Automated Testing with Selenium

SKILLSOFT ASPIRE JOURNEY

skillsoft

Głównym wyzwaniem przed którym stają dziś organizacje na całym świecie jest konieczność ciągłego podnoszenia umiejętności i poziomu wiedzy w ślad za gwałtownym rozwojem nowych technologii i zmian na globalnym rynku.

Stały rozwój i podnoszenie kwalifikacji w IT od dawna jest już rzeczą oczywistą, a możliwość zapewnienia wsparcia specjalistom chcącym stale się rozwijać jest jedną z głównych kart przetargowych w walce o pracownika.

Na rynku liczą się dziś ludzie, którzy posiadają konkretne kompetencje i zestaw umiejętności pozwalający im wykonywać zadania efektywnie, a nie Ci z najdłuższym stażem pracy.

Dziś, bardziej niż kiedykolwiek w cenie jest umiejętność budowania ścieżki kariery dla profesjonalistów IT, którzy wciąż chcą się liczyć na rynku pracy.

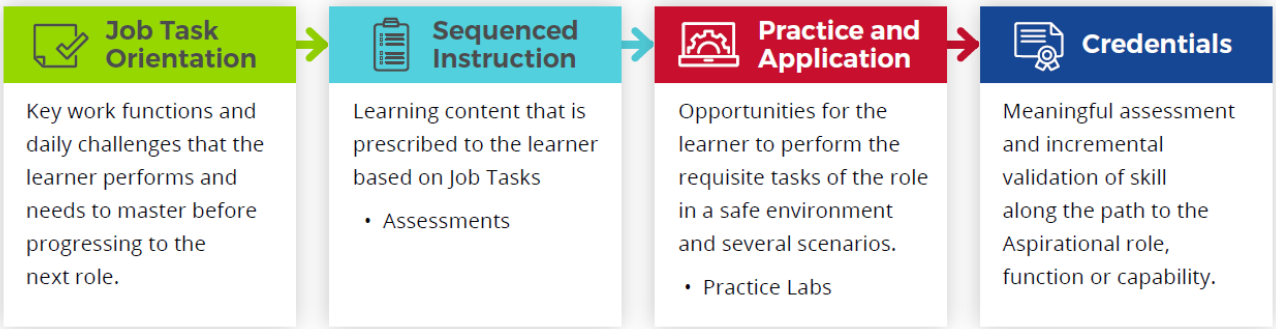
Skillsoft Aspire Journey stanowi odpowiedź na pytanie, jakie szkolenia muszę ukończyć, aby być przygotowanym do mojej wymarzonej pracy. Spośród kilkuset kanałów tematycznych dostępnych na naszej platformie szkoleniowej nasi specjaliści wybrali te, które naszym zdaniem najlepiej wyposażą uczących się w narzędzia potrzebne do realizacji zadań w nowej roli.

Skillsoft Aspire Journey to zestawy szkoleń i ćwiczeń w języku angielskim, które metodycznie, krok po kroku pozwalają specjalistom przejść od poziomu podstawowego do zaawansowanego.

Każda ścieżka zawiera szkolenia, laboratoria wirtualne, video i książki, które pomogą uczącym się osiągnąć pożądane kompetencje poświadczone certyfikatem.

Aspire Journey Model

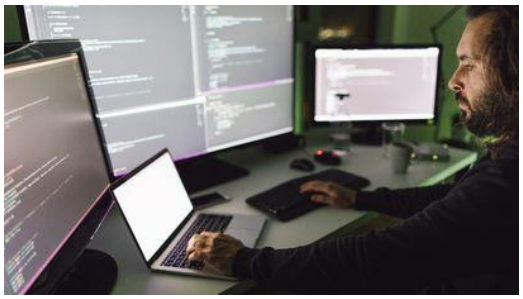
Cała ścieżka opiera się na 4-elementowym cyklu powtarzanym na kolejnych etapach nauki.



1. Określenie kluczowych funkcji i wyzwań, z którymi musi poradzić sobie uczący się w chwili obecnej, jak i tymi, z którymi przyjdzie mu się zmierzyć w nowej pracy.
2. Przejście zaprojektowanych ścieżek w proponowanej kolejności, wykonanie ćwiczeń i zaliczenie testów.
3. Przećwiczenie nowych umiejętności w kontrolowanym środowisku w oparciu o gotowe scenariusze działań. Laboratoria wirtualne Skillsoft
4. Certyfikat – zaliczenie testu końcowego na poziomie co najmniej 70% i uzyskanie certyfikatu potwierdzającego ukończenie danego etapu nauki.

