INSTITUTO FEDERAL
Santa Catarina
Câmpus Gaspar

Teste de Software

Prof. Marcos Rodrigo momo, M.Sc.
marcos.momo@ifsc.edu.br

Gaspar, junho 2021.

Roteiro da aula 9

- Gravação
- Aspectos gerais sobre teste
- Ferramentas para testes
  - JUnit / PHPUnit – teste de unidade
  - Selenium – teste funcional
  - EasyAccept – teste de aceitação
  - FindBugs – teste estático

Automação de testes

- Script de Teste
  - Arquivo que compõe os passos do caso de teste a serem executados sobre o sistema.
- Gravador ou "Recorder"
  - Grava na forma de uma linguagem própria passos da execução dos testes manuais.
- Executor de teste ou "Playback"
  - Recurso das ferramentas para re-executar tudo o que foi gravado no script de teste.

Automação de testes

Crie os casos de teste para depois automatizar

Não pense em automatizar tudo

Automatize o que é prioridade

Ferramentas têm que se adaptar às novas interfaces

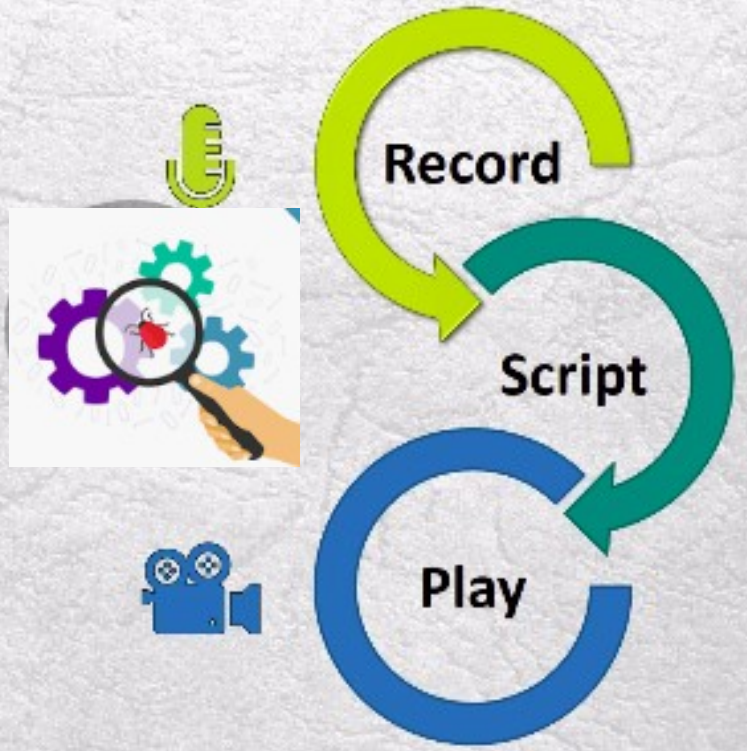
Automação requer experiência na ferramenta

Inicie a automação o quanto antes no projet

Automação em Teste de Software

Automatizar: Técnica de conceder à um dispositivo, a autonomia de controlar o próprio funcionamento, com a mínima interferência humana.

Em Teste de Software, consiste no uso de uma ferramenta que imita a interação do testador para com a aplicação.

