**Tutorial Selenium**

**Teste de Selênio**

**Excel ===================================**

Links do ToolsQA para referência de como implementar a leitura de arquivo Excel:

<https://www.toolsqa.com/selenium-webdriver/excel-in-selenium/>

<https://www.toolsqa.com/selenium-webdriver/data-driven-framework/>

\*Utilize as dependências abaixo para a implementação:

<dependency>  
    <groupId>org.apache.poi</groupId>  
    <artifactId>poi</artifactId>  
    <version>4.1.2</version>  
</dependency>

<dependency>  
    <groupId>org.apache.poi</groupId>  
    <artifactId>poi-ooxml</artifactId>  
    <version>4.1.2</version>  
</dependency>

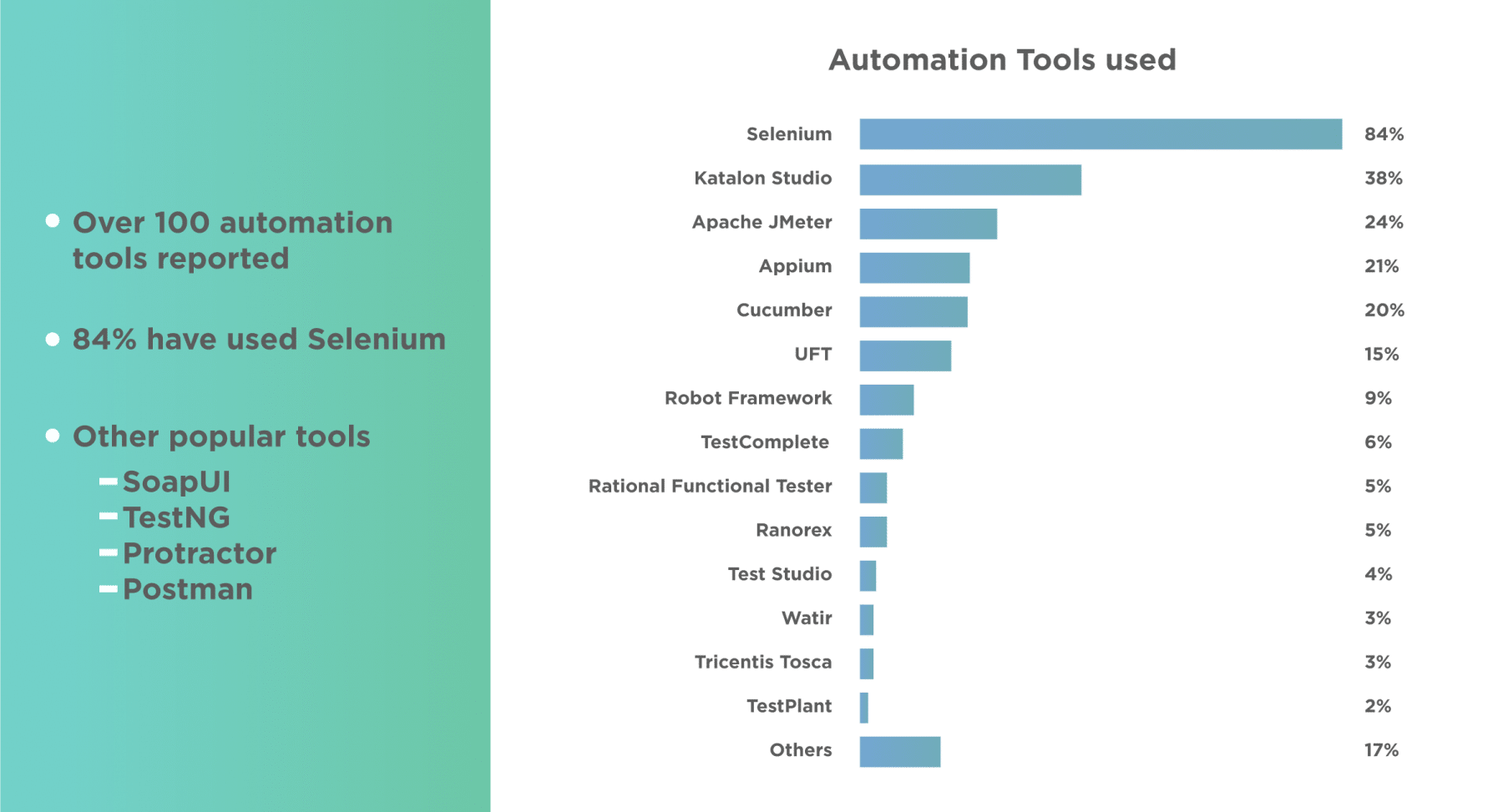
**======================================excel.**

Todos sabemos que o ***Teste de Software*** é parte integrante do SDLC e, às vezes, envolve muitas tarefas repetitivas, combinando cenários repetidamente. Agora, para reduzir esses esforços repetitivos, os testadores de software precisam da ajuda de algumas ferramentas que possam permitir que eles automatizem essas tarefas repetitivas. Um dos grandes conjuntos de ferramentas de automação é o ***Selenium*** que fornece os recursos para automatizar as ações do usuário em um aplicativo da web. É um projeto guarda-chuva de código aberto para uma variedade de ferramentas e bibliotecas que permitem e suportam a automação de navegadores da web. Vamos entender neste artigo todas as complexidades do ***Selenium Automation Testing***, cobrindo os detalhes nos seguintes tópicos:

* O que é Selênio?
* Como se originou o selênio?
* Quais são os vários componentes do selênio?
* Selenium IDE
* Selênio RC
* Selenium WebDriver (Selenium 4)
* Grade de Selênio
* Por que o selênio é popular?
* O que o selênio pode fazer?
* O que o Selenium não pode fazer?
* Além disso, quais são os pré-requisitos para aprender Selenium?
* Qual ferramenta Selenium se encaixa na sua necessidade?