Aspire Journey – AI Apprentice to AI Architect

Analizując trendy opisujące zachowanie użytkowników na naszych platformach szkoleniowych i współpracując ściśle z naszymi klientami na całym świecie Skillsoft wyselekcjonował najlepsze materiały szkoleniowe i ułożył je w ustrukturalizowaną ścieżkę rozwoju. Ścieżka zawiera ponad 18 godzin szkoleniowych.



16 courses
18h 6m 41s

- Selenium IDE
- Selenium WebDriver
- Selenium grid and patterns
- Selenium framework.
- Component testing,
- Automating browser windows
- Authentication tests
- Managing synchronization and page navigation.
- Test automation using Selenium
- Managing modern application components using Selenium
- Managing data elements using Python and Selenium
- Integrating TestNG and Selenium
- Applying BDD and TDD using Selenium.
- Selenium test enhancement using BrowserStack
- Database and log management in Selenium testing
- Use cases for Web app component automated testing
- Testing single-page applications

Prerequisites

- Knowledge of software testing
- Being familiar with Selenium

Track 1: Automated Testing (duration: 18h 6m 41s)

<table>
<tr>
<td> 

Selenium Deep Dive: Setting Up Selenium for Automated Testing</td>
<td> 

Selenium Deep Dive: Working with Selenium IDE</td>
</tr>
<tr>
<td>Objectives:

▪ recall essential Selenium features and the different types of application testing that can be automated using Selenium
▪ recognize the key components of Selenium that are used to describe test suites and test steps involved in managing the Selenium testing life cycle
▪ recognize Selenium Automation Framework features and the steps involved in implementing test automation
▪ set up Selenium on Python and Java environments
▪ configure Selenium with PyCharm and Eclipse to set up Selenium test automation environments for Python and Java
▪ create a sample test script to illustrate how to write and execute a Selenium test script in Python using the PyCharm IDE
▪ configure Eclipse with the PyDev plugin and create a sample test script to illustrate how to write and execute a Selenium test script in Python using the Eclipse IDE
▪ create Selenium tests in Python with Unittest and create a cleanup strategy to free resources post test execution
▪ generate an HTML test suite execution report for Selenium Test Suites using Unittest and HTMLTestRunner
▪ use various assertions in Unittest to create productive and simplified Selenium test cases</td>
<td>Objectives:

▪ recall the features of the Selenium IDE that make it a fast prototyping tool and recognize its essential components
▪ install the Selenium IDE and create test cases using the record-and-playback functionality of Selenium IDE
▪ configure the Selenium IDE Options Dialog Box and create and modify Selenese commands using the Editor
▪ create and execute test scripts in Selenium IDE using the record-and-playback feature
▪ list the set of Selenese commands that can be used in Selenium IDE to manipulate, examine, and verify the state of applications
▪ create and execute test cases manually using Selenese commands in Selenium IDE
▪ work with the Element Verification commands in Selenium IDE to improvise automation
▪ work with Wait commands to test page loading using Selenium IDE
▪ demonstrate test automation using the Selenium IDE to test modal and modeless dialogs for applications
▪ create and manage customized Selenium user extensions in the Selenium IDE using JavaScript</td>
</tr>
</table>

Selenium Deep Dive: Working With WebDriver

Objectives:

- recall the disadvantages of Selenium RC and list the benefits of using WebDriver as a browser automation framework
- recognize prominent components of WebDriver and how the architecture of Selenium WebDriver can help APIs to interact with browsers to automate tests
- set up GeckoDriver and launch Firefox browser to initiate testing components of application
- use the Selenium WebDriver navigation methods to effectively manage browser navigation scenarios
- recognize the role of locators in Selenium and list the prominent types of Selenium locators
- identify basic locator elements like the ID, Name, and LinkText using developer tools
- select elements of an application page using Class Name and Tag Name
- locate application page elements using an XPath Expression
- use CSS selectors to identify one or more elements in a web page and describe the benefits of using CSS selectors instead of XPath
- work with selectors to select web elements of modern applications
- generate customized XPath selectors from HTML attributes
- determine get or fetch attribute values using scripts rather than using developer tools
- generate customized CSS selectors from HTML attributes
- summarize the key concepts covered in this course