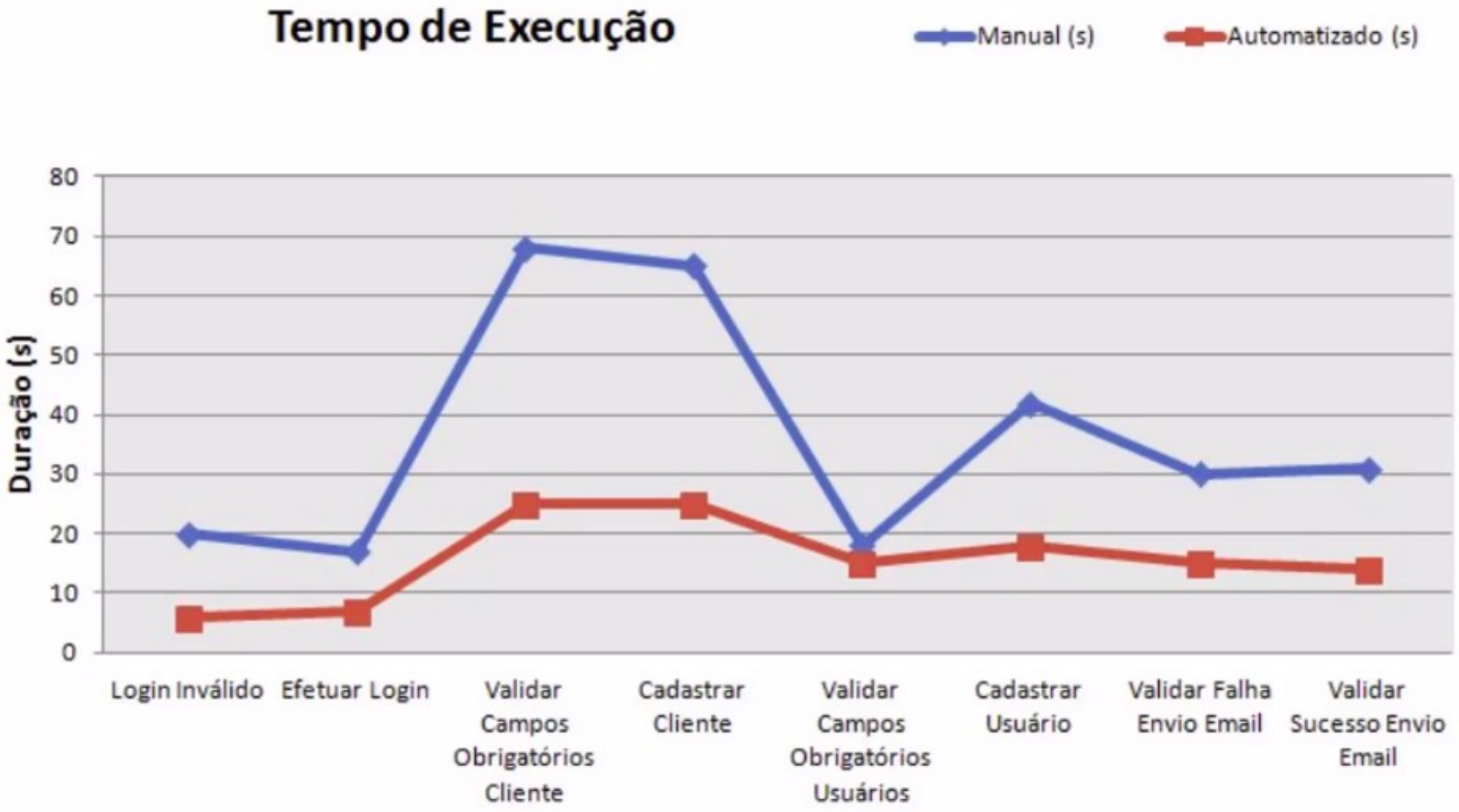


Por que Automatizar os Testes?

Visão Gerencial: As variáveis do Triângulo das Restrições (escopo, tempo e custo) se tornam mais independentes uma da outra sem que isso afete na qualidade do produto a ser entregue.

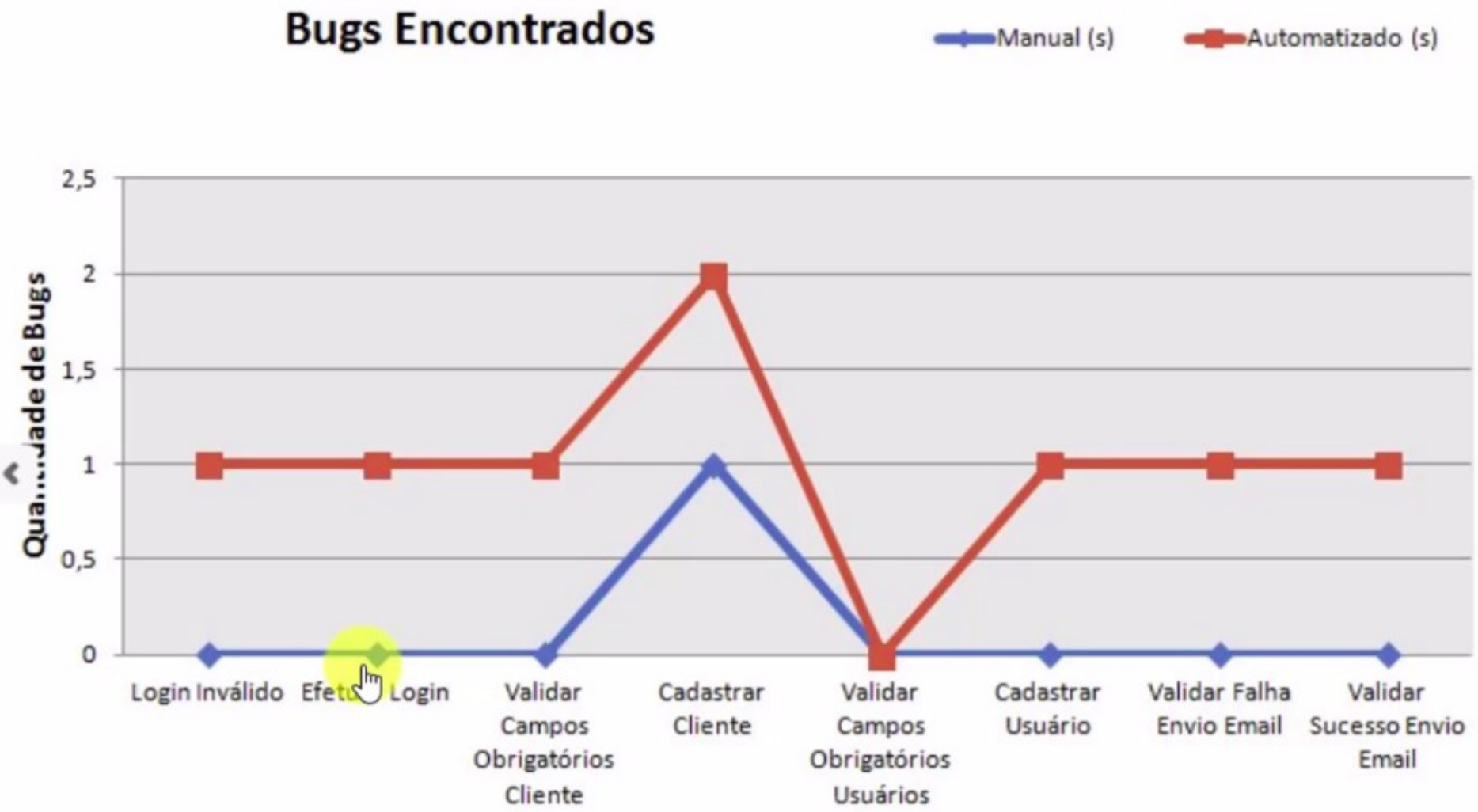
Visão Operacional:

