Selenium 4 Beginners

Rex Jones II

Java
Loop Statements

Table of Contents

Selenium 4 Beginners

Rex Jones II

Introduction

Slide 1 -Thumbnail

Hello Everybody, Welcome To Selenium 4 Beginners. My name is Rex Allen Jones II. We are going to discuss Loop Statements in Java. Click the Like button and Subscribe to the channel. After clicking Subscribe, click the bell icon to receive notification of each video. The video's Transcript, presentation, and code can be downloaded at https://tinyurl.com/Java-Loop-Statements

Slide 2 – Java Flow Control Statements

Loop Statements are Control Statements which control a program's flow of execution. All loops execute a set of instructions over and over while a condition is true.

Slide 3 – Java Loop Statements

Java provides 3 ways to execute a loop statement. The For Loop, While Loop, and Do-While Loop. The For Loop executes a statement after initializing a variable, adding a condition, and setting an iteration expression. The While Loop checks for a condition then determines if a statement should be executed but the Do-While Loop executes a statement first then checks a condition to determine if it should continue executing the statement.

Demo For Loop

Slide 4 – Java Loop Statements / For Loop

Let's demo the For Loop.

Eclipse IDE

Increment

We will start by writing 2 print statements. sysout "Start For Loop" and sysout "Stop For Loop \n". The back slash n is used for skipping a line. Both print statements will be our indicator when the loop start and stop. Next we write for then CTRL + SPACE and select the first option. Modify the statement by replacing i < array.length with i <= 10.

The keyword for starts the for loop. We have 3 statements within the parentheses separated by a semi-colon. The 1st statement is int i = 0, 2nd statement is i <= 10, and 3rd statement is i++.

int i = 0 is an initialization which sets the initial value of the loop control variable. In this case, the loop control variable is i and will start at 0. The 1st statement operates like a counter that controls the loop.

i <= 10 is a condition. This condition is a Boolean expression that determines if the loop repeats or not. The loop repeats if the condition is true but does not repeat if the condition is false. Our loop starts at 0 and stop when the condition is less than or equal to 10.

i++ is an iteration. An iteration defines the number of times the counter variable changes after each loop. This iteration will increment by 1. Therefore, the counter variable which is i will increase by 1 after each loop.

i++ is short for i = i + 1 which adds 1 to i. The short version of i = i + 1 is i += 1. All 3 versions add 1 to i. However, the default and conventional way to write an iteration is i++. Last is the body sysout “i = ” + i which is a statement that will get executed when the condition is true.

Let’s Run. The counter variable start printing at 0 and stop printing at 10. The condition keeps checking until i is less than or equal to 10.

I’m getting ready to show you through Debug so you can see how the for loop operate behind the code. Add a break point at Line 14. Debug. We see execution stopped at the for loop. Watch the starting value get set to 0 after I Step Into. So far, the starting value has been set and our program verified 0 is less than 10. Since the condition is true, execution stopped at the print statement.

Step Into and i = 0 is printed to the console. Execution is back at the for loop. Our program is now finished with the initialization statement int i = 0. From this point, the focus will be on the condition and iteration. Iteration i++ will increment the value by 1 then check the condition and verify if the value is less than or equal to 10.

Step Into and i is set to 1. Step Into again and the console shows i = 1. It will continue this same process of incrementing the value and checking the condition until counter variable i is no longer less than or equal to 10.

I’m going to Step Into until i = 10 using F5. i = 10. Now when I Step Into, the program will exit the for loop. We see Stop For Loop is highlighted. Step Into for the last time then we see the Console shows Start For Loop, counter variables from 0 to 10, and Stop For Loop.

This for loop shows what happens when the condition is true but look what happens when the condition is false. Change the condition to greater than or equal to. So starting out, 0 is not greater than 10.

Let’s Run. All we see is Start For Loop and Stop For Loop. The loop started and stopped but the code inside the for loop did not execute because the condition is false.

By mistake, we can create an infinite loop. An infinite loop is a loop that never terminates because the condition is always true. For example, let’s change the starting value to 20. This condition remains true because starting out 20 is greater than 10.

Let’s Run. Do you see what’s happening in the console? Execution will never stop so we must click the red button to Terminate our execution.

Let me change the values 20 to 0 and make the condition less than or equal to.

Decrement

I have a question for you. How would you count backwards from 10 to 0 using the for loop? We must use the decrement operator. Write the print statements. sysout “Start For Loop“ and sysout “Stop For Loop \n”

for then a parentheses, and both brackets. Within the parentheses, I'll write the initialization statement int x = 10 and a semi-colon. This makes our starting value 10. Next is the condition, x >=0 and a semi-colon. The first check will be true because 10 is greater than 0. Last is the iteration x—which will decrease the counter variable. Execution statement is sysout x = append x.

Let's Run. We see x starts at 10 and stops at 0. If you wanted to decrease the value by a number greater than 1, you can write x -= 2 to decrement by 2. Let's run and you will see what happens. The Console shows 10, 8, 6, 4, 2, 0.

Demo While Loop

Slide 5 – Java Loop Statements / While Loop

Let's demo the While Loop.

Eclipse IDE

sysout "Start While Loop" and sysout "Stop While Loop \n"

Start by assigning a value like int counter = 0 then write keyword while which begins the syntax. We have the option of writing our while loop from scratch or use intellisense CTRL + SPACE and select the 3rd option while loop with condition.