**O que é o teste de selênio?**

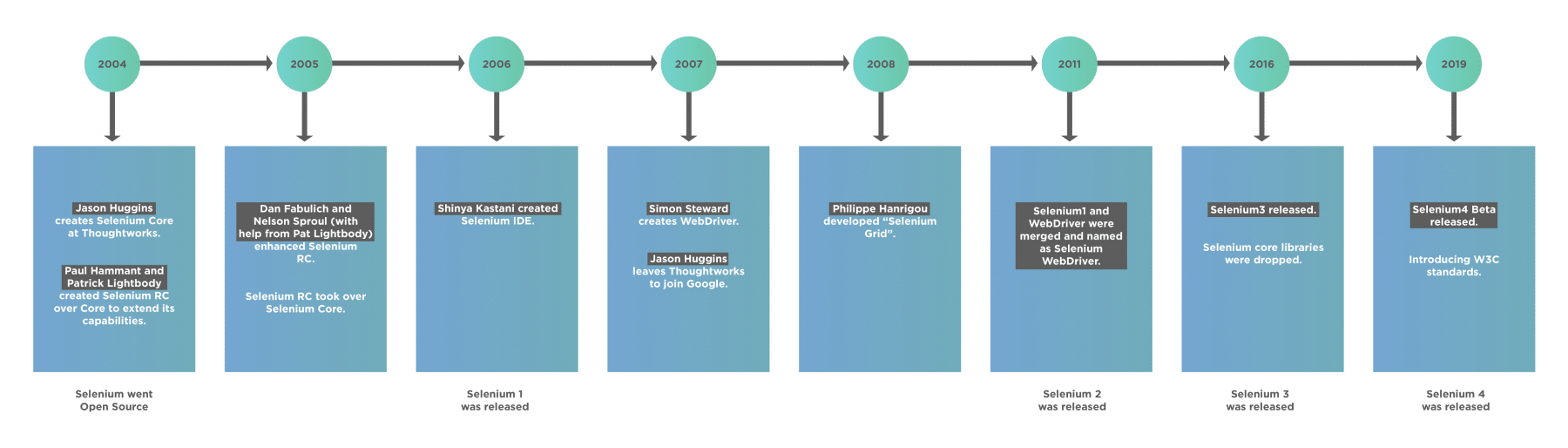
Conforme discutimos, o Selenium é um **conjunto de ferramentas de automação da Web de código aberto** que aproveita o poder dos navegadores da Web e ajuda a automatizar fluxos de trabalho de como os usuários interagem com o aplicativo da Web no navegador. O objetivo principal do Selenium, conforme destacado pelo site do Selenium, é - " O Selenium automatiza os navegadores, o que você faz com esse poder depende inteiramente de você ". Mesmo com as ferramentas e tecnologias emergentes, o Selenium ainda lidera o conselho na lista de ferramentas de automação da Web e testes de automação. A pesquisa abaixo realizada pela Katalon mostra a popularidade do Selenium:



Uma ferramenta tão popular e com um público tão amplo antes de entender os detalhes adicionais do pacote Selenium, vamos saber como o Selenium surgiu e qual foi seu ponto inicial:

**Como se originou o selênio?**

A história do Selenium remonta ao início dos anos 2000. Jason Huggins, engenheiro da ThoughtWorks, criou um módulo JavaScript . Seu nome era JavaScriptTestRunner, para a automação de um site interno. O nome Selênio vem de uma piada feita por Huggins, dizendo que ***" você pode curar o envenenamento por mercúrio tomando suplementos de selênio "***. Mesmo que isso fosse apenas uma piada enquanto zombavam de um concorrente chamado Mercury, outros que receberam o e-mail com a piada, pegaram o nome e correram com ele. A figura a seguir mostra a evolução do Selenium ao longo dos anos:

