Java Objects Part 1 Notes

Team Treehouse

May 22, 2020

1 Welcome Back

- *STRING.toLowerCase()*
  - Turns string into lowercase letter
- *STRING.contains(...)*
  - checks if value ... is contained inside *String*

2 Quiz 1

1. Please fill in the correct answer in each blank provided below.

```
String someWords = "These are words";
someWords.______("words");

```

Answer:

```
String someWords = "These are words";
someWords.contains("words");

```

2. The boolean datatype is used to store:

   A. numbers
   B. text

C. true or false values

Answer: C

3. Please fill in the correct answer in each blank provided below.

What operator do we use to ensure that both of these conditions are met:

```
    boolean isRefreshed = true;
    boolean isReadyToGetStarted = true;
    boolean shouldContinue = isRefreshed______isReadyToGetStarted;

```

Answer:

```
    boolean isRefreshed = true;
    boolean isReadyToGetStarted = true;
    boolean shouldContinue = isRefreshed && isReadyToGetStarted;

```

4. Please fill in the correct answer in each blank provided below.

```
    int weightOfCraigsKid = 50;
    int weightMonty = 130;
    if (weightMonty ______ weightOfCraigsKid) {
        console.printf("Whoa that's a huge dog!");
    }

```

Answer:

```
    int weightOfCraigsKid = 50;
    int weightMonty = 130;
    if (weightMonty > weightOfCraigsKid) {
        console.printf("Whoa that's a huge dog!");
    }

```

5. To define a new variable to store a name it would look something like this:

A. String firstName = "Bob";

B. "Bob" = first.name

C. firstName = "Bob";

D. first_name = 'Bob'

Answer: A

6. The boolean datatype is used to store:

   A. numbers
   B. text
   C. true or false values

   Answer: C

3 Creating Classes

```
class PezDispenser {  // <- 1. Class is created in a separate and matching file :)
    String characterName = "Yoda";
}

```

Listing 1: lesson_3/PezDispenser.java

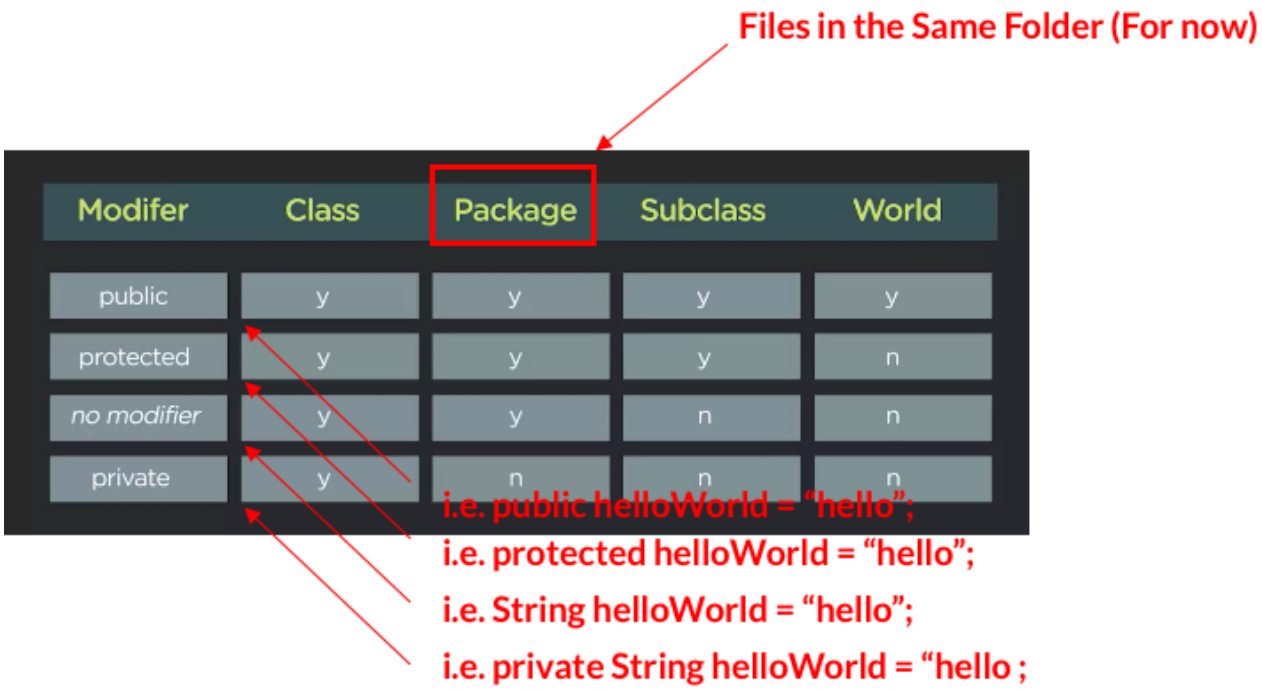
```
import java.io.Console;

public class Example {
    public static void main(String[] args) {

        System.out.println("We are making a new PEZ dispenser");

        PezDispenser dispenser = new PezDispenser(); // <- 2. And is used here :)

        System.out.printf("The dispenser is %s", dispenser.characterName);
        ...
    }
}

