Rex Jones II

(Transcript) Intro To Object-Oriented Programming

Rex Jones II

Intro To OOP

(Object-Oriented Programming)

Introduction

Hello and Welcome, in this series, let's talk about Object-Oriented Programming better known as OOP. In this Intro To OOP, we are going to discuss Structured Programming and Object-Oriented Programming.

OOP took the best ideas from Structured Programming and combined those ideas with a few new concepts. We have the option of organizing our program around code or we can organize our program around data. Structured Programming is organized around code while the OOP's concept is organized around data.

My name is Rex and I like to share programming and automation knowledge. If you are interested, connect with me on YouTube, Twitter, LinkedIn, Facebook, and GitHub. All of the documents including the transcript will be placed on GitHub. Videos are released 3 times a week on Monday, Wednesday, and Friday.

Slide 3

When it comes to Structured Programming, the programmer divides the whole program into smaller units. For example, let's take an employee working at a company. First, they submit their resume, Attend the interview or interviews, Get hired at the company, Perform their job responsibilities, then Receive a paycheck. The big program is divided into small units such as Calculate Salary, Calculate Bonus, and Calculate Health Benefits. That's what we call code acting on data.

Slide 4

Object-Oriented Programming is different. With this same example of an employee working at a company. Our program has a class called Employee. It's organized around data and the data controls access to the code. We define the data and we define the methods that are allowed to act on the data. The data is age, employeeID, and title while the methods are calculateSalary and calculateBonus.

Slide 5

Here's a side by side comparison of Structured and Object-Oriented Programming. There was a point when Structured Programming reached its limit and could not handle complex applications. Object-Oriented Programming was created to help developers deal with more and more and more and more challenges as the software application grew.

Slide 6

The OOP's concept deals with complex application using Classes and Objects. Our class is Employee and our objects are jane and john. Classes are the foundation which builds the entire Java programming language. Also, classes define the nature of an object. In other words, a class is a copy or template for an object. The same way class has age, our object jane has age and the value is 25. Just like the Employee class has employeeID and title. Jane has employeeID with a value of 34 and Automation Engineer as the title. The same goes for each method calculateSalary and calculateBonus. Jane has access to both methods calculateSalary and calculateBonus. We see how a class serves as a pattern for creating the object. In the same manner, our object john has access to the same data and methods.

Slide 7

All Object-Oriented Programming languages have 4 common traits. Those 4 traits are Encapsulation, Inheritance, Polymorphism, and Abstraction. In some cases, you might see some people say all OOP languages have 3 common traits: Encapsulation, Inheritance, and Polymorphism. Abstract Classes, Abstract Methods, and Interfaces falling under Inheritance. The concepts remain the same so it does not matter where we place them, as long as we understand the concepts.

- Encapsulation is a way to protect the data by keeping it safe from outside classes. It can only be accessed through a method within its own class
- Inheritance allows a hierarchy creation of classes. It's a component where 1 class is an extension of a different class. The class that is extended is called a superclass and the class that extends is called a subclass. However, I prefer to say parent class and child class. A child class receives the data and methods from the parent class. This incorporation happens in the declaration of a child class.
- Polymorphism means many forms. It allows an object to acquire many representations. All of those representations operate similar to each other. As a result, 1 interface has the ability to take on properties of more than 1 class.
- Abstraction is when the necessary details are made available. Abstract classes create a parent class that specifies only a general form shared by the child class. An abstract method contains no body and not implemented by the parent class.

That’s it for the Intro To OOP’s. Next, I will demo Encapsulation.