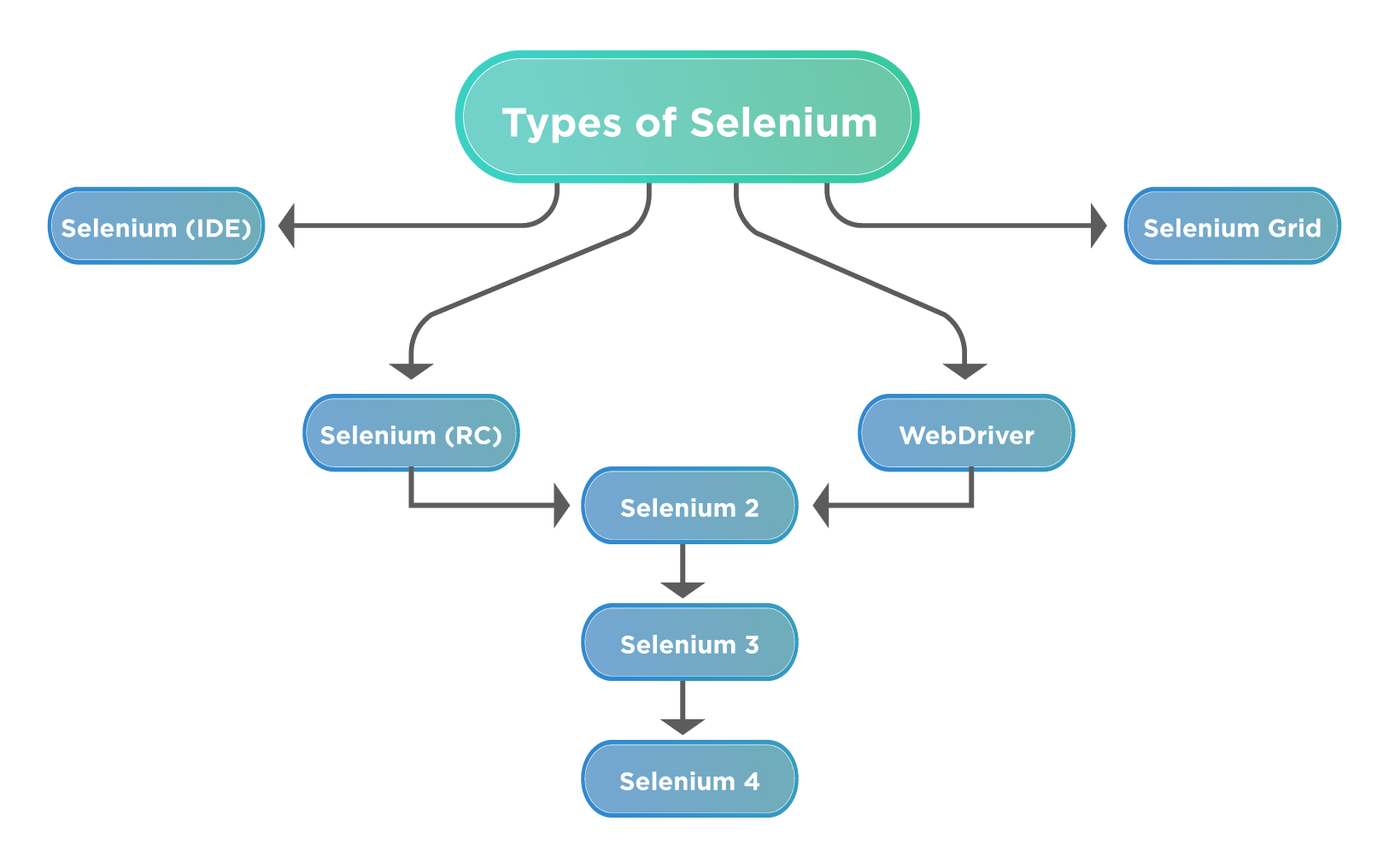


O Selenium4 ainda está em versão beta e o desenvolvimento ainda está em andamento.

**Quais são os vários componentes do selênio?**

Como discutimos acima, o Selenium não é apenas uma ferramenta de automação. **É um conjunto de ferramentas** e cada ferramenta do conjunto possui recursos exclusivos específicos que ajudam no design e desenvolvimento da estrutura de automação. Todos esses componentes podem ser usados individualmente ou podem ser combinados entre si para atingir um nível de automação de teste.

A figura abaixo mostra vários componentes do Selenium Suite:



Vamos entender todos esses componentes com mais detalhes:

**O que é Selenium IDE?**

***O Selenium IDE*** é uma extensão disponível para Firefox e Chrome, que possui a funcionalidade de gravação e reprodução disponível. O Selenium IDE agora também tem a capacidade de exportar o código em vários idiomas predefinidos. Ele também oferece a capacidade de usar um caso de teste dentro de outro.

**O que é Selênio RC?**

O Selenium RC é um servidor que atua como intermediário entre o usuário e o navegador que precisa interagir. RC usa Javascript para trabalhar com navegadores enquanto permite que os usuários escrevam código no idioma de sua escolha. Foi a versão principal do Selenium por algum tempo. O Selenium RC teve problemas com a One Origin Policy e foi **substituído pelo WebDriver**.

**O que é Selenium WebDriver?**

O Selenium WebDriver é o componente mais comumente usado do Selenium. O WebDriver permite que os usuários escrevam códigos personalizados no idioma de sua escolha e interajam com o navegador de sua escolha, por meio de drivers específicos do navegador. O WebDriver funciona no nível do sistema operacional e usa um protocolo chamado ***JSONWireProtocol*** para se comunicar com os navegadores.

Como mostrado acima, o Selenium 2 foi a implementação real da fusão do projeto WebDriver e RC . Os recursos do WebDriver e RC foram incorporados na versão 2 do WebDriver e chamados de Selenium 2.

O Selenium 3 foi uma atualização do Selenium 2 em muitos termos. O Selenium 3 tornou-se um padrão do World Wide Web Consortium (W3C). O Selenium RC mudou para um pacote legado e muitas melhorias e novos recursos foram adicionados para atender ao cenário da web em constante mudança.

Além disso, o Selenium 4 é o novo membro da família (última versão do selenium) e ainda está em fase beta para os usuários finais.