```

Listing 2: lesson_3/Example.java

4 Exercise 1

- Solution included in *exercise_1.java*

5 Access Modifiers



- Determines who is intended to access the information
- Adding access modifiers to attributes and methods in class is called **Encapsulation**

```
1    class PezDispenser {
2        private String characterName = "Yoda"; // <- 1. attribute is
     turned private
3    }
4
```

Listing 3: lesson_5/PezDispenser.java

```
1    import java.io.Console;
2
3    public class Example {
4        public static void main(String[] args) {
5
6            System.out.println("We are making a new PEZ dispenser");
7
8            PezDispenser dispenser = new PezDispenser();
9
10           System.out.printf("The dispenser is %s", dispenser.
     characterName); // <- 2. and it can't be accessed outside of class
11       }
12   }
13
```

Listing 4: lesson_5/Example.java

Notes:

- Files can be compiled and displayed by typing *javac Example.java && java Example* in terminal

6 Exercise 2

- Solution included in *exercise_2.java*

7 Methods

- is a collection of statements that are grouped together to perform an operation
- is like *verb*
- *ACCESS_MODIFIER DATA_TYPE get<ATTRIBUTE_NAME >* is called **getter**

```
1   public class PezDispenser {
2       private String characterName = "Yoda";
3
4       public String getCharacterName() { // <- 1. Getter method added
    here
5           return characterName;
6       }
7   }
8
```

Listing 5: lesson_7/PezDispenser.java

```
1   import java.io.Console;
2
3   public class Example {
4       public static void main(String[] args) {
5
6           System.out.println("We are making a new PEZ dispenser");
7
8           PezDispenser dispenser = new PezDispenser();
9
10          System.out.printf("The dispenser is %s", dispenser.
    getCharacterName()); // <- 2. And is used here
11      }
12  }
13
```

Listing 6: lesson_7/Example.java

8 Exercise 3

- Solution included in *exercise_3.java*

9 Constructors

- is a method that will run when class is instantiated.
- is created by writing a method with the same name as class
- *this* is like *self* in python

```
1   public class PezDispenser {
2       private String characterName = "Yoda";
3
4       public PezDispenser(String characterName) {
5           this.characterName = characterName;
6       }
7
8       public String getCharacterName() {
9           return characterName;
10      }
11  }
12
```

Listing 7: lesson_9/PezDispenser.java

```
1   import java.io.Console;
2
3   public class Example {
4       public static void main(String[] args) {
5
6           System.out.println("We are making a new PEZ dispenser");
7
8           PezDispenser dispenser = new PezDispenser("Yoda");
9
10          System.out.printf("The dispenser is %s", dispenser.
    getCharacterName()); // <- 2. And is used here
11      }
12  }
13
```

Listing 8: lesson_9/Example.java

10 Exercise 4

- Solution included in *exercise_4.java*