Selenium Deep Dive: Working with Selenium Grid & Patterns

Objectives:

- recall the key features of Selenium Grid and describe scenarios where Selenium Grid is a right fit
- recognize the architecture of Selenium Grid along with the differences between Grid 1 and Grid 2 versions
- configure Hub and Node for Selenium Grid Server and describe the use of the Grid console
- execute WebDriver tests in parallel using Selenium Grid
- troubleshoot errors while invoking browser code
- implement Page Object Model using Selenium
- implement Page Object Pattern to create layer of separation between test code and technical implementation
- use Object Repository Mechanism in Page Object Pattern and Page Factory
- fetch data in Selenium from Excel documents
- fetch data in Selenium from Excel documents using DataProvider
- handle Windows authentication pop-ups using Selenium

<table>
<tr>
<td> 
</td>
<td>Selenium Deep Dive: Component Testing with Selenium Framework</td>
<td> 
</td>
<td>Selenium Deep Dive: Automating Browser Windows & Authentication Tests</td>
</tr>
<tr>
<td colspan="2">Objectives:

▪ recall prominent application architectures and their components
▪ recognize the features of legacy and modern applications that can impact test automation using Selenium
▪ use the WebDriver Select method and work with the selectByIndex method to perform operations on a Select drop-down
▪ manage inline frames while defining test mechanisms using Selenium WebDriver
▪ manage JavaScript alerts using the Selenium WebDriver Alerts API
▪ use the selectByValue and deselectByValue methods to select and deselect all options that have values matching the specified arguments
▪ use the selectByVisibleText and deselectByVisibleText methods to select and deselect all options that display text matching the specified arguments
▪ write Selenium scripts to automate operations on a Select drop-down and retrieve the text of the values in the drop-down
▪ work with checkboxes in a WebDriver scenario
▪ use Selenium to test scenarios using multiple checkboxes and safechecks
▪ write Selenium scripts to handle and test confirmation pop-ups in an application
▪ write Selenium scripts to handle and test prompt pop-ups in a real-life scenario</td>
<td colspan="2">Objectives:

▪ list prominent web application technologies and frameworks that are used to build applications
▪ use Selenium scripts to handle multiple advertisement windows that hinder testing
▪ write Selenium scripts to handle basic authentication while working in the pre-production stage on Firefox
▪ write Selenium scripts to handle security pop-ups on Google Chrome
▪ install AutoIt and use it with Selenium WebDriver to handle and test authentication windows
▪ test use cases where you need to open new windows and perform operations
▪ switch from a default window to the desired window to perform operations while testing, and switch back to the default window to work on the main window
▪ test AutoComplete textbox using Selenium by providing pre-populated lists in applications
▪ upload files using the sendKeys method without using any third-party tools
▪ upload files by transferring the control from Selenium WebDriver to AutoIt</td>
</tr>
</table>

Selenium Deep Dive: Managing Synchronization & Page Navigation

Objectives:

- recognize the concept of synchronization and differentiate between conditional and unconditional synchronization
- list methods that can be used to handle keyboard and mouse events with Selenium WebDriver
- implement unconditional synchronization using the Selenium WebDriverWait and Thread.Sleep methods
- implement conditional synchronization using the Implicit Wait function of Selenium WebDriver
- use the Implicit Wait and Fluent Wait classes provided by Selenium WebDriver
- set browser width and height using Selenium WebDriver
- automate the resizing of web elements using Selenium scripts
- automate scrolling down to the bottom of web pages using JavaScript Executor
- use coordinates to scroll to particular positions on web pages by passing the coordinates using Selenium WebDriver
- automate scrolling to particular elements of web pages and perform operations on the elements
- download files using Selenium WebDriver and define browser settings with Firefox Profile preferences

Selenium Deep Dive: Security Test Automation

Objectives:

- outline the security mechanisms implemented in applications and their impact on test automation
- demonstrate the process of handling untrusted SSL certificates in Firefox while automating tests with Selenium
- show the process of handling security certificates in Chrome while automating tests with Selenium WebDriver
- manage security certificates in Internet Explorer while implementing test automation with Selenium
- perform parallel execution of tests using TestNG and Selenium
- illustrate how to handle a stale element exception error using Selenium WebDriver
- add cookies while automating test executions using WebDriver
- demonstrate the process of deleting cookies from a browser while automating tests with WebDriver
- delete named cookies from the current domain while automating test executions using Selenium
- call document cookie and parse results using Selenium WebDriver
- return the values of a cookie with a specific name and return null if no cookie is found with the given name