**O que é *[Selenium GRID](https://www.toolsqa.com/selenium-webdriver/selenium-grid/)*?**

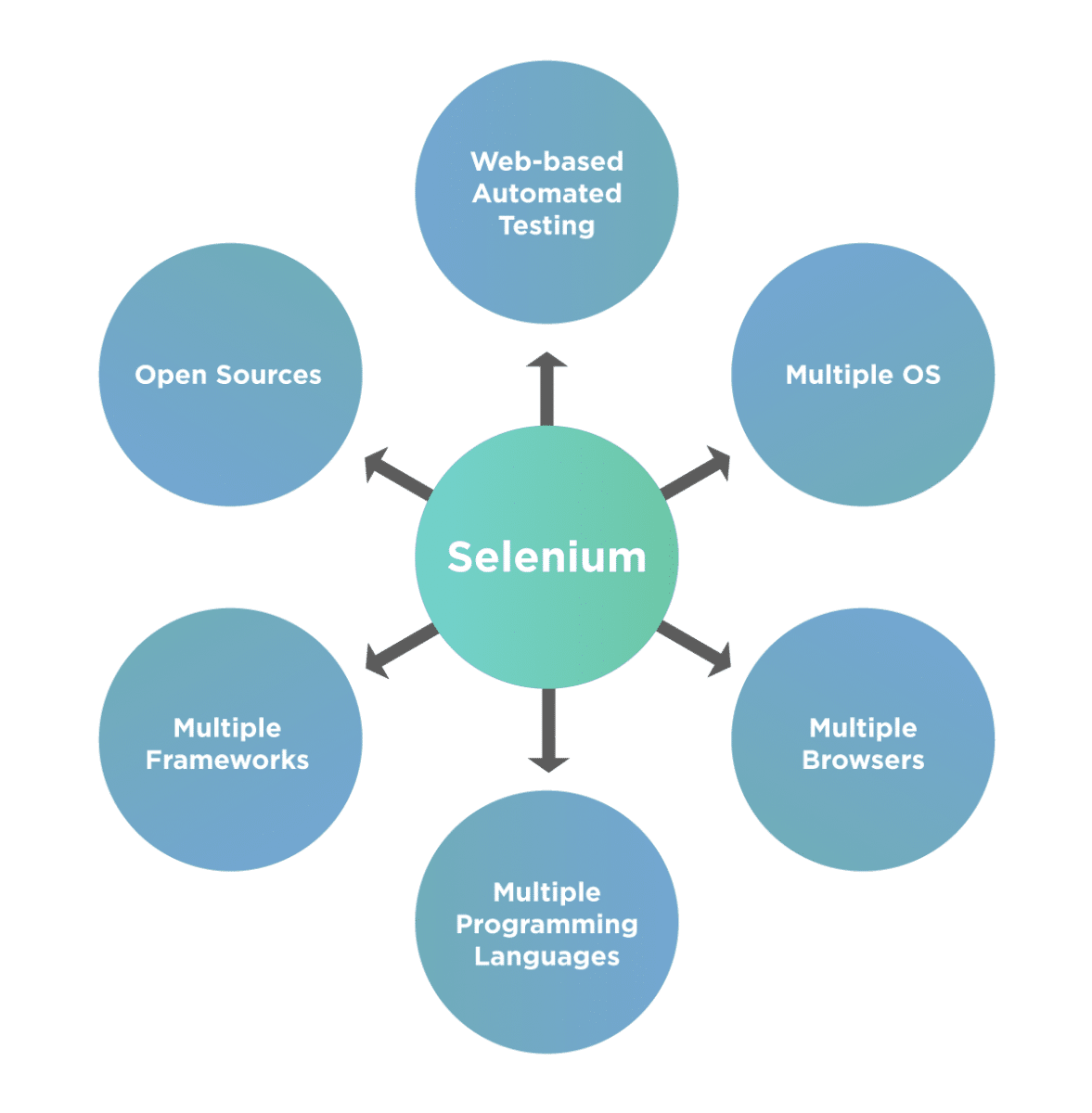
[***O Selenium GRID***](https://www.toolsqa.com/selenium-webdriver/selenium-grid/) permite que os usuários executem testes em **diferentes máquinas**, com diferentes navegadores e sistemas operacionais simultaneamente, o que permite executar testes em paralelo, economizando muito tempo e recursos de teste em várias máquinas.

**Por que o selênio é popular e para que serve o selênio?**

O Selenium se tornou uma das ferramentas mais populares para automatizar aplicativos da Web com uma grande comunidade por trás do suporte. Grandes gigantes multinacionais como Google, Apple, Amazon e muitas outras empresas da Fortune 500 usam o Selenium diariamente e até forneceram suporte para a ferramenta. Os contribuidores originais do Selenium ainda estão contribuindo ativamente de forma contínua, o que permitiu que o Selenium estivesse atualizado com os desafios dos testes modernos de aplicativos da web.

Muitas das principais empresas usam o ***LambdaTest*** para testes entre navegadores junto com o Selenium.

Além disso, conforme discutimos, o ***conjunto de ferramentas Selenium***, que compreende diferentes componentes como Selenium IDE, WebDriver, Grid, etc., está disponível para automação em diferentes navegadores, plataformas e linguagens de programação. O Selenium também oferece suporte a todos os principais sistemas operacionais, como macOS, Windows, Linux, e também oferece suporte a sistemas operacionais móveis, como iOS e Android . A figura abaixo mostra alguns dos recursos padrão atribuídos à popularidade do Selenium:



Fora desse horizonte, vamos entender o que o Selenium é capaz de fazer?

**O que o selênio pode fazer?**

São muitos os recursos que o Selenium oferece e que atribuem por ser uma das ferramentas de automação mais apreciadas do mercado. Vamos dar uma olhada rápida nesses recursos:

**Código aberto** – Ao contrário de ferramentas como o QTP, o Selenium é de código aberto, o que significa que nenhum custo é necessário para configurar e usar o Selenium. ***O Selenium é gratuito para baixar e usar***.

**Imitar ações do usuário** – Quase todas as ações do usuário do mundo real, como clicar no botão, arrastar e soltar seleção, caixas de seleção, pressionamentos de teclas, toques e rolagem, podem ser automatizadas usando o Selenium.

**Implementação Fácil** – O Selenium é famoso por ser amigável. Os usuários podem desenvolver extensões personalizadas para seu uso, pois o código é de código aberto.

**Suporte a idiomas** – O benefício mais significativo do Selenium é o amplo suporte para vários idiomas. O Selenium suporta linguagens de programação máximas como Java, Python, JavaScript, C#, Ruby, Perl, Haskell, Go, entre outras.

**Suporte ao navegador** – O Selenium pode funcionar em todos os fornecedores de navegadores existentes. O Selenium tem suporte para Chrome, Firefox, Edge, Internet Explorer, Safari.

**Suporte do sistema operacional** – as ligações do Selenium estão disponíveis para todos os sistemas operacionais primários, como Linux, macOS, Windows.

**Suporte a framework** – O Selenium suporta vários frameworks como **Maven, TestNG, PYTest, NUnit, Mocha, Jasmine**, etc. O Selenium se integra bem com ferramentas CI como Jenkins, Circle CI, GOCD, Travis CI, Gitlab, etc.

