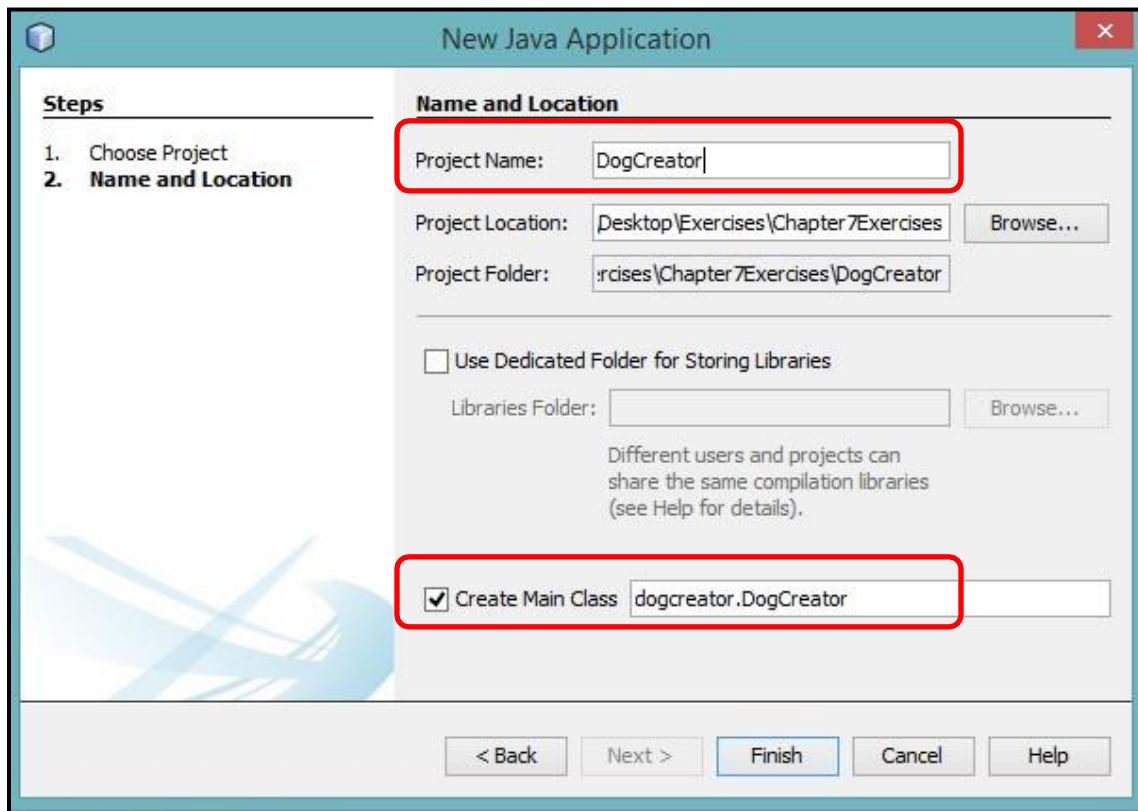


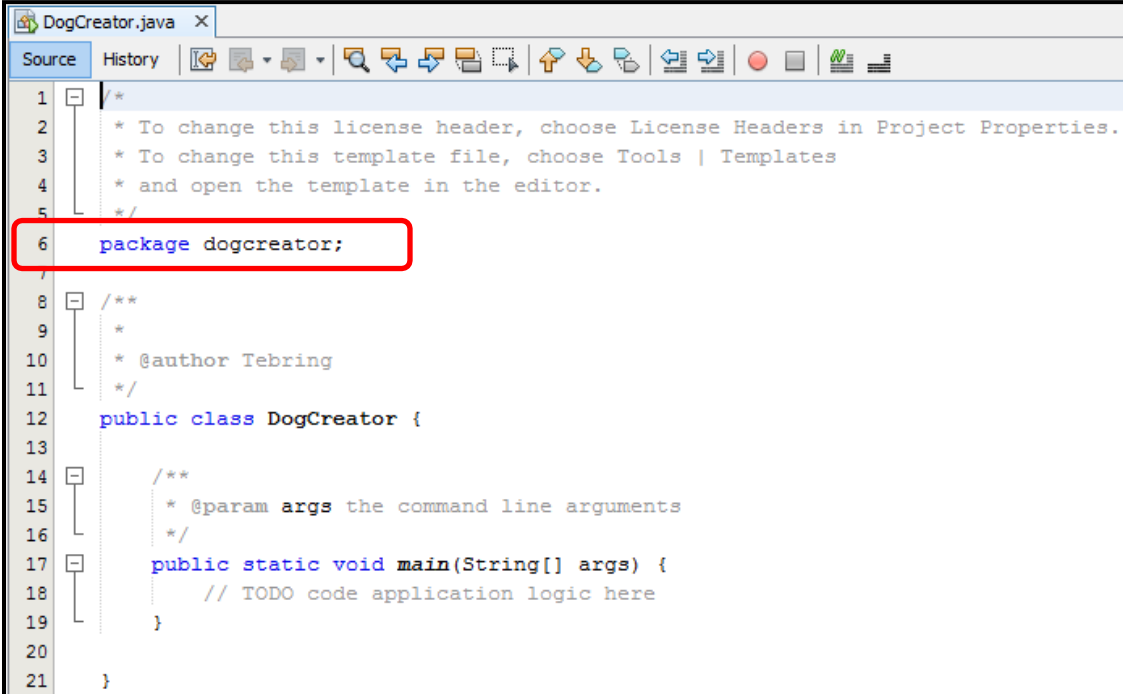
Setters and Getters Exercise

The following program is a user-defined Dog class that uses setter methods to set the name and weight instance variables, getter methods to get the name and weight instance variables, and a method to compare the weight of 2 animals.

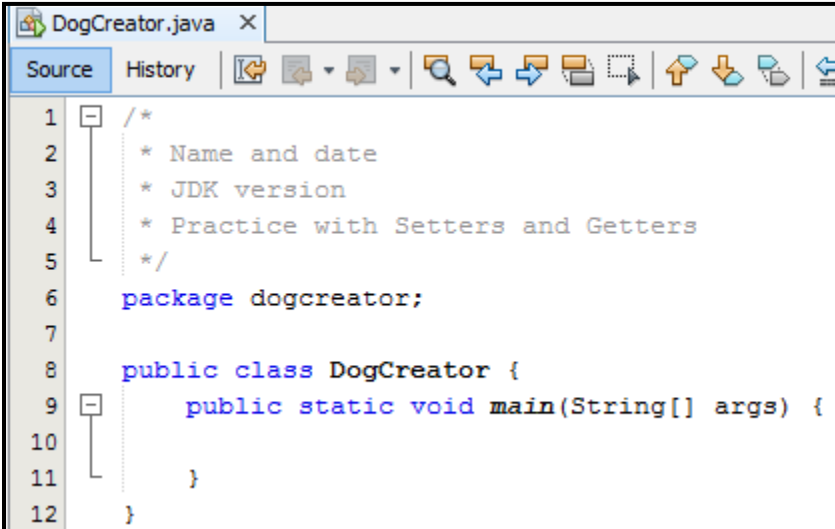
1. Get into NetBeans and start a new Java **application** program. The name of the project should be **DogCreator**. Take a look at the main class and notice that NetBeans automatically names it **dogcreator.DogCreator**. The name before the period is the package name which in this case is dogcreator and the name after the period is the main class name (filename) which in this case is DogCreator. Leave the generated main class name and click Finish.



2. There is a new line of code that is added on line 6. This line of code tells Java that the DogCreator file belongs to the dogcreator package (folder). Up until this point, we have been erasing the package statement by deleting the name before the period for our main class when we create our projects. When we erase the package name, NetBeans places our files into a default package. This made the code less confusing because default packages do not have the package line of code. If the files are placed into a named package, then you will need to have the package statement at the top of your code. We cannot erase this package line or the code will not compile. Go ahead and erase the comments throughout, add your own comments, but be careful not to erase the package statement. It is up to you from this point on if you want to create package names for your projects.

