Selenium Automation Testing

dridhOn

Course Overview:

dridhOn Automation Testing Course helps you learn Automation testing in Selenium using Python or Java and become a certified Automation Tester. As part of the course, you will learn Automation Testing in Selenium tool components such as Selenium IDE, RC, Selenium WebDriver, and Selenium Grid through hands-on projects and case studies to ensure that you are Industry ready upon completion of the course.

Training Features:

- 8X higher interaction in live online classes conducted by industry experts
- 120 Hrs. live Classes of RPA Developer with Interview Preparation
- 3 real-time industry projects with hands-on preparation
- Unlimited Interview Opportunities with Placement Support
- Industry-recognized course completion certificate

Delivery Mode:

- Online Live Virtual Instructor Led Training

Target Audiance:

- The Basic Requirement to start a career as an Automation tester, you'll need a Bachelor's degree or at least 1+ years of experience in Information Technology (IT). A Bachelor's degree in Technology justice will help you get the job.

Key Learning Outcomes:

- Python
- Java
- Seleinum
- Web Driver
- Appium
- IDE
- OOPS
- Framework Testing
- RC
- Seleiun Grid
- Server Testing
- Manual Testing
- Agile Method

Certification Details:

- Complete at least 85 percent of the course or attend one complete batch
- Successful completion and evaluation of the project

dridhOn

Getting Started with Selenium Web Driver and Python:

- Preparing your machine
- Installing Python
- Installing the Selenium package
- Browsing the Selenium Web Driver Python documentation
- Selecting an IDE
- PyCharm
- The PyDevEclipseplugin
- PyScripter
- Setting up PyCharm
- Taking your first step with selenium and Python
- Cross-browser support
- Setting up Internet Explorer
- Setting up Google Chrome
- Summary

Writing Tests Using unit test:

- The unit test library
- The Test Case class
- The set Up()method
- Writing tests
- Cleaning up the code
- Running the test
- Adding another test
- Class-level setUp() and tearDown()methods
- Assertions
- Test suites
- Generating the HTML test report
- Summary

Finding Elements:

- Using developer tools to find locators
- Inspecting pages and elements with Fire focusing the Firebugadd-in
- Inspecting pages and elements with Google Chrome
- Inspecting pages and elements with Internet Explorer
- Finding elements with Selenium Web Driver
- Using the find methods
- Finding elements using the ID attribute
- Finding elements using then a meat tribute
- Finding elements using the class name
- Finding elements using the tag name
- Finding elements using XPath
- Finding elements using CSS selectors
- Finding links
- Finding links with partial text
- Putting all the tests together using find methods
- Summary

Using the Selenium Python API for Element Interaction:

- Elements of HTML forms
- Understanding the Web Driver class
- Properties of the Web Driver class
- Methods of the Web Driver class
- Understanding the Web Element class
- Properties of the Web Element class
- Methods of the Web Element class
- Working with forms, text boxes, checkboxes, and radio buttons
- Checking whether the element is displayed and enabled
- Finding the element attribute value
- Using the is selected()method
- Using the clear()and send keys()methods
- Working with drop-down sand lists
- Understanding the Select class
- Properties of the Select class
- Methods of the Select class
- Working with alert sand pop-up windows
- Understanding the Alert class
- Properties of the Alert class
- Methods of the Alert class
- Automating browser navigation
- Summary

Synchronizing Tests:

- Using implicit wait
- Using explicit wait
- The expected condition class
- Waiting for an element to be abled
- Waiting for alerts
- Implementing custom wait conditions
- Summary

Cross-browser Testing:

- The Selenium stand-alone server
- Downloading the Selenium stand-alone server
- Launching the Selenium stand-alone server
- Running a test on the Selenium stand-alone server
- Adding support for Internet Explorer
- Adding support for Chrome
- Selenium Grid
- Launching the Selenium server as a hub
- Adding nodes
- Adding an IEnode
- Adding a Firefox node
- Adding a Chromenode
- Mac OSX with safari
- Running tests in grid
- Running test Sina cloud
- Using SauceLabs
- Summary