**Reutilização de código** – Scripts escritos para Selenium são compatíveis com vários navegadores. O mesmo código pode ser executado para vários navegadores **usando os respectivos binários do navegador** e em máquinas separadas em uma configuração de grade,

**Suporte da comunidade** – Como há muitos QAs trabalhando nessa ferramenta, é fácil encontrar recursos, tutoriais e suporte em comunidades como **Github, StackOverflow**, etc.

Além dos recursos, conforme mencionado acima, conforme discutimos, o Selenium oferece suporte a uma variedade de navegadores da Web, linguagens de programação e sistemas operacionais. A figura a seguir mostra alguns deles:



Não é uma lista exaustiva, mas certamente dá a ideia básica sobre o amplo suporte que o Selenium fornece em toda a plataforma. Agora, além desses recursos, também existem certas **limitações que o Selenium não é capaz**. Vamos entender quais são essas limitações das quais o Selenium não é capaz:

**O que o Selenium não pode fazer?**

Embora o Selenium possa ajudar na automação de muitas ações do usuário, também existem **certas limitações**. Poucos deles são:

* Sem suporte para automatizar aplicativos baseados em desktop nativos - o Selenium pode ser usado para automatizar aplicativos baseados na Web executados em navegadores da Web. No entanto, ele **não pode automatizar aplicativos baseados em desktop**.
* Sem suporte para asserções e validade - Selenium fornece controle para o navegador; no entanto, **o fornecimento de asserções e mecanismos de verificação não é fornecido pelo Selenium**. O Selenium precisa emparelhar com uma estrutura de teste como JUnit, TestNG, PyTest, etc. para asserções.
* Sem suporte para digitalização de imagem e código - A automação de ***cenários de digitalização de código, como leitura de código de barras, CAPTCHA não é possível usando o Selenium***.
* Sem suporte para teste de API - o Selenium imita as ações do usuário no navegador. Portanto, ***o Selenium não oferece a capacidade de testar a API***.
* Sem suporte para teste de desempenho - ***o Selenium não pode executar verificações de desempenho ou testes de desempenho dos aplicativos da web***.
* Sem relatórios embutidos - ***A capacidade de relatórios também não é fornecida pelo Selenium***. Ou seja; o relatório pode ser feito apenas emparelhando-o com uma estrutura como JUnit, TestNG.

**Quais são os pré-requisitos para aprender Selenium?**

Embora seja fácil aprender e iniciar a automação de aplicativos da Web usando o Selenium, é sempre útil garantir que o usuário conheça os seguintes conceitos antes de começar a aprender sobre o Selenium:

* O usuário conhece os fundamentos do teste manual
* Conhecimento básico de Coding numa linguagem de programação suportada pelo Selenium.
* O usuário possui conhecimentos básicos de HTML, CSS.
* Além disso, eles possuem o conhecimento básico de XML e JSON.
* Além disso, eles conhecem o DOM e identificam um elemento da web usando um localizador no DOM

No entanto, não é necessário saber de tudo isso. A ToolsQA projetou especialmente seu tutorial Selenium que um novato pode aprender e aprender a automação rapidamente. Com esses tutoriais do Selenium WebDriver, você poderá iniciar sua jornada de teste do Selenium. De fato, se você gosta de ser chamado de Selenium Professional, **recomendamos que você também faça a Certificação Selenium** .

Agora, antes de pular para o uso do pacote Selenium, a primeira confusão que surge é qual ferramenta Selenium precisa usar para qual tipo de cenário. Vamos entender algumas das situações em que uma determinada ferramenta Selenium atenderá às necessidades:

**Qual ferramenta de teste do Selenium atende às suas necessidades?**

Como existem vários componentes do Selenium, pode ser confuso para quem é novo no Selenium decidir qual parte usar para suas necessidades de automação. Você pode usar o guia abaixo para selecionar qual componente do Selenium atenderá melhor às suas necessidades:

* ***Selenium IDE***
  + *Se você quiser aprender sobre testes de automação e Selenium.*
  + *Se você tem pouca ou nenhuma experiência anterior em automação de teste.*
  + *Caso você queira escrever casos de teste simples e exportá-los posteriormente para RC ou WebDriver.*
  + *Se você deseja executar código JavaScript personalizado usando runScript*
  + *Se você deseja exportar o script em vários idiomas.*
  + *Caso você queira segmentar no Chrome e Firefox para teste.*
* ***Selênio RC***
  + *Se você deseja escrever casos de teste em linguagem mais expressiva do que IDE.*
  + *Se você quiser testar o aplicativo em um novo navegador que suporte JavaScript.*
  + *Ou, se você quiser testar o aplicativo, que é AJAX pesado.*
* ***Selenium WebDriverName***
  + *Se você deseja usar uma linguagem de programação específica para seus casos de teste de automação.*
  + *Se você deseja testar aplicativos em diferentes plataformas usando o Selenium Grid*
  + *Ou, se você quiser testar aplicativos em CI/CD.*
  + *Se você deseja testar aplicativos e gerar relatórios personalizados em formato HTML.*
  + *Se você deseja testar sites dinâmicos modernos com muitos dados.*
* ***Selenium GRID***
  + *Se você deseja executar seus casos de teste de automação em diferentes navegadores e sistemas operacionais simultaneamente.*
  + *Se você deseja executar um vasto conjunto de testes e deseja minimizar o tempo de execução.*

**Principais conclusões**

* O Selenium é uma ferramenta de automação da Web para todos os fins que atende às necessidades significativas de automação de aplicativos da Web.
* Além disso, o Selenium é um conjunto de ferramentas que consiste em vários componentes - IDE, RC, WebDriver, GRID.
* O Selenium WebDriver é a ferramenta mais popular do pacote Selenium.
* Além do exposto, a principal razão para a popularidade do Selenium é devido à sua ampla cobertura e suporte para diferentes sistemas operacionais, linguagens de programação e navegadores da Web.