From scratch, we write while then a parentheses and 2 Brackets. It will execute sysout "counter = " + counter over and over depending on the condition. Let's write while counter < 100 print counter equals a value. However, if I execute now, this while loop will create an infinite loop. The iteration step is missing so the code will never terminate. Let me show you. See how it keeps running.

We stop the loop by writing counter ++ to increment counter by 1. We have initialized the counter variable to 0 and the loop will repeat as long as the value is less than 100. The counter increments every time after printing the execution statement. Do you know what the last counter value will be since our condition is less than 100? Let's Run. The last value is 99. Soon as the counter value reaches 100, the program stops looping and exit the while loop.

Recall from the for loop that ++ is short for + 1. Therefore, counter ++ can be written as counter = counter + 1. Run again. Another shortcut is counter += 1 which will give us the same result. Let's increment by 10 then Run. The last value is 90.

I want to show you a false condition by changing less than 100 to greater than 100. 0 is not greater than 100 so the counter print statement will not execute. We see the While Loop started and stopped but no execution statement. Change back to less than 100. That's it for while loop.

Demo Do-While Loop

Slide 6 – Java Loop Statements / Do-While Loop

Let's demo the Do-While Loop.

Eclipse IDE

Write 2 print statements sysout “Start Do-While Loop“ and sysout “Stop Do-While Loop”.

Initialize the loop variable by writing int y = 0. Next write the keyword do which begins the syntax, CTRL + SPACE, and select the do-while statement. I’m going to add sysout “y = ” + y within the brackets then y ++ as the iteration so the loop can stop.

The while condition is written after the loop’s body. We want to execute the body while y is less than or equal to 20. The Do-While Loop will execute the print statement at least one time before checking if the condition is true or false. If the condition is false then the body execute one time. I’m going to make the condition false by changing the starting value from 0 to 100.

Here’s the breakdown. The do keyword starts the loop, executes the print statement by printing y = 100. It’s only supposed to execute while y is less than or equal to 20 so this statement will not execute anymore. Let’s run. We see the do-while loop start and execute y = 100 one time before stopping.

Change 100 back to 0 then Run. y starts at 0 and stops at 20.

We set the value to 0, started the loop by writing keyword do, added a print statement for execution, incremented the value by 1 then check for a condition while y is less than or equal to 20.

Recap All Loops

Let’s Recap all 3 Loop Statements.

Slide 7 – For Loop

The for loop starts by writing keyword for with 3 statements. The 1st statement is initialization which sets the counter variable. Counter variables can also be called loop variables because it is a variable that controls the loop. We have to make sure to complete the initialization statement with a semi-colon.

The 2nd statement is condition. A condition is a boolean expression that determines if the loop will repeat or not repeat. If the condition is true, the loop will repeat but it will not repeat if the condition is false. The condition also ends with a semi-colon.

However, the 3rd and final statement is iteration which does not have a semi-colon. We can increment or decrement the counter variable after each loop. Next we have an opening bracket followed by a statement or block of statements then a semi-colon. The closing bracket complete the for loop.

Slide 8 – For Loop Flow Chart

Let’s use the flow chart because some people get a better understanding watching the flow. We start by writing for followed by an initialization. After the initialization, is the condition. If the condition is false, then that stops the for loop. However, if the condition is true then our body which is a statement is executed. Next, we have an iteration before going back to the Condition. The loop repeats until the Condition is false.

Slide 9 – While Loop

The while loop starts by writing keyword while and a condition. An initialization statement is not in this slide but it helps the condition determine if the while loop should repeat or not repeat. The condition can be any boolean expression. We have an opening bracket then our statement body and semi-colon. Notice how the condition is checked at the top of the loop before writing a statement. The statement is only executed if the condition is true. One more bracket closes the while loop.

Slide 10 – While Loop Flow Chart

Here's the While Loop Flow Chart which starts with while. First we see the Condition. If the Condition is false, the while loop bypasses execution and stop. If the Condition is true, the while loop executes a statement then increments or decrements the value. Next the process continues all over again until the condition becomes false.

Slide 11 – Do-While Loop

Last is the Do-While Loop and it starts with keyword do followed by an opening bracket. Then we have a statement and semi-colon. This guarantees the statement will get executed at least one time. After the statement is a closing bracket. Keyword while and a condition is last. Notice how the Do-While Loop is different from the for loop and while loop. The condition is at the bottom of the loop but the for loop and while loop has a condition at the top of the loop.

Slide 12 – Do-While Loop Flow Chart

The flow chart also shows keyword do and a statement. Statements automatically gets executed once before an iteration. After the iteration, increments or decrements a counter variable, we have our while condition. If the while condition returns false, then the program stops running but it will keep running if the while condition is true. The do-while loop continues executing, iterating, and checking the while condition until false is returned from the condition.

Slide 13 – Which Loop Do I Use?

Java is flexible when it comes to loops. How do you know which loop to use for a certain task? I recommend using the for loop when you know how many times the loop should run. The While Loop should be used when you do not know how many times the loop should run. The Do-While Loop is used when the loop must run at least one time.

Download Documents

That's it and Thank You for watching Java's Loop Statements. If you are interested in the Transcript, Presentation, and Code, go to https://tinyurl.com/Java-Loop-Statements