[0580] EXTRA: Recipe for a POJO (Class)

Here is a standardized list of steps that you should take for almost every class you create. By following this "recipe" consistently every time you create a class, you will be sure that your classes work together effectively once you are done. You will also reduce the number of dumb mistakes and forgotten steps that usually plague student programs. Try to make these steps a habit. It will really help you in the long run!

Design Phase

- Do noun-verb analysis on your project to identify possible classes, properties, methods, etc. **[0555] Object Oriented Design via Noun-Verb Analysis (https://lwtech.instructure.com/courses/1841516/pages/0555-object-oriented-design-via-noun-verb-analysis)**
- Put on your "Class Designer" hat and spend some time thinking about the "customers" for your class.
  - As a "Class Designer," you try to forget about the Main program you are writing and think only about the class you are currently creating. Your goal is to create the best possible version of this class in the history of the universe! You want to make it as useful as possible to all programmers - not just you - that might try to use it in the future.

POJO Coding Phase

- Use a Singular noun as your class name
- Identify all of the attributes of your class and create private member variables for each attribute
- Create a constructor that takes all user-configurable attributes as its parameters.
- Add optional values to each of your constructor's parameters as appropriate.
- For each member variable in your class, if that attribute's value is needed by the customers of your class, create a public "Getter" for it.
- Add "Setters" as appropriate. Try to add as few setters as possible. Zero is a great number of setters to have.
- (Optional) Consider changing your getters and setters into .NET Properties
- Override the ToString() method and have it create a human-readable text version of your object and its properties.

General Purpose Class Coding Phase

- Revisit the verbs you identified in your class design. Create a method for each verb if your customers really need it.
- (Later, after we talk about exceptions) At the start of each method, have your code check every parameter in every method (and every constructor) for validity. Throw an appropriate exception if it is invalid.

Testing Phase

- Revisit each method of your class looking for incoming parameters that need to be checked for illegal values.
- (Later, after we talk about unit tests,) Create Unit Tests for each method in your class. Create tests for normal inputs, abnormal inputs, null inputs and edge cases.

Integration Phase

- Finally, take off your "Class Designer" hat and put your "Project Programmer" hat back on. Now, add your new class to your program and use it inside your Main() method to make sure it performs as expected.