Até agora, todos nós devemos estar claros sobre os fundamentos do teste do Selenium. Vamos para o próximo artigo, onde aprenderemos sobre os detalhes do " Selenium WebDriver

**Arquitetura do Selenium WebDriver**

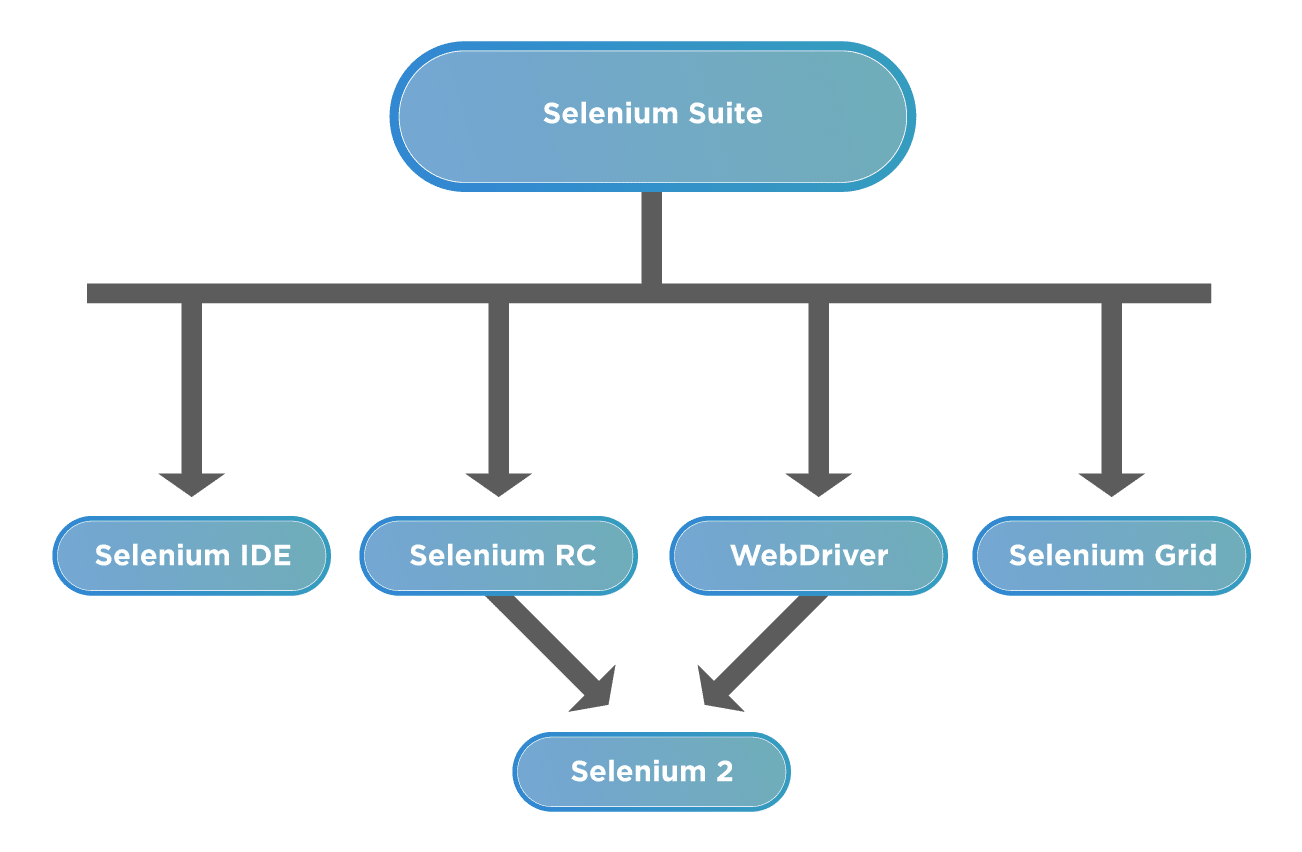
No mundo moderno, a cada dia que passa, centenas de aplicativos web e móveis estão sendo implantados na web. A equipe de controle de qualidade deve estar sempre atenta para garantir que esses aplicativos da Web sejam funcionais fora do ambiente de desenvolvimento, garantindo que a funcionalidade seja adequadamente testada antes do lançamento para o cliente. Anteriormente, essa tarefa tediosa era realizada pelos testadores manuais por meio do poder humano de observação, levando muito tempo. O Selênio entrou na indústria. Selenium é um conjunto de ferramentas viz. Selenium IDE, WebDriver, Selenium RC, etc. ajudam a equipe de controle de qualidade a simular ações do usuário no navegador da web e automatizar o fluxo do usuário, ajudando assim na execução de um grande número de casos de teste em um curto espaço de tempo.

O Selenium Webdriver é um dos membros críticos dessa família e é conhecido por sua diversidade e estabilidade para automação da web. O Selenium Webdriver tornou-se um tipo de fato para automação de interface do usuário e mais de 80% das empresas o estão usando. Vamos entender os detalhes desta ferramenta, cobrindo as seguintes seções:

* O que é Selenium WebDriver?
* Por que usar o Selenium WebDriver para automação da Web?
* Por que o Selenium WebDriver é popular?
* Quais são as desvantagens do Selenium WebDriver?
* Compreensão da arquitetura do Selenium WebDriver?
* Como o Selenium WebDriver funciona?
* Como usar o Selenium WebDriver para automação da Web

**O que é Selenium WebDriver?**

O Selenium WebDriver é um **conjunto de APIs de código aberto** que fornecem os recursos para interagir com qualquer um dos navegadores da Web modernos e, por sua vez, **automatizar as ações do usuário** com esse navegador. É um componente essencial da família Selenium. Como sabemos, o Selenium não é uma ferramenta independente; em vez disso, **é uma coleção de ferramentas** que compõem o conjunto Selenium, que foi criado quando dois projetos Selenium RC e WebDriver foram mesclados.



