Automação de testes I

Conteúdo

Introdução a testes automatizados

1. O que é, porque automatizar e quando automatizar.....
2. Introdução ao Eclipse/IntelliJ…….....
3. Testes web utilizando Cucumber, Selenium…….....
4. Conceitos, construir automação na prática…….....

O que é ?

Automação de testes é o uso de software para controlar a execução de testes de software através da aplicação de estratégias e ferramentas, comparando os resultados esperados com os resultados reais. Seus objetivos são a redução do envolvimento humano em atividades manuais, de tempo demandado e de custo final.

Por que automatizar ?

Eventualmente, quando trabalhamos com desenvolvimento de software, temos que tomar a decisão de automatizar ou não os cenários de testes gerados. Atualmente, automatizamos cenários de testes para evitar trabalho manual em excesso, garantir que não há regressões no software, obter um feedback mais rápido, economizar tempo executando testes repetidos.

Evitar trabalho manual em excesso é um grande motivo para realizar automação de testes em um software. Softwares em desenvolvimento requerem um grande esforço em testes, já que estão mudando constantemente. Quando um testador executa um mesmo cenário de teste inúmeras vezes, isso pode ocasionar erros, como por exemplo, pular algum passo importante do cenário. Afinal, somos humanos e estamos suscetíveis a errar.

Para garantir que não há regressões no software é importante ter um conjunto confiável de testes de regressão. É possível, então, automatizá-los para que você possa executá-los de maneira instantânea sem depender de um trabalho manual excessivo.

Quando automatizar ?

A importância da automação de testes está diretamente relacionada à qualidade do produto final. Assim, ao pensar em automatizar, é preciso estudar a sua viabilidade: com a automação conseguiremos obter ganho de tempo?

Conseguimos reduzir os custos e manter a qualidade?
Se a resposta for sim, outros fatores precisam ser analisados: a maturidade do time de processo do teste; grau de reutilização dos testes automatizados; conhecimento sobre o comportamento que é esperado do sistema a ser testado; e, ainda, o tempo disponível para a automação.

Também deve ser considerado o quão frequentes são as mudanças das funcionalidades a serem verificadas — pode não ser viável automatizar um teste de uma funcionalidade que poderá sofrer alterações amanhã — ; e se é possível garantir que a mesma qualidade de execução manual do teste será mantida em uma execução automatizada.

Leitura técnica

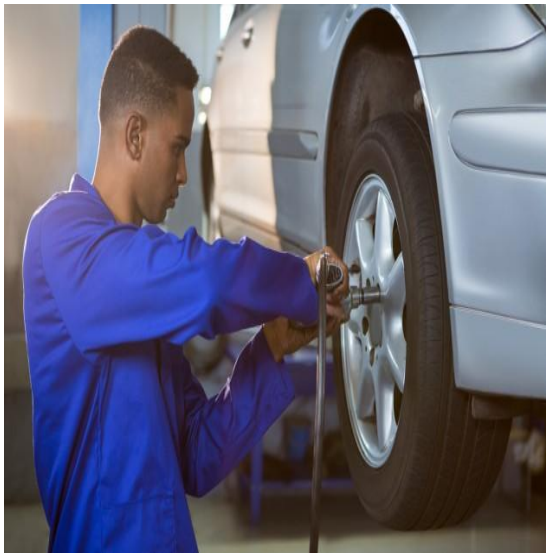
Clique aqui

O que preciso para Automatizar

Quando se trata de automação de testes de software, precisamos entender o que é automação de testes, o por que automatizar e quando automatizar, respondidas essas perguntas iniciais vem outra pergunta fundamental sobre quais são insumos necessários para automatizar testes, podemos dizer que para fazer uma automação é necessário que haja uma execução de um processo de forma automática, este processo por sua vez pode ser feito de diversas formas como através de:

- Software que grava a tela e executa os testes com base no que foi gravado
  - Ex: Selenium IDE
- Códigos de programação que fazem o papel de simular o uso do usuário, neste caso é necessário uma linguagem de programação + framework
  - Ex: Java + Selenium WebDriver, JavaScript + Cypress e Python + Robot

Simplificando o que é framework



Framework é como se fossem peças prontas que podem ser inseridas em um carro. Essas peças apresentam uma função específica e só funcionam dentro do contexto inteiro, por isso ajudam quando o motorista precisa economizar o dinheiro do conserto de alguma peça defeituosa.

