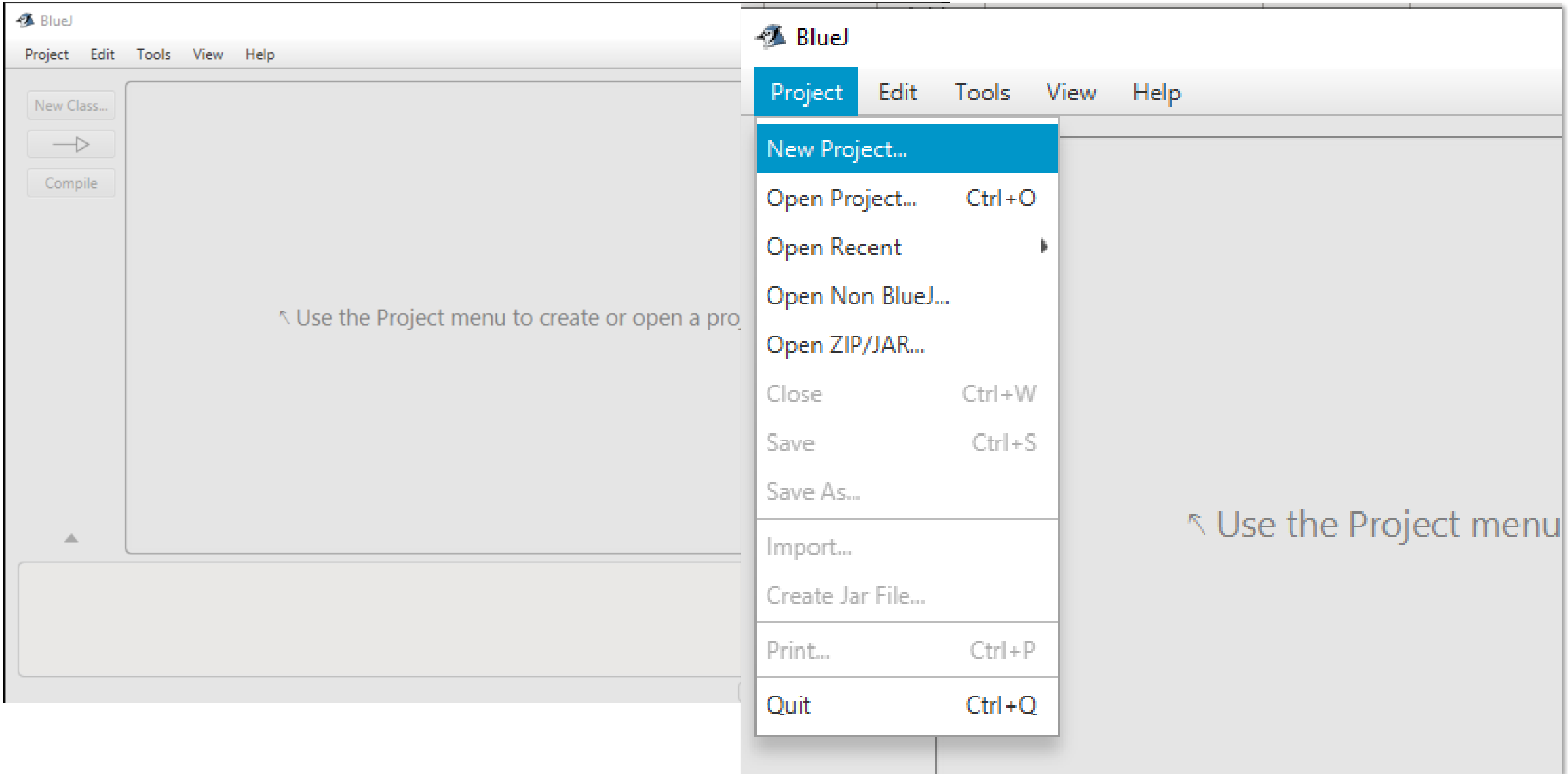
Declaring Fields in a Class

- Fields of a class represent properties (also called attributes) of objects of that class. Suppose every object of a Human class has two properties: a name and a gender. The Human class should include declarations of two fields: one to represent the name and one to represent the gender.
- The fields are declared inside the body of the class. The general syntax to declare a field in a class is