Selenium Deep Dive: Managing Modern Application Components

Objectives:

- read data from properties files using Java and Selenium and then use the read data to automate tests
- take screenshots when tests fail using Selenium WebDriver
- automate testing drag and drop functionality using WebDriver's Action's class
- automate testing the context menu functionality of applications using WebDriver's Actions class
- automate testing calendar scenarios using the jQuery DatePicker
- automate application testing using Kendo UI DatePicker
- demonstrate how to handle "Element is not clickable at point" exception scenarios with Selenium WebDriver
- fetch texts from PDF files while automating test executions using Selenium, Java, and TestNG
- use the HTTP client library to check the status codes of images on web pages and implement automated testing to find broken or invalid images
- find broken links on web pages using Selenium WebDriver and the HTTP client

Selenium Deep Dive: Managing Data Elements Using Python & Selenium

Objectives:

- demonstrate how to implement test execution in Python using Selenium RC Server
- automate the process of extracting email addresses from web pages using Selenium Python RegEx
- automate the process of extracting phone numbers using Selenium Python RegEx
- write automation test scripts to find all href elements using the XPath module provided by Selenium Python
- demonstrate the process of getting texts from image elements using Selenium Python
- show how to automate tests that verify JavaScript code execution using Selenium Python
- implement the Python exception logging module to capture full stack trace in the context of the try and except block in Python to catch and handle exceptions
- use Selenium Python and the ActionChains class to automate low-level interactions, such as mouse movements and mouse button actions
- implement WebDriver conditional commands using Selenium Python, to automate tests based on specified conditions
- use Selenium Python to fetch all available attributes using the "By class" locator provided by Selenium WebDriver

Selenium Deep Dive: Integrating TestNG & Selenium

Objectives:

- install and configure TestNG for productive test case authoring
- demonstrate the process of defining and creating multiple test cases using TestNG
- illustrate the process of using autogenerated XML and reports
- create dependencies between test cases using a real-time use case scenario
- integrate TestNG and Selenium and create dependencies between test cases using Selenium WebDriver and TestNG
- implement WebDriver listeners to demonstrate their usage in test automation scenarios
- implement TestNG listeners in Selenium Webdriver to automate the execution of test cases
- implement Hard Assertions in Selenium TestNG
- implement Soft Assertions in Selenium TestNG
- create a TestNG XML file and demonstrate its implementation in test automation

Selenium Deep Dive: Applying BDD & TDD Using Selenium

Objectives:

- define the concept of behavior-driven development (BDD) and list the benefits of test automation using BDD and Selenium
- install and configure the behave BDD framework, create a feature file, and generate the code scaffolding for it
- automate functional tests by passing arguments to step definitions using the behave framework and the step consolidation pattern
- manage page objects of a web application by implementing a step definition to adopt a bottom-up approach to page object planning
- model page object implementation with BDD frameworks and Selenium WebDriver, using step definition files to implement the Singleton design pattern and interface
- define the concept of test-driven development (TDD) and describe the benefits of using TDD with Selenium WebDriver
- demonstrate the TDD approach to writing functional tests for Django projects using Selenium WebDriver
- configure BDD environments by both integrating Cucumber and Selenium and by using Selenium WebDriver with Java
- interpret the behavior of data by implementing data-driven tests using a Cucumber data table
- implement functional test automation of form-based web pages using Cucumber and Selenium WebDriver

Selenium Deep Dive: Test Enhancement Using BrowserStack

Objectives:

- list the products provided by BrowserStack along with their usage in test automation with Selenium
- set up test environments using BrowserStack to test URLs in internal networks and configure the Selenium Test Suite to automate tests
- run multiple Selenium WebDriver tests in parallel through the same BrowserStack account
- configure BrowserStack to run Selenium WebDriver test cases on desktop and a remote Firefox browser
- configure BrowserStack to automate Selenium test executions on diversified browsers that are installed on various OSs and devices
- manage self-signed certificates and bypass bad SSL errors on non-navigation actions while test execution is in progress
- enable and disable Flash and popups by configuring Chrome and Firefox profiles for productive test automation
- list the advanced capabilities of BrowserStack that can be used to enhance functional testing with Selenium
- integrate Selenium and Lettuce to run automated tests in cloud environments
- integrate Selenium and Behave to run automated tests in cloud environments