O conceito é semelhante ao de biblioteca — códigos prontos para serem aplicados. No entanto, os frameworks podem ser compreendidos como uma série de bibliotecas, ou seja, uma estrutura ainda maior e mais robusta que permite configurar partes maiores do código.

Ferramentas para automação de testes

Linguagens mais usadas

Ferramentas para automação de testes por contexto

Web:

Mobile:

API:

O que vamos aprender neste curso ?

Neste curso vamos aprender automação de testes com a linguagem de programação Java + os frameworks Cucumber + Selenium WebDriver + JUnit, esse conjunto de ferramentas citado é um dos mais usados quando se fala em automação de testes, esse conjunto oferece uma solução completa e robusta

O que vamos aprender a seguir

1. Java……
2. Cucumber……
3. Selenium……
4. JUnit……

O que é Java ?

Java, situada entre Sumatra e Báli, é uma ilha repleta de vulcões no centro geográfico e econômico da Indonésia. Abriga mais de metade da população do país. A maior cidade de Java é a moderna Jacarta, capital da Indonésia. É onde ficam o grande Museu Nacional, um bairro antigo (Kota Tua) com prédios coloniais holandeses, além de shopping centers e hotéis de luxo. — Google

O que é Java ? (Linguagem de programação)

Java é uma linguagem de programação e plataforma computacional lançada pela primeira vez pela Sun Microsystems em 1995. Existem muitas aplicações e sites que não funcionarão, a menos que você tenha o Java instalado, e mais desses são criados todos os dias. O Java é rápido, seguro e confiável. De laptops a datacenters, consoles de games a supercomputadores científicos, telefones celulares à Internet, o Java está em todos os lugares! Fonte

Qual a diferença entre JDK, JRE e JVM, fonte

Uma grande confusão que paira sobre quem está começando a aprender sobre o mundo Java é a diferença entre JDK, JRE e JVM. Neste post você vai entender, de uma vez por todas, a diferença entre essas três siglas e porque é importante saber para que serve cada um.

A primeira coisa que você precisa saber é que o Java é muito conhecido por trazer o conceito de multi-plataforma. Na verdade esse é o motivo do grande sucesso do Java a mais de vinte anos! O famoso WORA (Write once, run anywhere.), ou em bom português: "Escreva uma vez, execute em qualquer lugar".

Na prática, só de entender esse conceito e como o Java faz para tornar possível escrever um único código e executá-lo em qualquer sistema operacional, você já vai perceber, por alto, a diferença entre JDK, JRE e JVM e onde cada um se encaixa.

Fluxo de execução do Java

O fluxo é basicamente o seguinte:

1. Você escreve o seu código-fonte (arquivo com a extensão .java).
2. Você utiliza o JDK para compilar o seu código-fonte e gerar o arquivo bytecode (arquivo com a extensão .class).
3. Para executar o seu programa, a JVM lê o seu arquivo compilado (.class) e as bibliotecas padrões do Java que estão no JRE.
4. Pronto, seu programa está rodando e todo mundo está feliz! :)

Então, a grosso modo, já deu pra perceber para quê serve cada um:

- **JDK** (Java Development Kit) - é o Kit de Desenvolvimento Java responsável por compilar código-fonte (.java) em bytecode (.class)
- **JVM** (Java Virtual Machine) - é a Máquina Virtual do Java responsável por executar o bytecode (.class)
- **JRE** (Java Runtime Environment) - Ambiente de Execução do Java que fornece as bibliotecas padrões do Java para o JDK compilar o seu código e para a JVM executar o seu programa.

JVM - Máquina Virtual do Java

Já vimos que a JVM é a responsável por executar os programas no formato bytecode, certo? Mas a JVM não é só isso, ela é na verdade o coração do Java, responsável por fornecer a tal capacidade de multi-plataforma.