- ✓ Execução dos testes com a mesma eficiência, porém em menos tempo.
- ✓ Redução drástica de falhas humanas oriundas de tarefas repetitivas (Teste de Regressão).
- ✓ Maior disponibilidade dos testadores, que poderão focar mais em atividades estratégicas.



Fonte: https://inoveteste.com.br/

3/4



Fonte: https://inoveteste.com.br/

Testes Automatizados x Testes Manuais

O teste manual não pode ser eliminado, mas sim reduzido ao máximo e focado em casos específicos onde seja muito caro automatizar ou que seja necessário de uma decisão humana.

Mesmo que a empresa automatize 99% dos testes, sempre haverá necessidade de pessoas para realizarem tarefas inviáveis aos robôs. **Ex**: Manutenção no projeto de teste, pesquisa e documentação de novas estratégias de teste, etc.

O robô jamais irá substituir o testador!!!

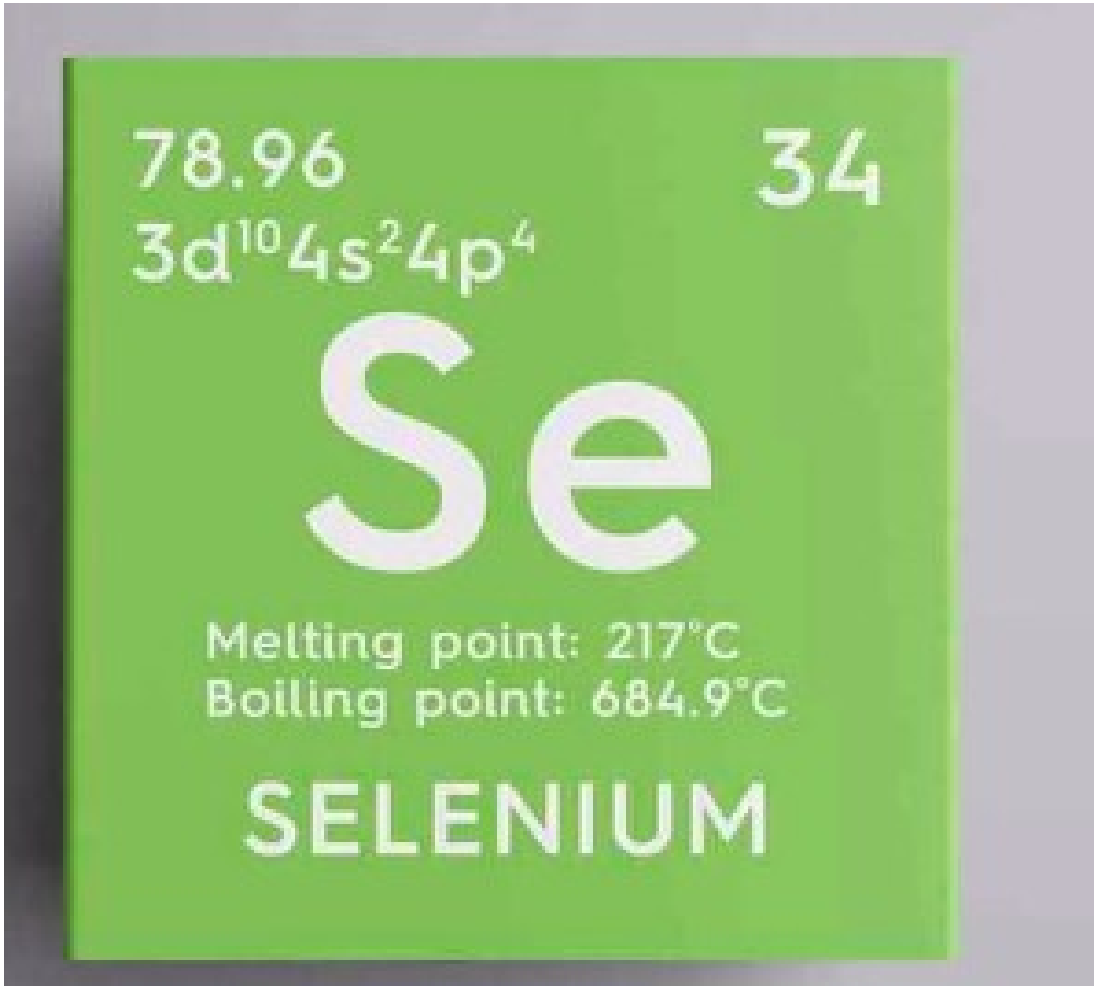
Considerações acerca da Automação

5W2H		
5W	What (O que?)	Saber quais sistemas e respectivos cenários de testes deverão ser automatizados.
	Who (Quem?)	Identificar quais integrantes da equipe possuem o perfil adequado para a atividade.
	Where (Onde?)	Decidir se será num ambiente a parte ou compartilhado com o desenvolvimento.
	When (Quando?)	Saber qual o momento certo para começar a automatizar.
	Why (Por que?)	Descobrir os reais motivos do porquê é necessário implementar a automação nos testes.
2H	How (Como?)	Definir técnicas e ferramentas que são utilizadas no processo de automação.
