Java Class

Classes and Objects

11th June 2020

OVERVIEW

We covered methods in the last unit. They were very useful for keeping things organized, but they also have another major use: as ways to create code that can run on a custom variable type! Objects are the foundation of Java (hence the name Object-Oriented Programming Language). Everything you use, from the random generator to the scanner to even strings are objects with their own set of methods behind them. Each object is created from a class. Think of the class as a blueprint for the object to be built off of. In this unit, you will be learning about and building a class from scratch.

GOALS

1. Understand what a class and object is and how they relate
    a. **VOCAB**
        i. Method
        ii. Class
        iii. Object
        iv. Return Type
        v. Constructor
        vi. Instance Variables
2. Write your own Class

INSTANCE VARIABLES

- The variables that are intrinsic to the class. Think of these as things the object needs to know before doing its job
    - Example: Before a restaurant can make an order, it needs to know how much food it is preparing

```
Order myorder = new Order(3,5,1)
```

- Where 3 is burgers, 5 is fries, and 1 is drinks

- The special syntax for these variables

```
int myint;
double mydouble;
String mystring;
```

- Notice how we aren't defining them, just reserving spots in memory. The defining comes next

THE CONSTRUCTOR

- A special method that is used to set the values of instance variables inside of a class. Essential for any class.
- Special syntax: `public <NameOfClass>(variables go here){}`
    - Make note that there is no return type here

GETTERS

- Now that we defined our variables, we need a way to obtain them and modify them. Enter getters and setters
- We can't just say car.miles = 10, that's illegal (because of course it is). So we need to create special methods to access these variables.
- What two things can we do to variables? Return them (get) and modify them (set)
- Basic get method

```
public <typeofdatatoreturn> get[nameofvariable](){
Return <nameofvariable>;
}
```

- If the variable was an int named mynumber then the method would be

```
public int getMyNumber(){
Return MyNumber;
}
```

SETTERS

- Now we must create a method to change the value of the variable.
- This is done by inverting the setter method, like so

```
public void set[nameofvariable](<datatypeofvariable> i){
<nameofvariable> = i;
}
```

- So if the name of the variable was mynumber, it would look like this

```
public void setMyNumber(int i){
myNumber = i;
}
```

PROJECT

You're on the clock. Actually, you're making it.

We just explained how classes and objects worked, and you saw a guide for a car. Now what if we were to make a new object, a clock! Our first step is to think of what a clock needs to know before it can tell time, it needs to know the hour, minutes, and seconds before it can tell us what time it is. Your project must have:

1. Three instance variables, all integers, named sec, min, hour.
2. A constructor to set these
3. Getters and setters for all three (nine total)
4. An extra method that gives the sum of the three variables, named varSum that returns an integer.