Isso porque depois que você instala a JVM no sistema operacional (Windows, Linux, Mac, etc.), Ela é capaz de interpretar e executar o programa compilado em formato bytecode em qualquer um desses SOs.

Um coisa legal de perceber aqui é que a JVM executa um programa bytecode, mesmo que este programa bytecode não tenha sido escrito em Java necessariamente.

Isso mesmo! Você não precisa programar na linguagem java, para executar um programa na JVM. Você pode criar seus programas em outras linguagens que também geram arquivos bytecode. Existem várias delas, alguns exemplos são: Scala, JRuby, Jython, Clojure, Groovy, etc..

Outro ponto importante é que a JVM é, na verdade, uma especificação. Isso significa que uma JVM pode ser desenvolvida por qualquer organização, desde que sigam as especificações para a Java Virtual Machine.

JRE - Ambiente de Execução Java

O JRE provê os requisitos mínimos para executar um programa java. Ele contém uma JVM, os pacotes básicos do Java (API core), por exemplo o pacote lang que tem a classe String.

Por fim o JRE também provê ferramentas para executar os programas java. Uma delas é o executável java.exe, que é utilizado para executar uma classe java que contém um método main(String args[]).

Agora você já sabe que quando você executa o comando java SuaClasse, é o JRE que provê esse programa java.exe que você está usando.

JDK - Kit de Desenvolvimento Java

O JDK é Kit que provê ferramentas para o desenvolvimento de programas Java. Ou seja, ele contém um compilador, um depurador e o próprio JRE para você executar os seus programas.

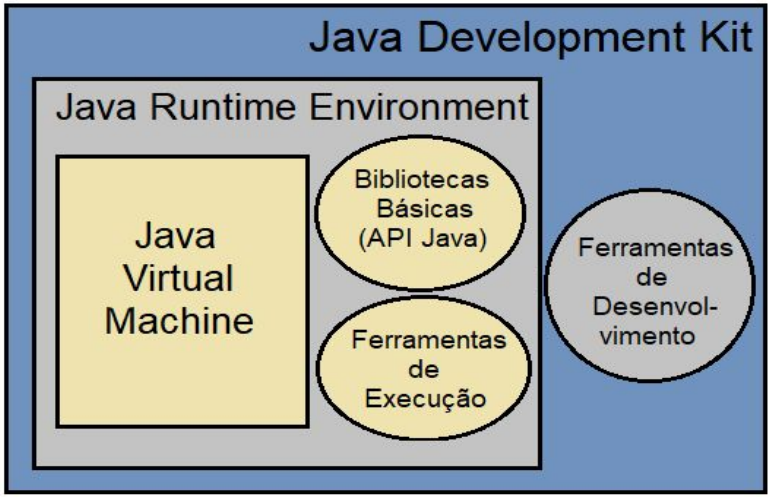
É o JDK que nos provê o programa javac.exe, que compila códigos .java em bytecodes.class.

Agora você já sabe q eu quando você executa o comando javac SuaClasse.java, é o JDK que provê esse programa javac.exe que você está usando.

Uma visão geral

Agora que você já sabe um pouco mais sobre JDK, JRE e JVM, você já deve ter percebido que a JDK traz também um JRE, e o JRE, por sua vez, traz também uma implementação da JVM.

A imagem abaixo ilustra a relação entre esses componentes.



Vamos instalar

Tutorial

O que é algoritmo ?

O que é Algoritmo?

Embora as vezes não percebemos, utilizamos algoritmos no nosso dia-a-dia e não sabemos. Para a execução de alguma tarefa ou mesmo resolver algum problema, muitas vezes inconscientemente executamos algoritmos. Mas o que é Algoritmo?

Algoritmo é simplesmente uma "receita" para executarmos uma tarefa ou resolver algum problema. E como toda receita, um algoritmo também deve ser finito. Se seguirmos uma receita de bolo corretamente, conseguiremos fazer o bolo. A computação utiliza muito esse recurso, então se você pretende aprender programação, obviamente deve saber o que é algoritmo.