	How Much (Quanto custa?)	Calcular os custos para automatizar e verificar se compensa aos resultados propostos.

Princípios da Automação de Testes

- ✓ Projete os casos de testes para depois automatizar;
- ✓ Não pense em automatizar 100% de tudo. Foque na prioridade;
- ✓ Projete a automação de testes de forma que facilite revisões;
- ✓ Inicie a automação de testes o mais cedo possível em um projeto;

Introdução ao Selenium



Fonte:https://www.selenium.dev

Selenium

- Traduzido do inglês-Selenium é uma estrutura portátil para testar aplicativos da web
- O Selenium fornece uma ferramenta de reprodução para criar testes funcionais sem a necessidade de aprender uma linguagem de *script* de teste.

O que é Selenium e como surgiu?

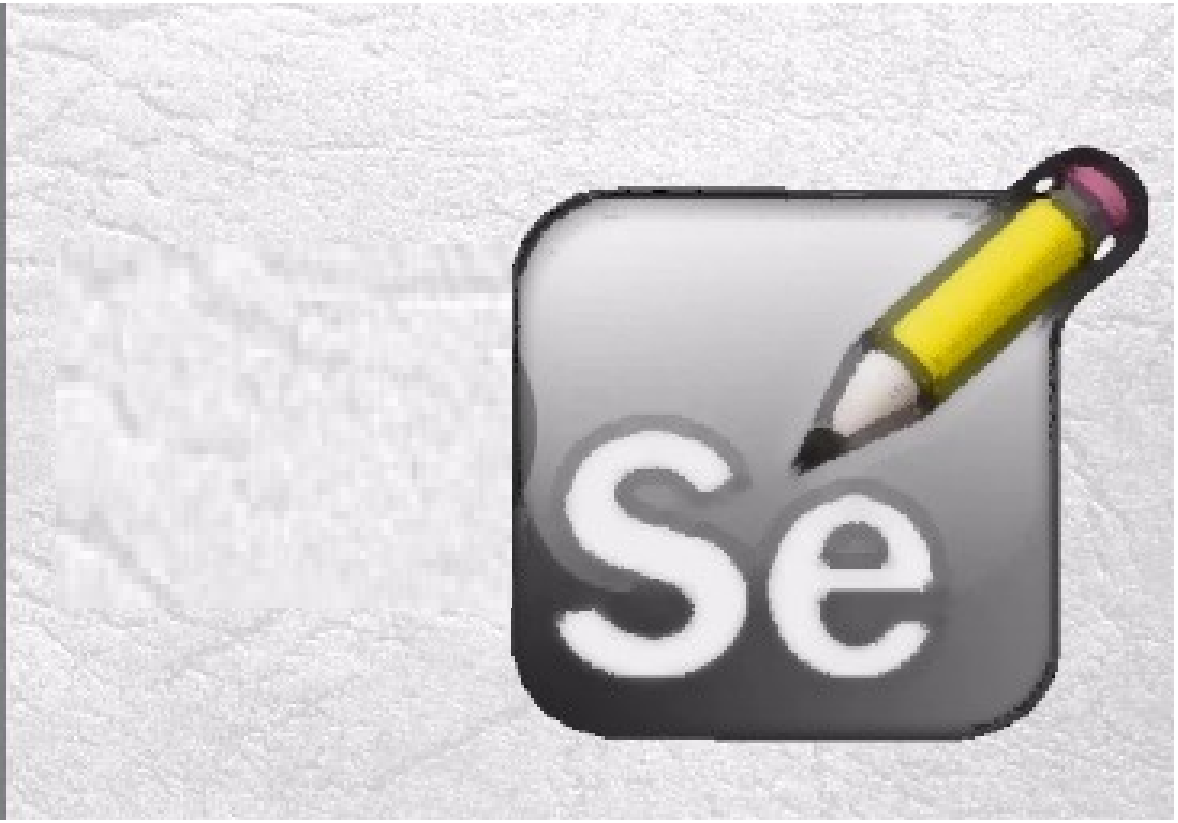
Selenium é uma suíte composta por 4 ferramentas de automação de testes para aplicações web. E são elas: **IDE**, **Remote Control**, **WebDriver** e **GRID**.

Em **2004**, o testador **Jason Huggins** estava testando uma aplicação interna da **ThoughtWorks**, quando percebeu que poderia gerenciar melhor seu tempo nas atividades de teste manuais. Para isso, ele criou uma biblioteca Javascript que interagia com o browser. A esse projeto foi lhe concedido o nome de **Selenium**, passando a ser uma ferramenta teste alternativa (open source) para testadores que não suportavam mais ficar dependentes da ferramenta **Quick Test**, da empresa **Mercury**, que por sinal era paga e bem cara. Na química, **o antídoto do Mercúrio é o Selênio**. Está aí, o porquê do nome.