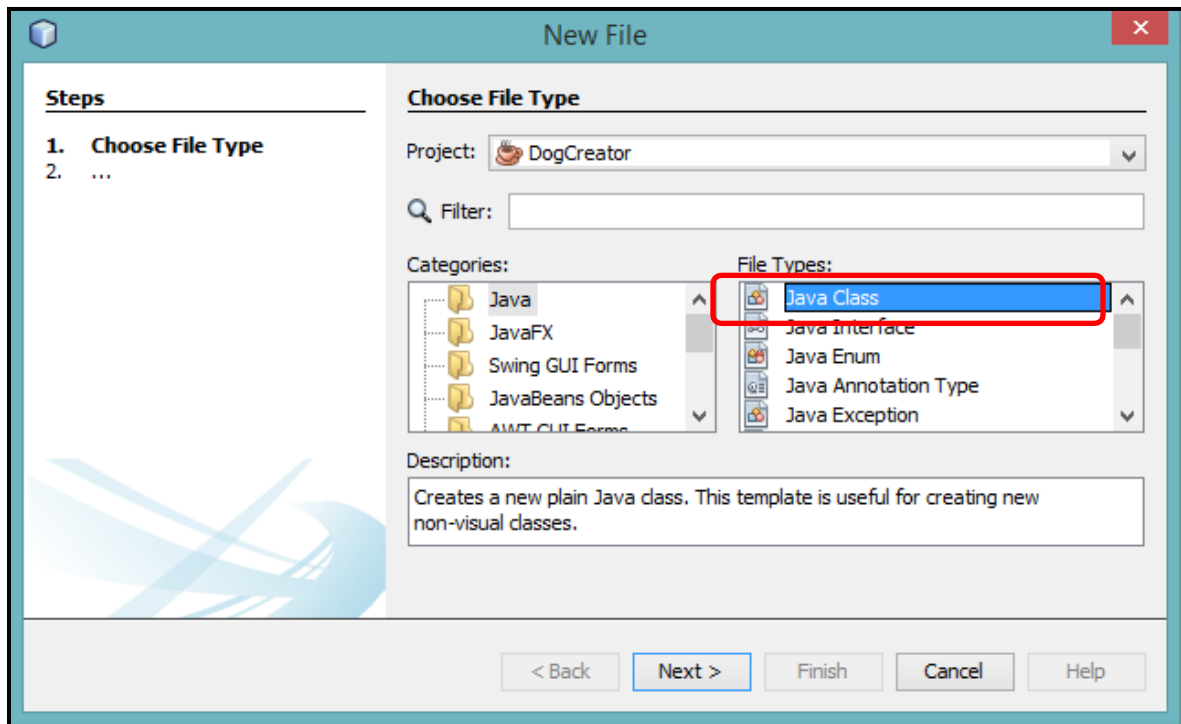


```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package dogcreator;
7
8  /**
9   *
10  * @author Tebring
11  */
12  public class DogCreator {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          // TODO code application logic here
19      }
20
21  }
```