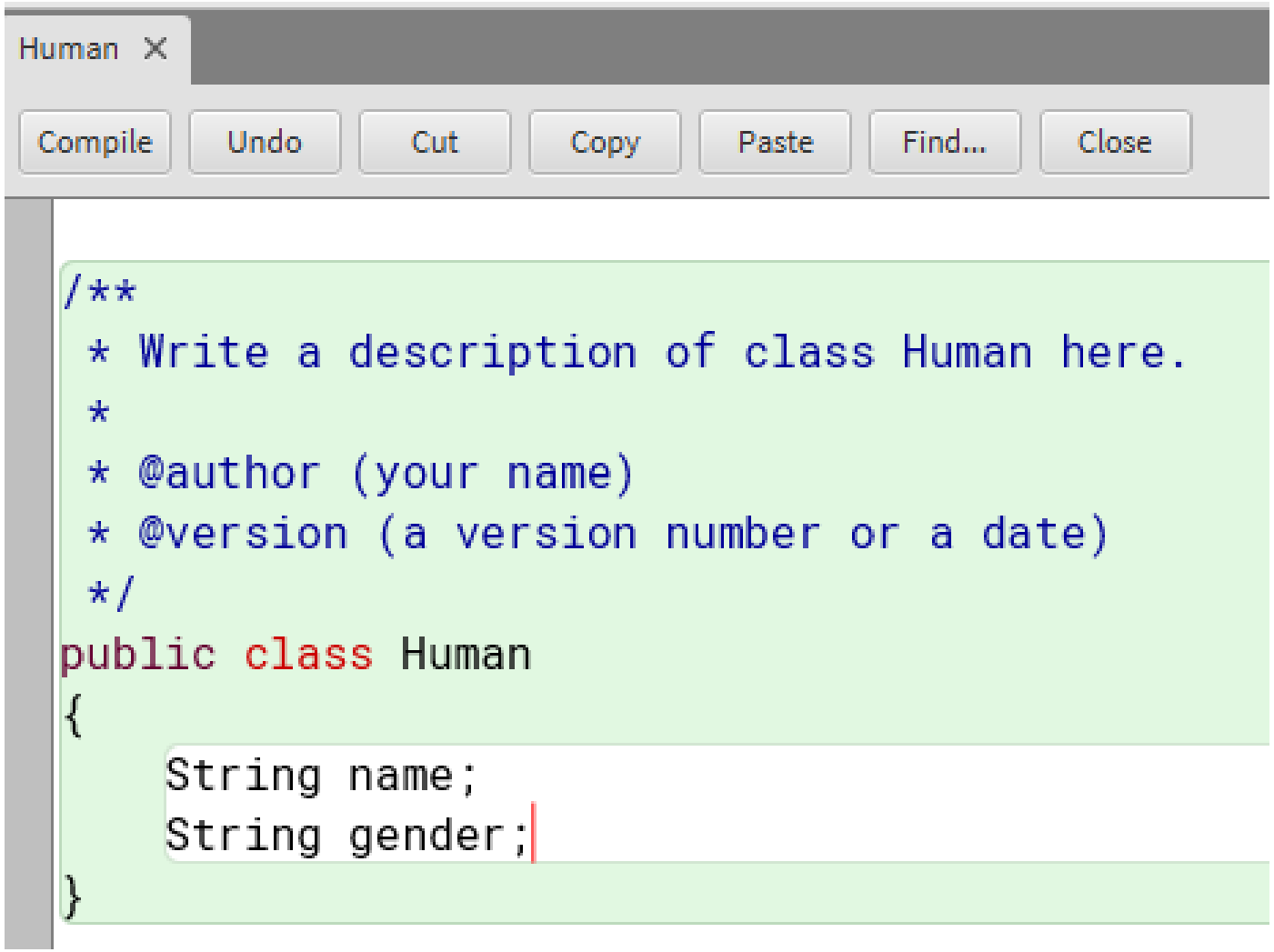
```
[modifiers] class <class-name> {
 // A field declaration
 [modifiers] <data-type> <field-name> [= <initial-value>];
}
```

- A field declaration can use zero or more modifiers. The data type of the field precedes its name.
- Optionally, you can also initialize each field with a value. If you do not want to initialize a field, its declaration should end with a semicolon after its name

Percobaan Class dan Object dengan BlueJ



Class Human


Human
Compile
Undo
Cut
Copy
Paste
Find...
Close
/**
* Write a description of class Human here.
*
* @author (your name)
* @version (a version number or a date)
*/
public class Human
{
String name;
String gender;
}

Class Human

- The Human class declares two fields: name and gender. Both fields are of the String type. Every instance (or object) of the Human class will have a copy of these two fields
- Java lets you declare two types of fields for a class:
  - Class fields
  - Instance fields
- Class fields are also known as class variables. Instance fields are also known as instance variables. In the previous snippet of code, name and gender are two instance variables of the Human class. Java has a different way to declare class variables. All class variables must be declared using the static keyword as a modifier.

Human

Compile | Undo | Cut | Copy | Paste | Find... | Close | Source Code

```java
/**
 * Write a description of class Human here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Human
{
    String name;
    String gender;
    static long count; // A class variable because of the static modifier
}
```

A class variable is also known as a static variable. An instance variable is also known as a non-static variable.

