Selenium 4 Beginners

Rex Jones II

Understanding Java Variables & Operators

Table of Contents

Introduction

Hello Everybody, Welcome To Selenium 4 Beginners. My name is Rex Allen Jones II.

In this video's Tutorial Plan, I will cover Java Data Types, Java Variables, and Java Operators. Data Types serve as the foundation of data manipulation in Java. Variables are used for storing data values in a memory location. Operators are symbols that perform operations and return a result.

If you are interested, you can download this presentation and automation code at https://tinyurl.com/JavaVariablesOperators

Java Data Types

Now, let's dive into Java Data Types.

There are 2 types of Data Types for Java. We have Primitive Data Types and Class Data Types. Both Data Types are called many names but I prefer to call them Non-Object and Object. I will briefly cover the Non-Object Data Types in this video. However, the Non-Object and Object Data Types were covered in my other video (Part 1) Building Blocks For Selenium.

The Primitive Data Types have 4 categories: Integer, Floating Point, Character, and Boolean. Integer and Floating Point hold numeric data, Character hold text, while Boolean holds true or false values. Within these 4 categories, there's a total of 8 Data Types.

4 of the Data Types are located in the Integer category: int, byte, short, and long. 2 of the Data Types are located in the Floating Point category: double and float. Character has 1 Data Type which is char and Boolean also has 1 Data Type which is boolean. The boolean Data Type holds either a true or false value. Although there are 8 Primitive Data Types, int, double, and boolean are the most used Data Types.

The int Data Type supports numerical values without a decimal while double supports numerical values with a decimal. boolean supports conditional statements that return true or false and can also be assigned a true or false value.

Let's go over to Eclipse and get started with Data Types. I will focus on the 3 most used Data Types. However, the same concepts apply to all Data Types. Method name will be demoDataTypes. We start by writing the Data Type: int, double, and boolean. The Data Type makes sure we assign the correct type of data. We saw in our presentation slide that int supports numbers without decimals, double supports numbers with decimals, and boolean support true or false values

Java Variables

Next, we have Java Variables

There are 3 phases of variables. We must name our variable, declare our variable, and initialize our variable.

Let's get started with naming our variables.

Java has a few rules for naming a variable. It can contain case sensitive letters, numbers, dollar sign, and underscore. This rule is important because the variables are case sensitive. We make sure to enter an uppercase letter when it should be uppercase and a lowercase letter when it should be lowercase.

The next rule – it can begin with a letter, dollar sign, or underscore. Most of the times, a variable will begin with a letter. These 2 rules are what we can do with a Java Variable. Now, let's see what we cannot do with a Java Variable.

The next rule – it cannot begin with a number

It cannot contain a space or special character except a dollar sign or underscore. The only special character, I've seen in a variable is an underscore.

The last rule is – it cannot contain a Reserve keyword. A Reserve keyword is reserved for special actions. They have a predefined meaning in Java.

Here's a list of the 50 reserved keywords in Java. Notice our 8 Primitive Data Types are included in this list. They have a special use for defining our data: boolean, byte, char, double, float, int, long, and short

Next is Variable Declaration.

Variable declaration is declaring a variable exist in our class. All variables are associated with a data type. That's why we see in our syntax Data Type is written before the variable name. Variable name is the name of the variable being declared.

Let me back up and explain syntax. Syntax is a set of rules that specify a combination words and symbols. The words in this syntax is Data Type and Variable Name while the symbol is a semi-colon. Semi-colons are written at the end of many statements to complete the statement. It's similar to a period at the end of a sentence.

The last phase for variables is Variable Initialization.

Variable Initialization is when we give an initial value to a variable before the value is used in our program. There is a left side and right side of every variable initialization statement. The left side displays a variable name while the right side displays an expression. An expression is a value assigned to a variable name.

Let's go back to Eclipse and demo Variables. I'm going to add Variables to the Method name.

We are going to start with declaring a variable. The Variable Name is always written after it's Data Type. Our Data Types are int, double, and boolean. The names will be year for int, amount for double, and outcome for boolean. That's how we declare variables: Data Type followed by a Variable Name and Semi-Colon.

When it comes to initializing a variable, there are 2 ways to initialize a variable. We can initialize a variable at declaration or after declaration. At declaration, is when we assign a value on the same line as the Data Type and Variable Name: int year = 2034; After declaration, is when we assign a value on a different line as the Data Type and Variable Name: amount = 23.45; Either way is okay: assigning a value at declaration or after declaration.

Recall the comments regarding int, double, and boolean. Let's make sure the data stored in the variable is correct. We cannot assign any values other than true or false to a boolean Data Type and the same goes for any Data Type. For example, an error will show up if I assign 20 to outcome. The error states cannot convert int to boolean. If I change to 20 include a decimal: 20.20 The error changes to show cannot convert double to boolean.

Let's print the values for each variable:

"What Year Did You Pay $" + amount + " For Gas? " + year

"I Hope That Outcome Is " + outcome

Run

What Year Did You Pay $67.89 For Gas? 2034

I Hope That Outcome Is false.

Java Operators

Next is Java Operators.

There are several types of Java Operators but in this video, I will cover 4 operators. First, is the Assignment Operator. Second, are the Arithmetic Operators. Third, are the Relational Operators. Fourth, are the Logical Operators.