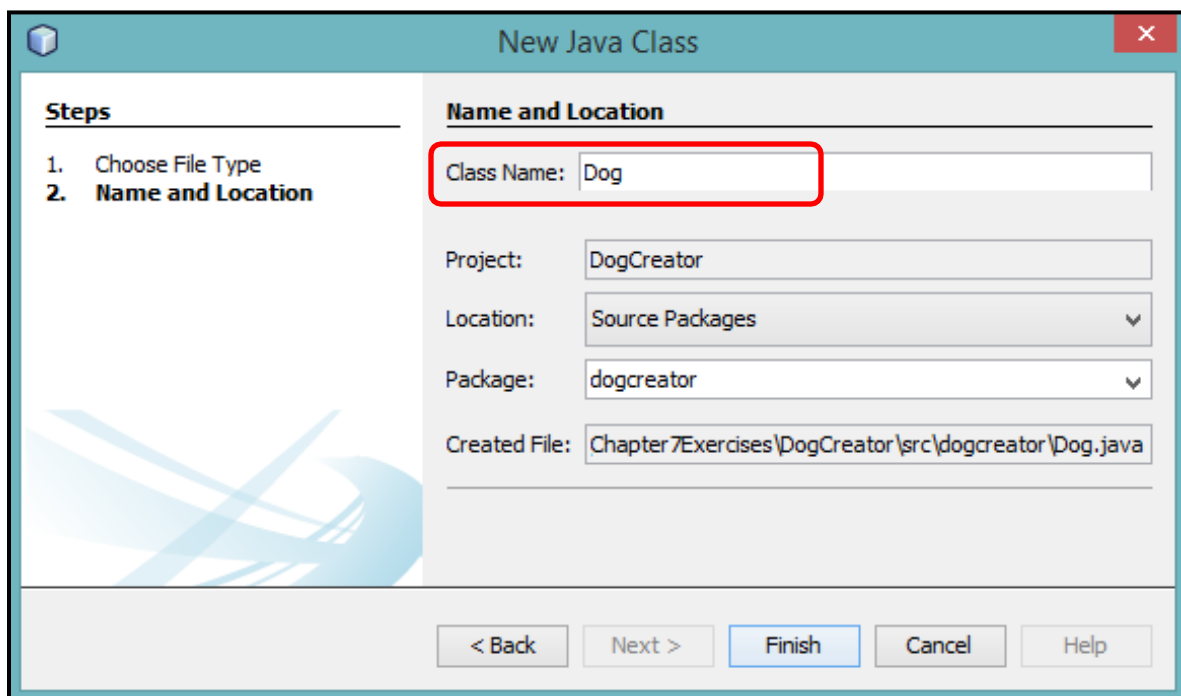


```
1  /*
2  * Name and date
3  * JDK version
4  * Practice with Setters and Getters
5  */
6  package dogcreator;
7
8  public class DogCreator {
9      public static void main(String[] args) {
10
11      }
12  }
```

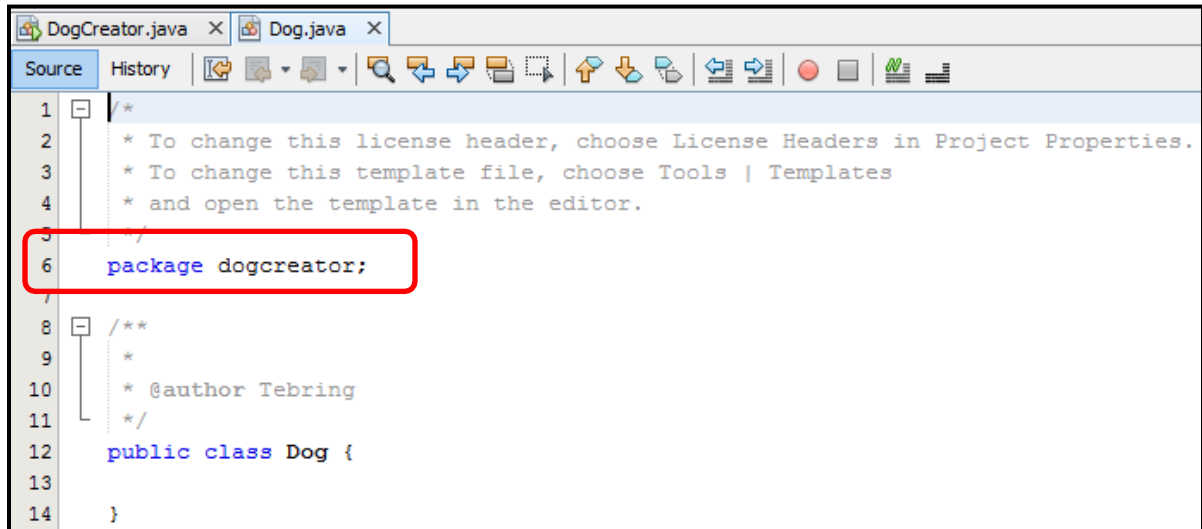
- Click on **New File...** from the **File** drop down menu in NetBeans. Click **Next**.



