**7. What is polymorphism and give examples from your automation?**

- Polymorphism is one of the object-oriented concept where a single action can take multiple forms. In simple words, we can say it is the ability to exhibit more than one form.

- There are two types of polymorphism like Overloading (compile-time polymorphism) and Overriding (run-time polymorphism).

**Method Overloading:**

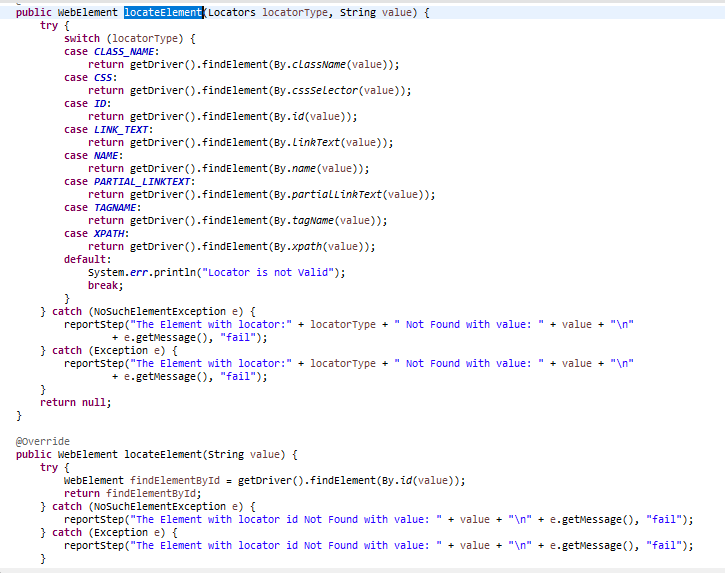
* Overloading means, a same class can have multiple methods with the same name but different parameter lists (different data types or a different number of parameters).
* The purpose of method overloading is to simplify verbose.

(When you have multiple methods with similar functionality but different parameter sets, it becomes easier for developers to identify which method to use based on the context)

- In my framework, we have many method overloading methods, some of them are locateElement(),startApp()

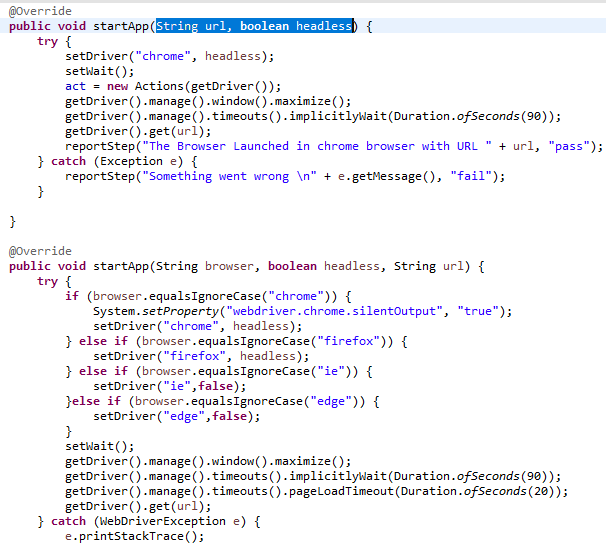
**Eg1: LocateElement()**

- In the SeleniumBase Class, there are two ways to find an element. The first method is called `locateElement(String value)`, where we should only need to provide a string value, usually the value of id locator. The second method is `locateElement(Locator locatorType, String value)`, where we can give two things: the type of locator to use (like ID or something else) and the value of that locator to find the element.



**Example 2:**

- In SeleniumBase, there are two ways to begin using an application based on its web address (URL). The first method is called `startApp(String url, boolean headless)`. With this method, you provide the URL as a string and a boolean value to indicate whether to start the application in a hidden (headless) mode. By default, it uses the Chrome browser. The second method is `startApp(String browser, boolean headless, String url)`. Here, you give three things: the specific browser you want to use (like Chrome or Firefox), whether you want it to be in headless mode,and the URL of the application you want to load.



**Method Overriding:**

Method overriding is a concept in object-oriented programming (OOP) where a subclass (child class) provides a specific implementation for a method that is already defined in its superclass (parent class). The subclass modifies the behavior of the method to suit its own needs while keeping the method signature (name and parameters) the same as in the parent class.

For example:

In Java,

- toString() in Object class is overridden in String class.

- equals() in Object class is overridden in String class

In Selenium,

- quit() in RemoteWebdriver class is overridden in ChromiumDriver class

In our framework we haven’t applied overriding.