The Assignment Operator assigns a value to a variable. In this syntax example, I have variable name, equal sign then expression. Sometimes a person may say Variable Name but most of the times people say variable for short. There are times a person may say expression. Other times a person may say value. All of those terms are used interchangeably in Java. The purpose is to store data in a memory location so that data can be used later in your program.

Next, are the Arithmetic Operators: Addition, Subtraction, Multiplication, and Division. Addition is a plus symbol that adds a value on both sides of the operator. The same symbol is also used for joining strings. We saw this example when printing to the Console. Let's go back to Eclipse.

The 1st print statement shows, plus amount plus For Gas + year. These plus symbols are used to join the string written in red with the variable. The 2nd print statement shows, plus outcome. This is known as string concatenation. We also see the assignment operator 3 times when assigning a value to year, amount, and outcome.

Subtraction uses a hyphen symbol that subtracts the right operand from the left operand. An operand is an object manipulated by an operator.

Multiplication uses an asterisk symbol for multiplying values. Last but not least, is Division which uses a forward slash symbol for dividing the left operand by the right operand. These are the 4 main Arithmetic Operators.

In addition to these 4 Operators. Java also has a Modulus Operator which uses a percent symbol to divide the left operand by the right operand then returns the remainder. For example, 5 modulus 4 will return 1. 1 is the remainder of 5 divided by 4. The modulus operator is no big deal because most projects do not use this operator.

However, we use an Increment Operator, which increases the operand's value by one. This operator adds 1 to the value. That's why we see 2 plus symbols. The Decrement Operator is the opposite of an Increment Operator. A Decrement Operator decreases the operand's value by one.

We can use the Increment and Decrement Operators as Pre or Post. Pre-Increment and Pre-Decrement places the plus or minus symbol in front of the variable name. Post Increment and Post Decrement places the plus or minus symbol after the variable name. The difference between Pre and Post is: Pre-increments and Pre-decrements performs their operation before using the variable while Post increments and Post decrements performs their operation after using the variable.

Let's take a look at a couple of Arithmetic Operators.

(@Test / public void demoArithmeticOperators) Let's assign a value to total by using the Addition Operator: int total = 3 + 4 then print the value sysout "What Is The Total Of 3 + 4? " + total

Run – What Is The Total Of 3 plus 4? 7

Let's use the Pre-Increment Operator: int incrementTotal = ++total then print the value sysout "What Is The New Total After Incrementing 3 + 4? " + incrementTotal

Run – The New Total Is 8 because 3 + 4 is 7 then increment 7 by 1 equal 8. This is similar to 7 plus 1 equal 8.

Next are the Relational Operators which are used to compare values. There are 6 operators that verifies the relationship between the left operand and right operand. Equal To verifies if both operands are equal. In Math, we use 1 Equal Symbol but in Java we use 2 Equal Symbols for Equal To. Not Equal To has an exclamation point and 1 Equal Symbol to verify if both operands are not equal.

Greater Than has a closing angular bracket symbol that verifies if the left operand is greater than the right operand. Greater Than Or Equal To combines 2 symbols: the greater than and equal to symbols that verifies if the left operand is greater than or equal to the right operand.

Less Than is an opening angular bracket that verifies if the left operand is less than the right operand. Less Than Or Equal To combines 2 symbols: the less than and equal to symbols to verify if the left operand is less than or equal to the right operand.

Let's demo the Relational Operators.

(@ Test / public void demoRelationalOperators) This time, we are going to assign x the value of 5 and y the value of 25. Compare both values by printing, Is 5 Equal To 25?, + (x == y). We use 2 Equal Symbols. Relational Operators always return a boolean value of true of false.

Let's write another print statement "Is 5 Less Than 25? + (5 < 25)". This time, I used the values directly rather than using x and y. An error will show up if we do not surround the variables or values with a parenthesis.

The error states Incompatible operand types String and int. The String operand type is written in red "Is 5 Equal To 25" while the int operand type is x that's assigned a value of 5. Adding a parenthesis around x and y, informs the compiler we are comparing 2 int operand types.

Let's Run / Is 5 Equal To 25? false and Is 5 Less Than 25? true

Next are the Logical Operators. Logical AND – Logical OR. Logical AND has 2 ampersand symbols that returns true if both operands are true. It also returns false if one operand or both operands are false. Logical OR has 2 vertical lines that returns true if one operand or both operands are true. It also returns false if both operands are false.

Let's go back to Eclipse and demo Logical Operators.

(@ Test / public void demoLogicalOperators) Assign 10 to x and 100 to y. First, let's check the Logical AND operator
sysout "Do You Know If 10 and 100 Is Greater Than 50? " + (x > 50 && y > 50) Are both operands true? No, so this statement will return false

Now, let's check the Logical OR operator. Copy and Paste the And Operator then modify the statement. Change AND to OR. All we need is one operand to be true and this statement will return true.

Run / Do You Know If 10 and 100 Is Greater Than 50? false / Do You Know If 10 or 100 Is Greater Than 50? True

That's it for Understanding Java Variables and Operators. Thank You. You can download the presentation and Java code at https://tinyurl.com/JavaVariablesOperators