|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Section** | **Lecture** | **Slide** | **Slide Content** | **Transcript** |
| Introduction | Introduction to Building Automation Frameworks using Selenium and Java | About Me |  | Hi All! My Name is Rajesh Iyer. I am working in the Software Testing industry for over 10 years. I have architected and developed some sophisticated automation frameworks for some renowned clients in my organization. During my tenure I have majorly worked with Selenium RC and Selenium Webdriver, REST API testing and also Performance Testing. Though my educational background does not qualify me to be in the Software Testing industry, I consider myself lucky to get a start. And that was the only luck I had. After that it was all about continuous learning that has lead me to get into more challenging positions and also help some of my junior colleagues in their careers. |
| Why I created this course |  | Right since my school days, I have the passion for teaching. I have conducted trainings for my junior colleagues and also have provided mentorship to a handful of my colleagues. Now I want to widen my horizons and reach out to a lot of people. That is why I have created this course.  The other aspect is, I believe that the Automation is not just about knowing how to use Selenium Webdriver, but there is a science to developing a robust and sophisticated framework, that can be reused over and over to provide:   1. Better ROI for automation 2. Maintainability of the tests 3. Quick Turnaround Time in developing the tests   If you have these goals while developing a framework, your approach will be driven in this direction. The objective of this course is to help you achieve the above goals. |
| What to expect from this course |  | The first thing you can expect from this course is that there won’t be those boring theories about Java, Selenium or any of the different aspect of building frameworks. I don’t want the students of this course get all the information and when they come out of this course wonder “Aww! How do I apply this information now?”  You can expect a lot of interesting stories and real-world examples, that will help you understand every concept in detail and a few recommendations on how to apply the principles you will be learning throughout the course. So are you ready to get started? Let’s first learn Java or for some it will be brush up on Java. |
| Java Programming | Basic Java | Introduction |  | At the beginning of the course, I said I will not be giving the boring theories of Java or Selenium, but I thought it worthwhile to share the concept of how Java actually runs your code.  The program that you write is a .java file.  There is a java compiler or javac program which converts the .java file to a .class file which is a bytecode data.  The core of the Java is the JVM i.e Java Virtual Machine. This JVM is responsible for running your application. The JVM contains a Byte Code Verifier and Class Loader, which will check the byte codes of the class file. Once the check passes, there is another section called the Java Intrepreter and JIT compiler.  Java Interpreter reads the bytecode and performs the functions accordingly. The disadvantage of the Java Interpreter is that it slow as it has to look up the meaning of each byte during execution.  This is overcome by the Just-In-Time Compiler, which compiles the bytecode to the native code just before the execution.  Java can be installed with Java Development Kit (JDK) or Java Runtime Environment (JRE). Both come with JVM packaged into it. However, JDK has some additional Java APIs which will assist the developers in |
| Installation |  |  |
| Creating a class |  |  |
| static keyword |  |  |
| this keyword |  |  |
| super keyword |  |  |
| final keyword |  |  |
| Conditional and Controlling statements |  |  |
| Working with arrays |  |  |
| Creating an abstract class |  |  |
| Creating an interface and implementation |  |  |
| File Handling |  |  |
| String and String Manipulations |  |  |
|  |  |  |
| OOPS Concepts | Regular Expression handling |  |  |
| Polymorphism |  |  |
| Abstraction |  |  |
| Encapsulation |  |  |
| Advanced Java | Collections framework |  |  |
| Generics |  |  |
| Reflections |  |  |
| SOLID principles | Introduction |  |  |
| Single Responsibility Principle |  |  |
| Open for extension, Closed for modification Principle |  |  |
| Liskov Substitution Principle |  |  |
| Interface Segregation Principle |  |  |
| Dependency Inversion Principle |  |  |
| Design Patterns |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Frameworks | Types of Automation Frameworks | Data driven Framework |  |  |
| Keyword driven Framework |  |  |
| Behaviour Driven Framework |  |  |
| Hybrid Framework |  |  |
| Different components of an Automation Framework | Selenium Webdriver |  |  |
| TestNG Framework |  |  |
| Maven |  |  |
| Dataproviders |  |  |
| Custom Page Creation |  |  |
| Object Repository Management |  |  |
| Reporting |  |  |
| Continuous Integration with Jenkins |  |  |
| Selenium Webdriver | Introduction |  |  |  |
| How does Selenium Work? |  |  |  |
| Different implementations of Webdriver |  |  |  |
| Working with UI Components Using Webdriver |  |  |  |
| Handling Waits in Selenium Webdriver |  |  |  |
| Taking Screenshots using Remote Webdriver |  |  |  |
| Working with Actions API |  |  |  |
| Using JavascriptExecutor |  |  |  |
| Conclusion |  |  |  |
| Maven | Introduction |  |  |  |
| What is Maven? |  |  |  |
| What is pom.xml? |  |  |  |
| How to add dependencies to your project |  |  |  |
| How to add plugins to your project |  |  |  |
| Conclusion |  |  |  |
| TestNG Framework | Introduction |  |  |  |
| What is TestNG |  |  |  |
| Difference between TestNG and Junit |  |  |  |
| Different annotations in TestNG |  |  |  |
| Working with Dataprovider annotation |  |  |  |
| Dependent tests using TestNG |  |  |  |
| Grouping of tests in TestNG |  |  |  |
| Running tests in parallel |  |  |  |
| Writing Sample scripts with TestNG |  |  |  |
| Dataproviders | How to manage data in your framework? |  |  |  |
| Understanding the Apache POI API |  |  |  |
| Creating an utility to read an excel file using Apache POI API |  |  |  |
| How to pass data to the TestNG test |  |  |  |
| Object Repository Management | Identifying Web Elements |  |  |  |
| Advanced CSS and XPATH identifiers |  |  |  |
| Using Firebug and Firepath to identify elements |  |  |  |
| Creating Custom Page Methods | What is a Page Factory Pattern? |  |  |  |
| Creating a Page using Page Factory pattern |  |  |  |
| Creating Object Repository for a Page |  |  |  |
| Reporting and Logging | How to report the execution status |  |  |  |
| Adding logs to your tests |  |  |  |
| Integrating the reports using ATU reports |  |  |  |
| Conclusion | Final Thoughts |  |  |  |