Por que usar Selenium?

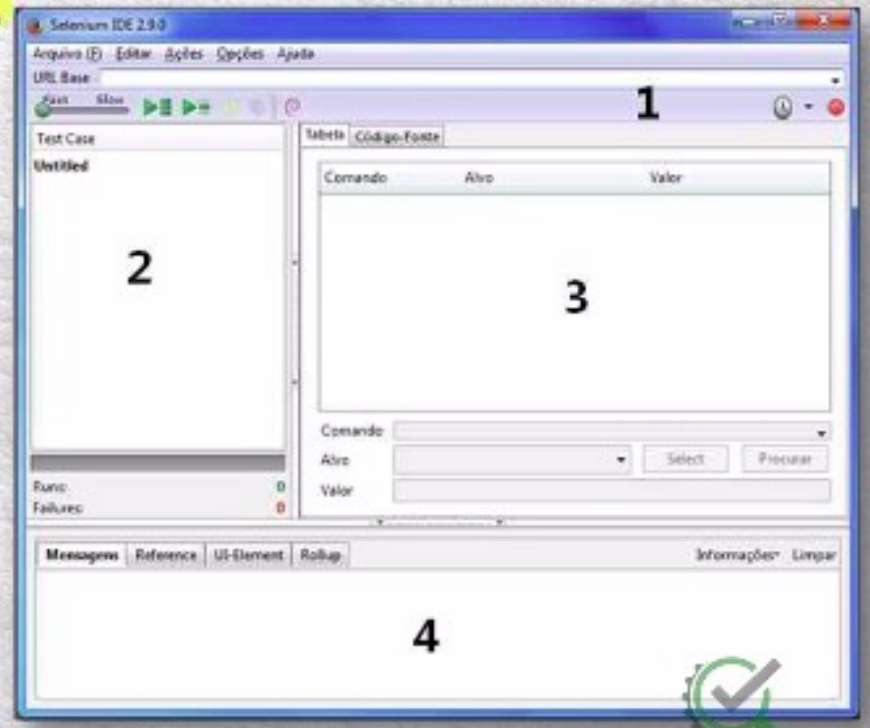
- Criar e executar scripts de testes independente do browser ou sistema operacional;
- Realizar Testes de Carga/Estresse através da execução de teste em paralelo;
- Adicionar plug-ins que permitem elaborar scripts de testes robustos e que atendem as necessidades dos negócios;
- Integrar os scripts de teste a um projeto de teste, seja em Java, C#, PHP, Python ou Ruby.

Selenium IDE



IDE

1. **Barra de Ferramentas:** Possui as funcionalidades que gerenciam os testes como gravar, executar, pausar, etc.
2. **Lista de Casos de Teste:** Lista dos casos de teste que compõem a suíte de teste em uso.
3. **Editor de Script:** Espaço para editar o script do caso de teste selecionado, podendo definir o step em que o teste irá iniciar ou parar.
4. **Rodapé:** Log da execução do caso de teste selecionado.



Fazer um teste funcional com Selenium https://www.selenium.dev/

- Instalar o plugin do navegador
  - https://www.youtube.com/watch?v=DomPPe_fe-o
  - Exemplo de uso para buscar um CEP
- Criar um script de teste automatizado
- Executar o script de teste

WebDriver

- Selenium Webdriver é uma ferramenta que oferece uma API que permite a escrita de forma mais produtiva e organizada de scripts de testes com Java.

- O Selenium WebDriver faz chamadas diretamente ao navegador utilizando o suporte à automação nativo de cada navegador

- Assim os testes escritos com o WebDriver são bastante realistas, pois chama diretamente o navegador

- Além disso, o Selenium Webdriver suporta praticamente todos os navegadores web existentes: Google Chrome, Firefox, Internet Explorer, Safari, Opera, etc.

Biblioteca Selenium

- Criar projeto
- Baixar a biblioteca do Selenium
  - A biblioteca do Selenium possibilita a aplicação comunicar com o navegador
  - Oferece uma conjunto de métodos para interação com páginas web
    - Por exemplo: abrir uma página, escrever em um campo de busca, clicar em um botão etc…
- Baixar o Chromedriver
  - Uma instância para execução no navegador do chrome
- Baixar e incorporar ao projeto
  - Adicionar a biblioteca – build Path
  - Criar um projeto Java com Maven
    - Pom.xml é o arquivo que contém a configuração do projeto e suas dependências

INSTITUTO FEDERAL
Santa Catarina
Câmpus Gaspar

For Firefox

```
System.setProperty("WebDriver.gecko.driver","C:\\Users\\shalini\\Downloads
WebDriver driver = new FirefoxDriver();
```

For Chrome

```
System.setProperty("WebDriver.chrome.driver","C:\\Users\\shalini\\Download
WebDriver driver = new ChromeDriver();
```

For Opera

```
System.setProperty(“WebDriver.opera.driver”,”C:\\Users\\shalini\\Downloads
WebDriver driver = new OperaDriver();
```

For Internet Explorer

```
System.setProperty("WebDriver.ie.driver", " C:\\Users\\shalini\\Downloads\
WebDriver driver=new InternetExplorerDriver();
```

EasyAccept

- O EasyAccept é uma ferramenta que ajuda a criar e executar testes de aceitação de maneira fácil, rápida e limpa, preenchendo a lacuna de comunicação entre clientes e desenvolvedores de software.