Selenium Deep Dive: Database & Log Management in Selenium Testing

Objectives:

- recall the role of database and log management in functional test automation
- describe the relevance of Selenium Python's EventListener and EventFiringWebDriver
- demonstrate the use of Listener to track before and after statuses when a click event is fired on web components under test with Selenium Python
- utilize databases to define and store test data and use it to automate functional testing with Selenium Python
- use Selenium Python to write test scripts for managing cookies that store session-specific web application data
- demonstrate the use of Python Imaging Library to crop elements out of web pages with Selenium Python
- locate frames and manage nested iframes in Selenium Python
- define the concept of logging and list the various log levels supported by Python and available to use in Selenium Python
- write scripts using Selenium Python to manage logs and prepare audit trails during automated test execution
- list the exceptions that can be raised while using Selenium Python to automate functional test cases

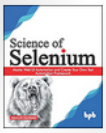
 Selenium Deep Dive: Use Cases for Web App Component Automated	 Testing Selenium Deep Dive: Testing Single-page Applications
Objectives: ▪ describe the lifecycle of web applications and the typical taxonomy involved in managing common web application functional scenarios ▪ automate testing the mandatory fields of a form for correct value validation ▪ automate web page testing to ensure all mandatory form fields are marked with an asterisk sign ▪ automate testing scenarios where the systems or applications under test should not display error messages for the optional fields in the form ▪ automate testing to validate the year fields for leap year validation and correctness ▪ automate numeric field testing to ensure only valid literals can be accepted and in cases where invalid literals are entered the correct error messages are displayed ▪ automate testing scenarios where users cannot enter negative numbers or zero on the text fields of a web page ▪ automate testing scenarios where the currency field on a web page displaying monetary values is displayed in the correct format ▪ automate testing scenarios where confirmation messages must be displayed whenever users initiate resource deletion or updating on web pages ▪ automate testing to confirm a privacy policy is defined and displayed and that users are not allowed to move on to the next step until the privacy policy is accepted ▪ automate testing scenarios where in the event of any functionality failure the users are redirected to a custom error page ▪ automate testing to confirm proper usage of cookies as per the Web 2.0 standards ▪ automate the testing of appropriate implementation of multi-factor authentication ▪ automate testing scenarios where the animated GIFs must show similar animation effects on all browsers ▪ automate testing scenarios where a user's account is locked after three failed password attempts, and the user is notified via email	Objectives: ▪ recall the architecture and benefits of single-page applications ▪ list the prominent testing frameworks that can be used for functional testing of single-page applications ▪ install and configure WebdriverIO and write test scripts to execute tests using a specs file ▪ automate testing scenarios where modern web applications use WebSocket as the networking layer ▪ write programs using Java to test auto suggestions and auto complete text box controls in modern web applications ▪ automate the testing of context menu behavior in modern web applications ▪ demonstrate the implementation of test automation to solve reCAPTCHAs using Selenium ▪ set up and execute Selenium WebDriver test scripts on Android Emulator ▪ execute Selenium WebDriver test scripts on browsers using their mobile user agent ▪ prepare Selenium to conduct web application load testing by allowing users to re-use existing functional tests by executing them with virtual concurrent users ▪ integrate Selenium scripts with JMeter to conduct performance tests ▪ list the essential testing strategies that can be adopted in a microservice architecture and Selenium's role in this ▪ use the Requests package to test API operations with Selenium Python ▪ implement end-to-end testing of modern applications written in React with Selenium WebDriver and Node.js ▪ recognize the limitations associated with Selenium WebDriver that test engineers need to be aware of before planning test architectures

Bookshelf Optional

BOOK

Selenium Framework Design in Keyword-Driven Testing:...

2



BOOK

Science of Selenium: Master Web UI Automation and...

11

BOOK

Hands-On Functional Test Automation: With Visual...

38

BOOK

Selenium WebDriver Recipes in C#, Second Edition

54

FOLLOW US ON:

www.skilltech.pl

email: biuro@skilltech.pl

tel. +48 22 44 88 827

SkillTech
Technology hired for excellence