O WebDriver foi integrado ao Selenium RC para superar algumas das limitações do Selenium RC e agora se tornou o padrão para automação da Web. Você deve estar ciente de que após o Selenium 2, novas versões foram lançadas. Até agora, em janeiro de 2021, eles atingiram o Selenium 4.

**Por que usar o Selenium WebDriver para automação da Web?**

Agora que sabemos o que é o Selenium WebDriver e o que ele faz, vamos ver por que ele é a escolha ideal para automação da web.

|  |  |
| --- | --- |
| ***Recurso*\*** | ***Descrição*** |
| ***Automação de Página Web Dinâmica*** | O Selenium *WebDriver* pode automatizar sites dinâmicos onde o conteúdo das páginas é alterado pelas ações do usuário. |
| ***Funciona perto do navegador*** | Os fornecedores de navegadores enviam sua implementação do *WebDriver*. Portanto, eles estão fortemente acoplados ao navegador, proporcionando uma melhor experiência de teste. |
| ***tecnologia agnóstica*** | O Selenium *WebDriver* permite automatizar os casos de teste para todas as aplicações web, independentemente da tecnologia utilizada para o desenvolvimento da aplicação em teste. |
| ***Imita o usuário real*** | O Selenium *WebDriver* permite que o controle de qualidade ***imite as ações do usuário nos sites***. O Selenium WebDriver pode imitar ações típicas do usuário, como preenchimento de formulário, clique, clique duplo, pressionamento de tecla, bem como ações avançadas do usuário, como arrastar e soltar, clicar e segurar, etc. |
| ***Suporta testes entre navegadores*** | O Selenium *WebDriver* tem a vantagem mais significativa ao fazer testes entre navegadores - onde um QA pode testar o mesmo site, usando o mesmo código em diferentes navegadores. Ele permite a verificação e validação de casos de teste em vários conjuntos de navegadores ao mesmo tempo. |
| ***Suporta Execução paralela*** | Se houver mais scripts a serem executados em vários navegadores, executá-los um por um é demorado. Assim, o Selenium *WebDriver* permite a execução paralela, utilizando frameworks como o TestNG, para que a execução dos casos de teste seja mais rápida. Isso permite a execução em larga escala de casos de teste em um curto espaço de tempo. |
| ***Ver Resultados da Execução*** | O Selenium *WebDriver* permite que um QA visualize a execução ao vivo do teste automatizado executado no sistema do computador, bem como em qualquer outra máquina de pipeline CI/CD, suportando funcionalidades como captura de tela, gravação em vídeo de casos de teste, etc. |
| ***Suporta técnicas de desenvolvimento modernas*** | O Selenium *WebDriver* integra-se muito bem com os princípios modernos de desenvolvimento de software, como o Behavior Driven Development, por meio da integração com a biblioteca Cucumber. |

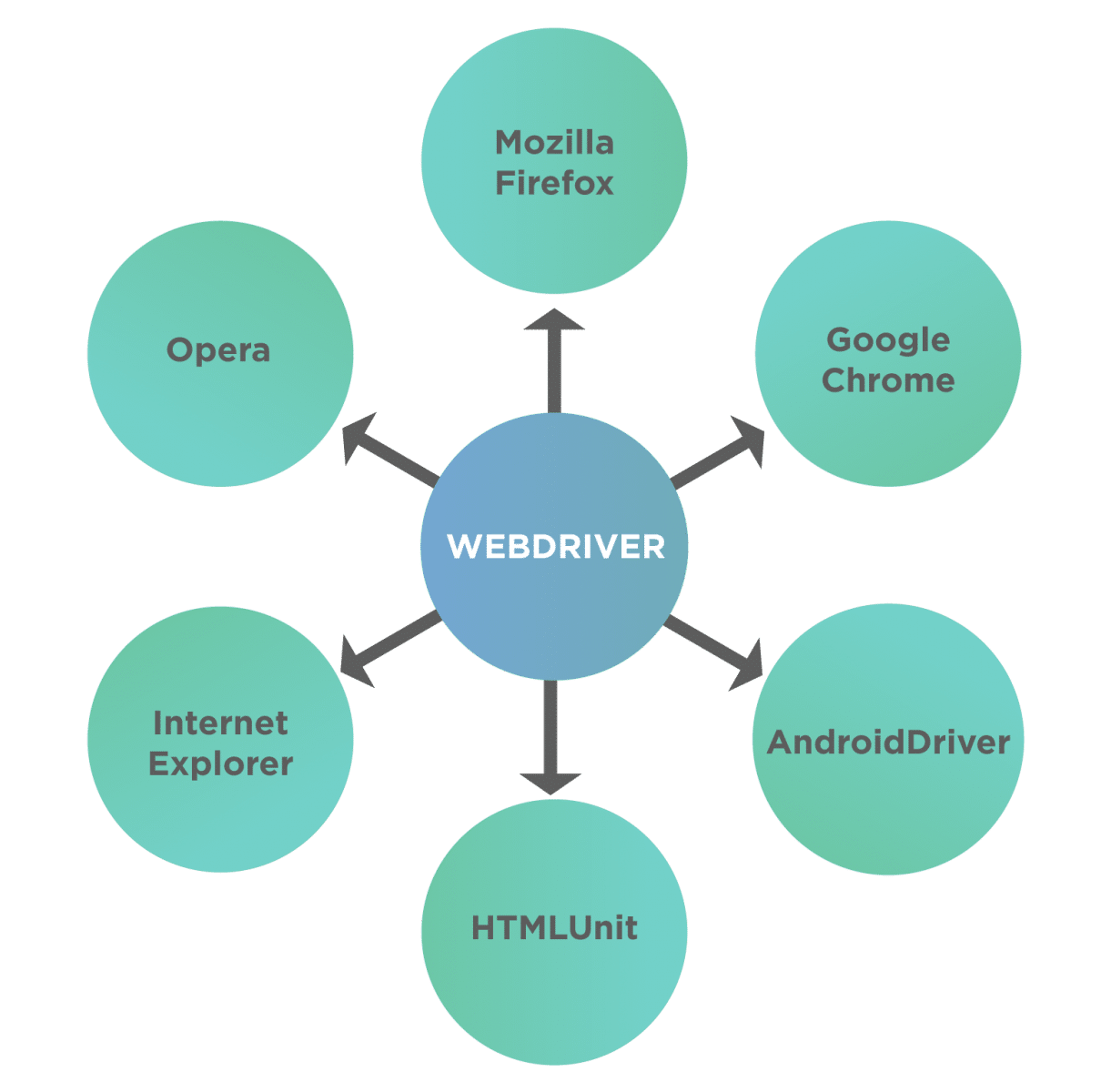
De um modo geral, o Selenium WebDriver é uma das partes mais importantes das suítes Selenium, que suporta quase todos os recursos necessários para a automação de um aplicativo da web.