Exemplo de algoritmo

Imagine o trabalho de um recepcionista de cinema, ele deve conferir os bilhetes e direcionar o cliente para a sala correta. Além disso, se o cliente estiver 30 minutos adiantado o recepcionista deve informar que a sala do filme ainda não está aberta. E quando o cliente estiver 30 minutos atrasado o recepcionista deve informar que a entrada não é mais permitida

Obs: Essas regras não são 100% verdade, foram definidas apenas para fins didáticos

Vamos escrever um algoritmo para descrever a atividade do recepcionista.

Algoritmo Recepcionista de Cinema

Início

1 - Solicitar ao cliente o bilhete do filme.

2 - Conferir a data e o horário do filme no bilhete.

Se data/hora atual > data/hora do filme + 30 minutos Então

3 - Informar ao cliente que o tempo limite para entrada foi excedido.

4 - Não permitir a entrada.

Senão Se data/hora atual < data/hora do filme - 30 minutos Então

5 - Informar ao cliente que a sala do filme ainda não foi liberada para entrada.

6 - Não permitir a entrada.

Senão

7 - Permitir a entrada.

8 - Indicar ao cliente onde fica a sala do filme.

Fim-Se

Fim

O que são variáveis ?

O que é variáveis ?

Programas de computador utilizam os recursos de hardware mais básicos para executar algoritmos. Enquanto o processador executa os cálculos, a memória é responsável por armazenar dados e servi-los ao processador. O recurso utilizado nos programas para escrever e ler dados da memória do computador é conhecido como variável, que é simplesmente um espaço na memória o qual reservamos e damos um nome. Por exemplo, podemos criar uma variável chamada "idade" para armazenar a idade de uma pessoa. Você pode imaginar uma variável como uma gaveta "etiquetada" em um armário.


Gaveta 1
Gaveta 2
Gaveta 3

O que são condicionais ?

O que são condicionais ?

Estrutura de seleção (expressão condicional ou ainda construção condicional) é, na ciência da computação, uma estrutura de desvio do fluxo de controle presente em linguagens de programação que realiza diferentes computações ou ações dependendo se a seleção (ou condição) é verdadeira ou falsa, em que a expressão é processada e transformada em um valor booleano. Nas linguagens de programação, usamos as palavras em inglês para expressar uma estrutura de seleção, como if, else if e else.

O que são estruturas de repetição ?

O que são estruturas de repetição ?

Dentro da lógica de programação é uma estrutura que permite executar mais de uma vez o mesmo comando ou conjunto de comandos, de acordo com uma condição ou com um contador.

São utilizadas, por exemplo, para repetir ações semelhantes que são executadas para todos os elementos de uma lista de dados, ou simplesmente para repetir um mesmo processamento até que a condição seja satisfeita.

Existem 4 estruturas de repetição básica para praticamente todas as linguagens de programação, seja Java ou javascript.

1. While (enquanto)
2. Do… While (faça enquanto)
3. For (para)
4. Foreach

O que são objetos ?

O que é objeto ?

Um objeto é um elemento computacional que representa, no domínio da solução, alguma entidade (abstrata ou concreta) do domínio de interesse do problema sob análise. Objetos similares são agrupados em classes.

No paradigma de orientação a objetos, tudo pode ser potencialmente representado como um objeto. Sob o ponto de vista da programação, um objeto não é muito diferente de uma variável no paradigma de programação convencional. Por exemplo, quando define-se uma variável do tipo int em C ou em Java, essa variável tem:

um espaço em memória para registrar o seu estado atual (um valor);
um conjunto de operações associadas que podem ser aplicadas a ela, através dos operadores definidos na linguagem que podem ser aplicados a valores inteiros (soma, subtração, inversão de sinal, multiplicação, divisão inteira, resto da divisão inteira, incremento, decremento).
Da mesma forma, quando se cria um objeto, esse objeto adquire um espaço em memória para armazenar seu estado (os valores de seu conjunto de atributos, definidos pela classe) e um conjunto de operações que podem ser aplicadas ao objeto (o conjunto de métodos definidos pela classe).

Um programa orientado a objetos é composto por um conjunto de objetos que interagem através de "trocas de mensagens". Na prática, essa troca de mensagem traduz-se na invocação de métodos entre objetos.

Algoritmo do site adulto

Exemplo