- Name the class **Dog** and click **Finish**.

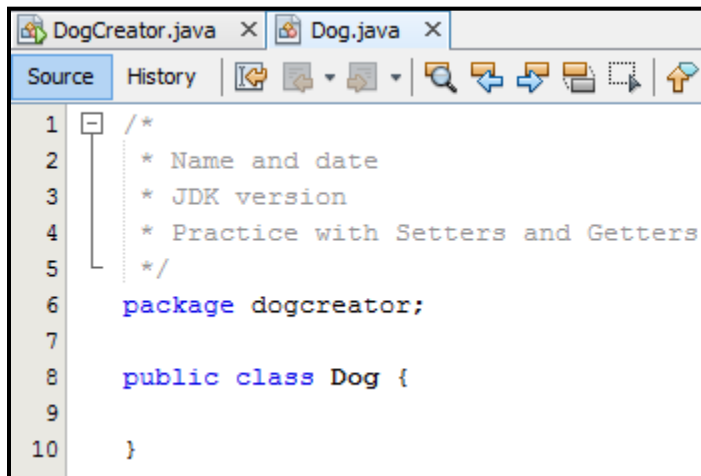


5. You should notice that this Dog class file has the dogcreator package statement same as the DogCreator class file. This statement tells Java that the Dog class belongs to the dogcreator package (folder). You must leave the line of code in the file.



```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package dogcreator;
7
8  /**
9   *
10   * @author Tebring
11   */
12  public class Dog {
13
14  }
```

6. Please adjust the comments to be your own.



```
1  /*
2   * Name and date
3   * JDK version
4   * Practice with Setters and Getters
5   */
6  package dogcreator;
7
8  public class Dog {
9
10 }
```

7. In the **Dog** class file, we are going to add instance variables (properties) for the dog (object). Remember that instance is just another word for object. For this example, we will use **name** and **weight** as our properties. There are many more properties for the dog such as birthdate, type, etc., but we will stick with just 2 to make the program easier to understand. Since we do not have a name and weight before we create the dog object, we will initialize (set) the name to null and the weight to 0. Make these instance variables private. Making a variable private, means that it cannot be accessed directly from another class. This will force the use of setters and getters to access the variables.

```
/*
 * Name and date
 * JDK version
 * Practice with Setters and Getters
 */
package dogcreator;

public class Dog {
    //define instance variables
    private String name = null;
    private double weight = 0;
}
```

8. Next, we are going to create a setter for the name of the dog called **setName**. Make sure that you are in the Dog class. This method is going to be *public* so that the DogCreator class can access this method. We do not add the word *static* to our method since this method is going to be used on an object. We are going to make this method *void* because it is not going to return any data. The method should have a parameter to take in the name; we will name this variable **newName** and it will be a string since the name will be text. The purpose of this method is to set the name of the dog (this.name) to the name that is passed into the method (newName). The word **this** refers to the current object that the method was called on and is optional. Type in the following code for the setName method.

```
/*
 * Name and date
 * JDK version
 * Practice with Setters and Getters
 */
package dogcreator;

public class Dog {
    //define instance variables
    private String name = null;
    private double weight = 0;

    //Setter method to set the animal name
    public void setName(String newName) {
        this.name = newName;
    }
}
```