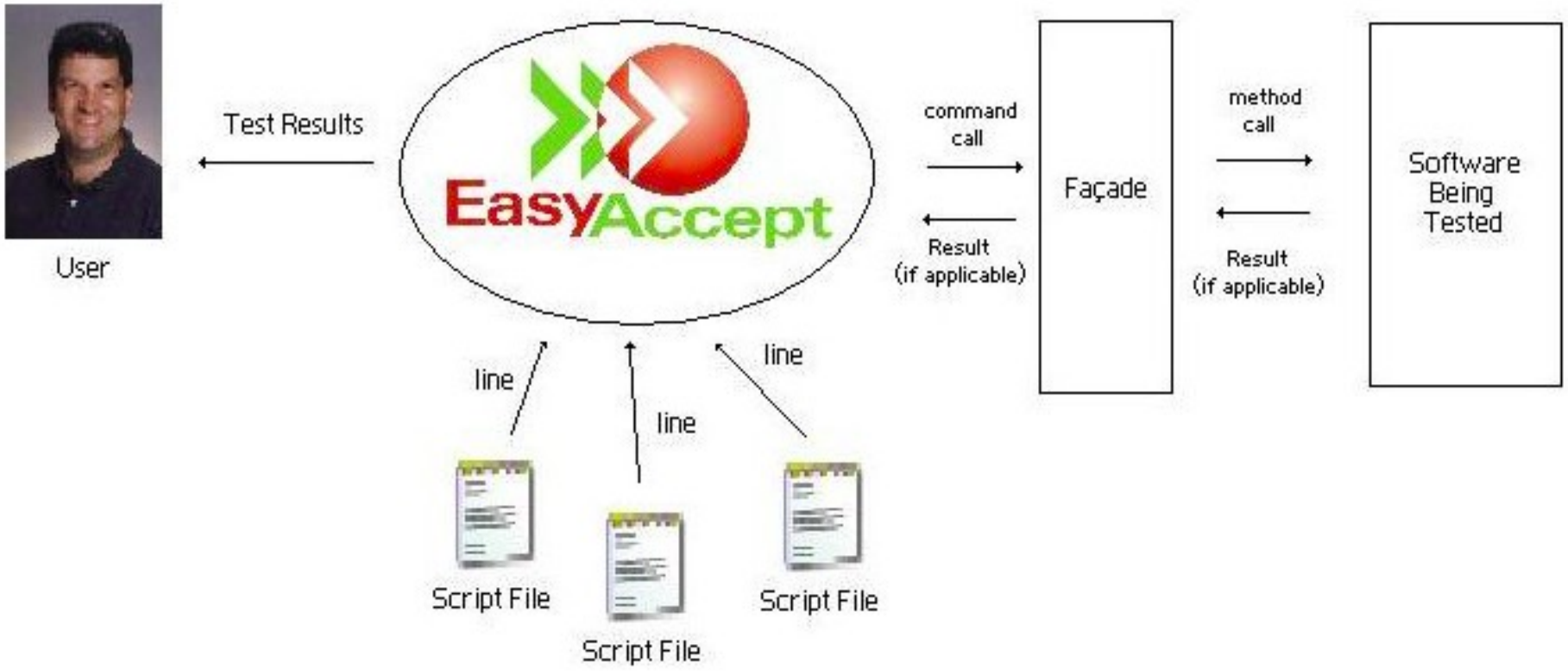
- Você pode fazer o download da versão mais recente e começar a usá-la imediatamente, mas os visitantes iniciantes são incentivados a ler este breve tutorial antes de se familiarizar com o uso. Um manual também está disponível se você desejar uma descrição detalhada de sua operação.

- O EasyAccept está sendo desenvolvido em um projeto de código aberto sob licença GNU, realizado no SourceForge. Sinta-se livre para participar do projeto e nos ajudar a melhorar o produto. Uma lista de histórias futuras planejadas de usuário está disponível aqui, onde você também pode sugerir novos recursos.

EasyAccept

- EasyAccept é uma ferramenta que ajuda a criar testes de aceitação
- Possibilita que clientes e os desenvolvedores de software possam se comunicar de forma mais clara.
- É uma ferramenta OpenSource
- O EasyAccept pode ser utilizado para testar sistemas em Java ou em Python.

Como funciona o EasyAccept



http://easyaccept.sourceforge.net/

Passo 1/3

- Os clientes (ou quem está ajudando os clientes) escrevem testes de aceitação em arquivos de texto simples usando o editor que escolherem.

Passo-a-passo 2/3

- Os desenvolvedores gravam uma única Fachada no programa que está sendo testado que contém métodos correspondentes aos comandos de script usados nos testes de aceitação. A fachada pode até já existir, usada para outros fins.

