Week 05 Research

Cody Schliebe

6/2/2023

OOP

1. The four pillars of Object Oriented Programming are: encapsulation, inheritance, polymorphism, and abstraction. Encapsulation is basically how you keep your application secure. You have different chunks, or objects, that hold and handle different bits of information, and the information is only available for use by the objects that need it. This is a good way to hide sensitive information from a user. Inheritance is the process of passing on functionality from a parent class to a child class. If code is already present in a parent class, it would be silly to re-write it for each child class, so this code can be inherited by the child and used or added to. Polymorphism is the fact that a single object can react differently, or behave differently, depending on how a function calls it. Abstraction, finally, is the practice of breaking an application into many abstract units. This both hides complexity from a user, but also makes it more digestible for a coder who has to debug, change, or further engineer the code or application. [1]
2. In the videos, the speaker talks about the difference between a class and an object being like the difference between a blueprint and a product. I prefer to think of it as a recipe vs. a cake. A class is like a recipe for red-velvet cake, which has its ingredients, its amounts, its steps, its order, and its baking time. How do you get the red color in the cake? When do you fold in the vinegar and baking soda? What frosting do you use? The object is the cake you make using the recipe, and the object is only as good as the class that describes it. If you have an overly simple class, you'll have a fairly one-dimensional object with few issues, but if you have a ridiculously complex class, you'll end up with a super-detailed object, but there are more chances your cake could get screwed up along the way. [2]

Exceptions

5. An exception is an event that happens during coding that is not what you intended. These can range from minor annoyances (forgetting to capitalize

"String") to errors that can cause you to sink hours into research. Today, for example, I spent two hours trying to figure out why, when I tried to print the contents of the hands that the card players were holding, it printed several instances of week05.Card@5e9f23b4 or similar. I knew what the gibberish was (memory locations, etc.), but it took lots of research on Stack Overflow and similar sites to pinpoint what was actually happening. The reason, according to the collective experience of the coding masses? I did something wrong. Exceptions are often meant to help you correct your own errors, as I'll discuss next, but sometimes the program happily thinks it's doing nothing weird and you end up wasting an afternoon only to begrudgingly go to the answers to see how the instructor did it. [3]

6. The two types of exceptions in Java are Checked Exceptions and Unchecked Exceptions. Checked exceptions, or IO or compile time exceptions, are thrown up at compiler time, but they are often highlighted before you ever try to run your application. The above example of not capitalizing String, for example, will get you a big red line and, even if you get your program to run, it won't behave as it should. Usually checked exceptions won't allow you to run the program. Unchecked exceptions, or runtime exceptions, are exceptions that happen while your application is running. If you try to reference an index in an array that's outside of the existing range, the compiler will think the code looks fine. When you run it, though, the alarm bells go off. If you were to write code that, for example, took the string "bigger and " and concatenated it to itself, then ran it through an infinite loop, the code looks fine but it could quickly fill up your computer's memory and bring your system to a screeching halt. I believe Java has checks for this, but I crashed a few computers when I used to play around with QBasic. [4]

Sources

1. https://www.freecodecamp.org/news/four-pillars-of-object-oriented-programming/
2. https://www.w3schools.blog/object-and-class-in-java
3. https://www.w3schools.blog/exception-handling-tutorial-java
4. https://www.geeksforgeeks.org/checked-vs-unchecked-exceptions-in-java/