9. Now we are going to create a setter for the weight of the dog called **setWeight**. You should be in the Dog class. This method is going to be *public* so that the DogCreator class can access this method. We do not add the word *static* to our method since this method is going to be used on an object. We are going to make this method *void* because it is not going to return any data. The method should have a parameter to take in the weight; we will name this variable **newWeight** and it will be a double since the weight can be a decimal amount. The purpose of this method is to set the weight of the dog (*this.weight*) to the weight that is passed into the method (*newWeight*), but only if the weight that is passed in is greater than 0. The word **this** refers to the current object that the method was called on and is optional. Type in the following code for the setName method:

```
/*
 * Name and date
 * JDK version
 * Practice with Setters and Getters
 */
package dogcreator;

public class Dog {
    //define instance variables
    private String name = null;
    private double weight = 0;

    //Setter method to set the animal name
    public void setName(String newName) {
        this.name = newName;
    }

    //Setter method to check validity of data and set dog weight
    public void setWeight(double newWeight) {
        if (newWeight > 0) {
            this.weight = newWeight;
        } else {
            System.out.println("Weight cannot be negative or zero.");
        }
    }
}
```

10. Let's add a dog constructor to the Dog class. Constructors are methods that have the same name and capitalization as the class. When an instance (object) of a class is created, the constructor is automatically called. The name of this constructor should be **Dog** and it should have 2 parameters to take in the name and weight (newName and newWeight) when the object is created. This constructor will call the setName and setWeight methods to set the name and weight of the dog object. Add the following constructor to the program.

```
1  /*
2   * Name and date
3   * JDK version
4   * Practice with Setters and Getters
5   */
6  package dogcreator;
7
8  public class Dog {
9      //define instance variables
10     private String name = null;
11     private double weight = 0;
12
13     //Dog Constructor
14     public Dog(String newName, double newWeight) {
15         setName(newName);
16         setWeight(newWeight);
17     }
18
19     //Setter method to set the animal name
20     public void setName(String newName) {
21         this.name = newName;
22     }
23
24     //Setter method to check validity of data and set dog weight
25     public void setWeight(double newWeight) {
26         if (newWeight > 0) {
27             this.weight = newWeight;
28         } else {
29             System.out.println("Weight cannot be negative or zero.");
30         }
31     }
32 }
```

Note: you will notice that there are warnings (yellow light bulb) on the setName and setWeight method calls (line 15 and 16). We will fix this in the next step.

11. You may notice warnings on the setName and setWeight method calls in the constructor. The warning may be worded as follows: *Overridable method call in constructor*. These are just warnings, not errors and so you can ignore them, but we will fix the code so that the setters are not overridable. If you had a class that extended the Dog class (known as a subclass), it could have a setName and setWeight method. Subclass methods override the superclass (Dog) and this could create an issue for our constructor in the Dog class which uses these methods to set the instance variables. Right now, we do not have a subclass that is going to have setName and setWeight methods and so it will not make a difference with our functionality. It is not imperative that we heed to these warnings, but let's change it so that you know how. Since it is not good practice to use methods in a constructor that can be overridden, let's change our setter methods (setName and setWeight) to be **final** as shown below. Setting a method to final tells Java that it cannot be overridden by a subclass.

```
1  /*
2   * Name and date
3   * JDK version
4   * Practice with Setters and Getters
5   */
6  package dogcreator;
7
8  public class Dog {
9      //define instance variables
10     private String name = null;
11     private double weight = 0;
12
13     //Dog Constructor
14     public Dog(String newName, double newWeight) {
15         setName(newName);
16         setWeight(newWeight);
17     }
18
19     //Setter method to set the animal name
20     public final void setName(String newName) {
21         this.name = newName;
22     }
23
24     //Setter method to check validity of data and set dog weight
25     public final void setWeight(double newWeight) {
26         if (newWeight > 0) {
27             this.weight = newWeight;
28         } else {
29             System.out.println("Weight cannot be negative or zero.");
30         }
31     }
32 }
```


12. Now that we have the basic code for creating a dog object with a name and weight, let's go ahead and create some dogs. We are going to create 3 dogs: Texie, Juicy, and Penny. Each of these dogs will have a different starting weight. The name and weight of the dog will get passed in as arguments to the Dog constructor in the Dog class. Click on the DogCreator file and add the following code.

```
/*
 * Name and date
 * JDK version
 * Practice with Setters and Getters
 */
package dogcreator;

public class DogCreator {
    public static void main(String[] args) {
        //Create dog objects
        Dog dog1 = new Dog("Texie", 25);
        Dog dog2 = new Dog("Juicy", 15);
        Dog dog3 = new Dog("Penny", 28);
    }
}
```