Passo-a-passo 3/3

- O EasyAccept pega um ou mais arquivos txt de script e o Facade, executa os scripts pelos métodos do Facade e compara os resultados esperados com os resultados reais produzidos pelo programa. As divergências são prontamente indicadas; caso contrário, é exibida uma mensagem informando que todas as execuções foram aprovadas.

FindBugs

- O FindBugs é uma ferramenta de código aberto utilizado pelos desenvolvedores de software para fazer uma auditoria ou inspeção no código de forma automatizada
- Esta ferramenta examina as sua classes procurando por possíveis problemas no código durante a fase de desenvolvimento
- O FindBugs analisa o código fonte ou mesmo o código objeto (bytecode para programas Java) do programa procurando por padrões conhecidos.

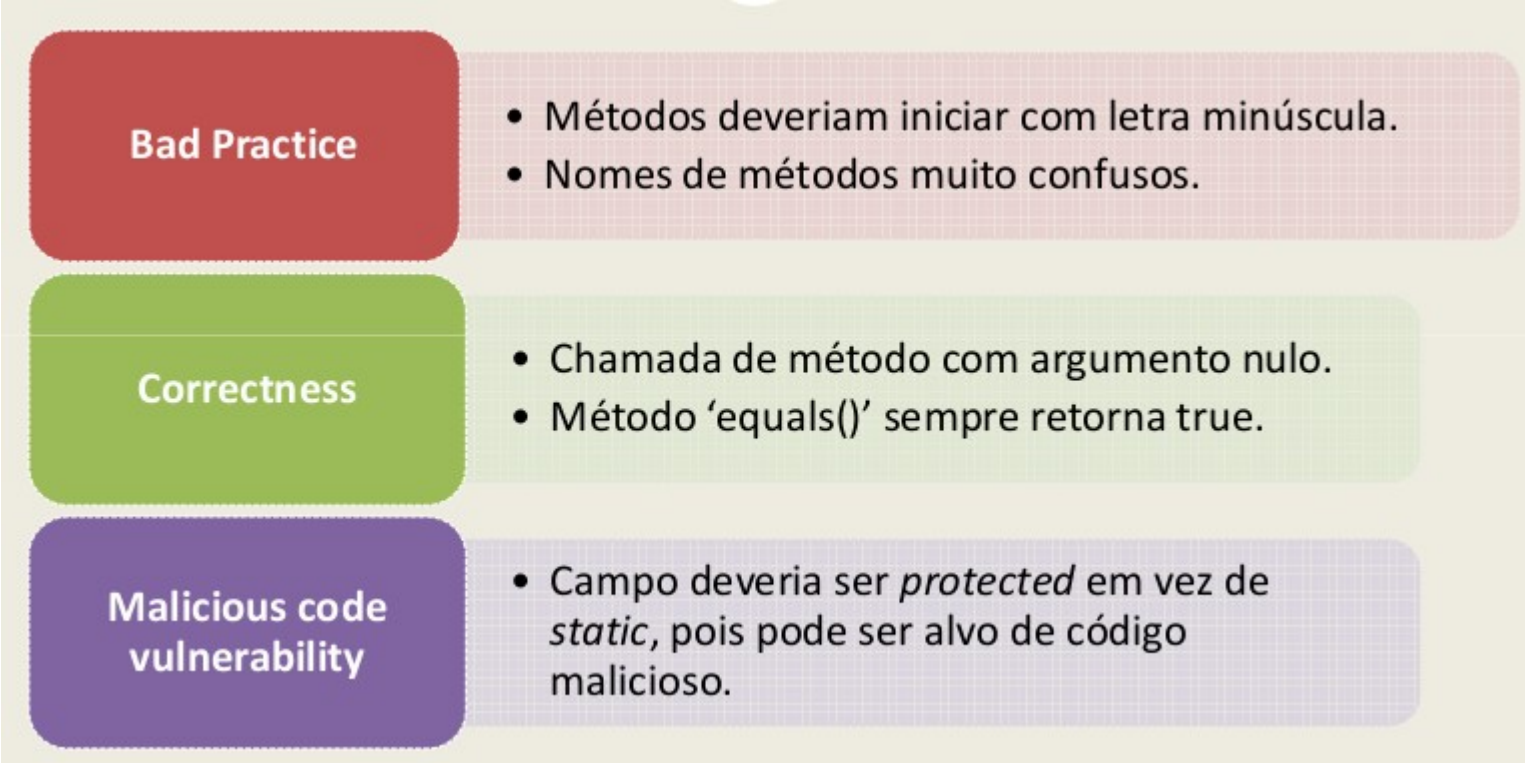
FindBugs

- O FindBugs é criterioso e apresenta um relatório bem interessante. Entre as informações interessantes que o Find Bugs pode informar tem-se:
  - Acesso a referências null
  - Uso inapropriado de APIs
  - Overflow em vetores
  - Divisão por zero
  - Campos estáticos não declarados como final
  - Uma classe que sobrescreve o método equals(Object) mas não sobrescreve o método hashCode()
  - Um campo estático e final que referencia um vetor mutável
  - Entre muitos outros.

