Object Oriented Programming

Workshop

William L. Honig, Ph.D.

Create a full OOP Class

Document in professionally (Javadoc)

Use and test your Class

Refine your Class

Step 0

Pick your Topic: Pick something in the real world you like / enjoy / appreciate. It needs to be something that has several instances. You need a singular noun to describe it. Example: Person

What determines the State (internal data) for objects of this type? You need to have at least three pieces of information. They can be simple or complex. Example: name, birthdate, location.

What operations / manipulations / access / transformations would a program need to do on these kinds of things? Think beyond simple set and get of the state data. Example: marry another person, be born (create a new object), have a child, enroll in university.

Done? Create a text file with all this information documented.

Step 1

Create a class: write some oop code. It should have (in order)

1. Some constructors (including a default constructor)
2. Suitable data attributes, each with a useful type
3. Several methods or operations. Start with just the signatures.

Done? You are done (for now) if you have something that compiles with proper OOP structure

Step 2

Add documentation: adapt your text output from step 0 into Javadoc and pseudo code as

comments in the class. Details will be provided.

Done? You are done if *someone else* can read and understand your class

Step 3

Use and test your class: create a separate text class, containing main. Use it to

1. Create 3 or more objects of your class
2. Prompt the user (or use code) to get values and/or instructions for what to do with your objects.
3. Print out information about your objects

Done? You are done if main tests all parts of your class (even if the results don't always make sense).

Step 4

Refine your class: improve the class structure using good OOP. You may need to consider any of these

1. Improve the methods so they are more realistic
2. Add additional information to the object state information (and modify the methods as necessary to support them)
3. Invest other related classes that your original class can use. Example: a class for Marriage, a class for BirthDate, a class for Spouse (hum…not sure I'd do that way).

Done? Probably never, or at least until you have really implemented the full class concept as originally conceived.

v1.0 (Original) 15 Oct 2017