13. If you run this program, you will notice that nothing happens. It creates the 3 dogs, but you won't see this happening. If you want to find out the dog's name and weight, you would need to access the name and weight variables. If the name and weight variables weren't private, then you could access them directly (example: dog1.name or dog1.weight). Because our instance variables are private (defined in the Dog class), we will need to access them through getters. To create or getters, let's go back to the Dog class. Our getters are going to be *public* so that we can access them from the DogCreator file, they will not have the word *static* since they will be used on a dog object. The getName method will return the name of the dog (this.name) which is a string and the getWeight method will return the weight of the dog (this.weight) which is a double. The word **this** refers to the current object that the method was called on and is optional. Add the following getters under the setters in the Dog class.

```
//Getter method to get the animal name
public String getName() {
    return this.name;
}

//Getter method to get dog weight
public double getWeight() {
    return this.weight;
}
```

Note: you could make these methods final if you didn't want them to be overridden by a subclass.

14. Now that we have the getters in place for the Dog class, let's test them out by going to the DogCreator class and adding the following code:

```
/*
 * Name and date
 * JDK version
 * Practice with Setters and Getters
 */
package dogcreator;

public class DogCreator {
    public static void main(String[] args) {
        //Create dog objects
        Dog dog1 = new Dog("Texie", 25);
        Dog dog2 = new Dog("Juicy", 15);
        Dog dog3 = new Dog("Penny", 28);

        //Print out dog name and weight
        System.out.println("Name: " + dog1.getName());
        System.out.println("Weight: " + dog1.getWeight()+"lbs.");
        System.out.println("Name: " + dog2.getName());
        System.out.println("Weight: " + dog2.getWeight()+"lbs.");
        System.out.println("Name: " + dog3.getName());
        System.out.println("Weight: " + dog3.getWeight()+"lbs.");
        System.out.print("\n");
    }
}
```

15. Run the program. You should get the following output:

```
Name: Texie
Weight: 25.0lbs.
Name: Juicy
Weight: 15.0lbs.
Name: Penny
Weight: 28.0lbs.
```

16. Let's make it more interesting by comparing the dog weights. In the Dog class, we will add a **compare** method. This method will be *public* so that it can be accessed by the DogCreator file, it will return a string stating which dog weighs more, and it will take in another dog object as a parameter to do the comparison. We will create a variable named **message** and initialize it to *null*. This variable will be used to hold the message that we want to return. We will need to use conditionals to do the comparison to determine which dog is heavier between 2 dogs. When the method is called from the DogCreator method it will be called on a dog object (dog object will be referred to as *this*) and it will pass in another dog object to compare it to (dog object will be referred to as *dogCompare*).

```
//method to compare weight of 2 dogs
public String compare(Dog dogCompare) {
    String message = null;
    if(dogCompare.weight > this.weight) {
        message = dogCompare.name + " weighs more than " + this.name;
    } else if(dogCompare.weight < this.weight){
        message = this.name + " weighs more than " + dogCompare.name;
    } else {
        message = this.name + " weighs equal to " + dogCompare.name;
    }
    return message;
}
```