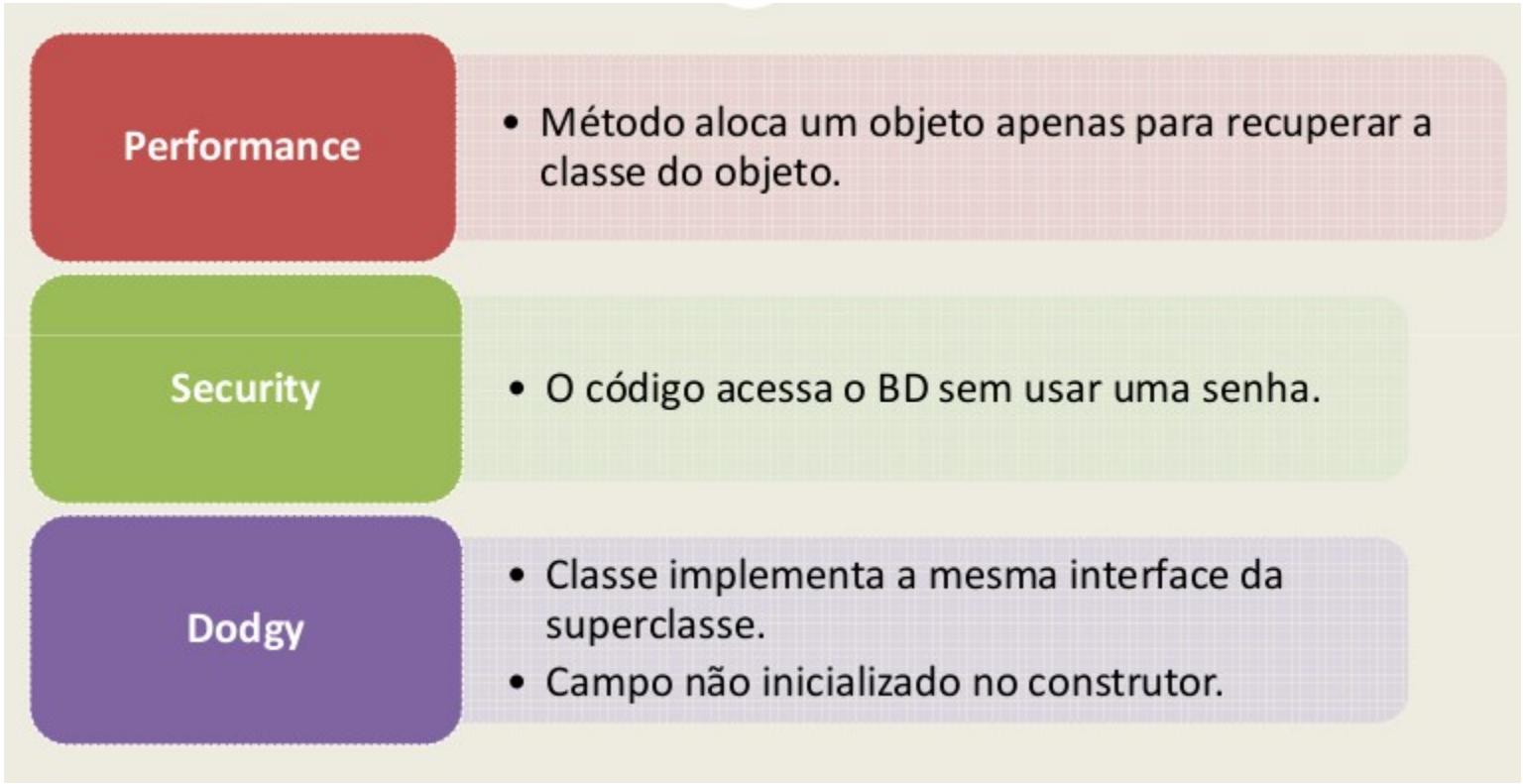
FindBugs

- Atualmente a ferramenta FindBugs provê suporte a mais de 250 padrões de erros sendo mais de uma centena deles classificados como erros de correção
- Além disso, permite que programadores escrevam seus próprios padrões de erros
- Com o relatório exposto pelo FindBugs é possível que os desenvolvedor, arquitetos e líderes de projetos avaliem melhor o código, o projeto e inclusive a equipe sob uma outra perspectiva, além de fornecer um feedback de onde os desenvolvedores estão mais desatentos.

Categoria dos bugs



Categoria dos bugs



Referência

- EasyAccept. Disponível em: <http://easyaccept.sourceforge.net/>
- Análise Estática de Código com FindBugs. Disponível em: <https://www.devmedia.com.br/analise-estatica-de-codigo-com-findbugs/25916>.
- FindBugs. Disponível em: http://findbugs.sourceforge.net/bugDescriptions.html
- Introdução ao FindBugs. Instalando e Usando a ferramenta. Disponível em:<https://www.youtube.com/watch?v=oFk5C2TffQM>
- Selenium. Disponível em: https://www.selenium.dev/
- IoveTeste. Disponível em: https://inoveteste.com.br/
- Selenium. Getting started with WebDriver. Disponível em: <https://www.selenium.dev/documentation/en/>
- Introdução ao Selenium WebDriver. Disponível em:<http://www.testesautomatizados.com.br/introducao-selenium-webdriver/>
- Repositório Maven. https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java/4.0.0-alpha-6