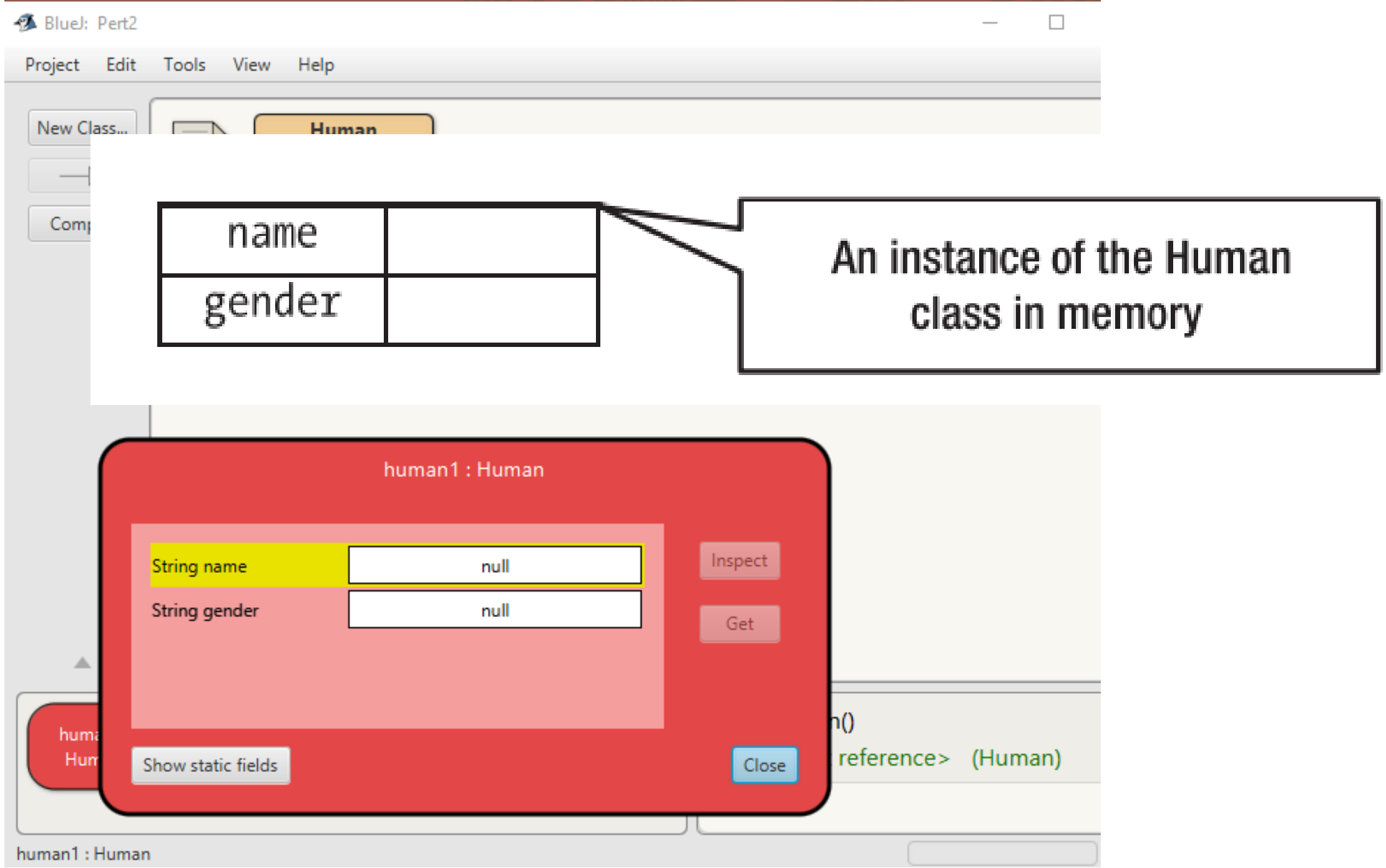
Creating Instances of a Class

- The following is the general syntax to create an instance of a class:

new <Call-to-Class-Constructor>;

- The new operator is followed by a call to the constructor of the class whose instance is being created.
- The new operator creates an instance of a class by allocating the memory on heap. The following statement creates an instance of the Human class:

```
new Human();
```


BlueJ: Pert2
Project
Edit
Tools
View
Help
New Class...
Human
name
gender
An instance of the Human class in memory
human1 : Human
String name
null
String gender
null
Inspect
Get
Show static fields
Close
reference>
(Human)
human1 : Human

Creating Instances of a Class(2)

- Here, Human() is a call to the constructor of the Human class. Did you add any constructor to your Human class? No. You have not added any constructor to your Human class. You have added only three fields to it.
- How can you use a constructor for a class that you have not added? When you do not add a constructor to
- a class, the Java compiler adds one for you. The constructor that is added by the Java compiler is called a
- default constructor. The default constructor accepts no arguments. The name of the constructor of a class is
- the same as the class name

Reference Type

- The name of a class defines a new reference type in Java. A variable of a specific reference type can store the reference of an instance of the same reference type in memory.
- Suppose you want to declare a reference variable, which will store a reference of an instance of the Human class. You will declare the variable as shown:

Human jack;

- Here, Human is the class name, which is also a reference type, and jack is a variable of that type. In other words, jack is a reference variable of Human type. The jack variable can be used to store a reference of an instance of the Human class.
- The new operator allocates the memory for a new instance of a class and returns the reference (or the indirect pointer) to that instance. You need to store the reference returned by the new operator in a reference variable:

jack = new Human();