17. The finished Dog class should look as follows:

```
1  /*
2   * Name and date
3   * JDK version
4   * Practice with Setters and Getters
5   */
6  package dogcreator;
7
8  public class Dog {
9      //define instance variables
10     private String name = null;
11     private double weight = 0;
12
13     //Dog Constructor
14     public Dog(String newName, double newWeight) {
15         setName(newName);
16         setWeight(newWeight);
17     }
18
19     //Setter method to set the animal name
20     public final void setName(String newName) {
21         this.name = newName;
22     }
23
24     //Setter method to check validity of data and set dog weight
25     public final void setWeight(double newWeight) {
26         if (newWeight > 0) {
27             this.weight = newWeight;
28         } else {
29             System.out.println("Weight cannot be negative or zero.");
30         }
31     }
32
33     //Getter method to get the animal name
34     public String getName() {
35         return this.name;
36     }
37
38     //Getter method to get dog weight
39     public double getWeight() {
40         return this.weight;
41     }
42
43     //method to compare weight of 2 dogs
44     public String compare(Dog dogCompare) {
45         String message = null;
46         if (dogCompare.weight > this.weight) {
47             message = dogCompare.name + " weighs more than " + this.name;
48         } else if (dogCompare.weight < this.weight) {
49             message = this.name + " weighs more than " + dogCompare.name;
50         } else {
51             message = this.name + " weighs equal to " + dogCompare.name;
52         }
53         return message;
54     }
55 }
```

Chapter 9 – Classes and Objects

397 | Page

18. Let's test the compare method that we just created by going back to the DogCreator class and calling the compare method using our 3 dogs as shown below:

```
public static void main(String[] args) {  
    //Create dog objects  
    Dog dog1 = new Dog("Texie", 25);  
    Dog dog2 = new Dog("Juicy", 15);  
    Dog dog3 = new Dog("Penny", 28);  
  
    //Print out dog name and weight  
    System.out.println("Name: " + dog1.getName());  
    System.out.println("Weight: " + dog1.getWeight()+"lbs.");  
    System.out.println("Name: " + dog2.getName());  
    System.out.println("Weight: " + dog2.getWeight()+"lbs.");  
    System.out.println("Name: " + dog3.getName());  
    System.out.println("Weight: " + dog3.getWeight()+"lbs.");  
    System.out.print("\n");  
  
    //Comparisons  
    System.out.println(dog1.compare(dog2));  
    System.out.println(dog1.compare(dog3));  
    System.out.println(dog2.compare(dog3));  
    System.out.print("\n");  
}
```

19. You should now have the following output:

```
Name: Texie  
Weight: 25.0lbs.  
Name: Juicy  
Weight: 15.0lbs.  
Name: Penny  
Weight: 28.0lbs.  
  
Texie weighs more than Juicy  
Penny weighs more than Texie  
Penny weighs more than Juicy
```

20. Now let's say that Texie gains weight. We can adjust her weight using the setWeight method and then we can do the comparisons again using the compare method to see if anything has changed. Add the following code to the DogCreator class:

```

public static void main(String[] args) {
    //Create dog objects
    Dog dog1 = new Dog("Texie", 25);
    Dog dog2 = new Dog("Juicy", 15);
    Dog dog3 = new Dog("Penny", 28);

    //Print out dog name and weight
    System.out.println("Name: " + dog1.getName());
    System.out.println("Weight: " + dog1.getWeight()+"lbs.");
    System.out.println("Name: " + dog2.getName());
    System.out.println("Weight: " + dog2.getWeight()+"lbs.");
    System.out.println("Name: " + dog3.getName());
    System.out.println("Weight: " + dog3.getWeight()+"lbs.");
    System.out.print("\n");

    //Comparisons
    System.out.println(dog1.compare(dog2));
    System.out.println(dog1.compare(dog3));
    System.out.println(dog2.compare(dog3));
    System.out.print("\n");

    //dog1 gains weight
    dog1.setWeight(28);
    System.out.println("Name: " + dog1.getName());
    System.out.println("Weight: " + dog1.getWeight()+"lbs.");
    System.out.println(dog1.compare(dog3));
}

```

21. Run the program again to see that Texie gained weight and now her weight is equal to Penny's weight.

```

Name: Texie
Weight: 25.0lbs.
Name: Juicy
Weight: 15.0lbs.
Name: Penny
Weight: 28.0lbs.

Texie weighs more than Juicy
Penny weighs more than Texie
Penny weighs more than Juicy

Name: Texie
Weight: 28.0lbs.
Texie weighs equal to Penny

```