**Por que o Selenium WebDriver é popular?**

Além dos recursos mencionados acima, o WebDriver , sendo parte da família Selenium, também englobava algumas das características únicas, o que aumenta sua popularidade como ferramenta de automação da web. Algumas dessas características são:

Além dos recursos mencionados acima, o WebDriver , sendo parte da família Selenium, também englobava algumas das características únicas, o que aumenta sua popularidade como ferramenta de automação da web. Algumas dessas características são:

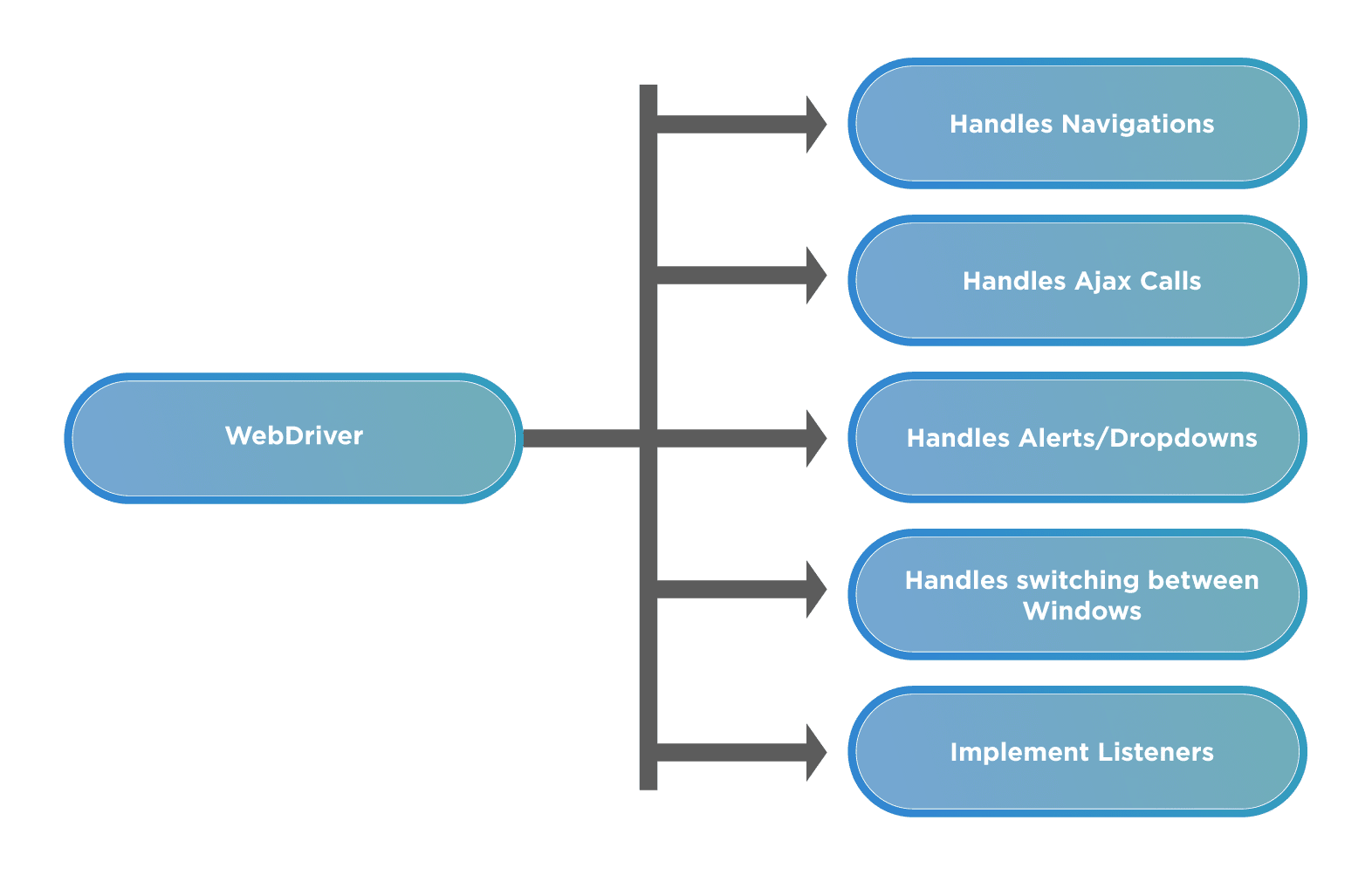
* **Compatibilidade com vários navegadores** - Uma das principais razões para a popularidade do Selenium e do WebDriver é o suporte entre navegadores usando o mesmo código. Ele oferece a capacidade de executar um código específico que imita um usuário do mundo real usando o suporte nativo de um navegador para acessar chamadas de API diretas sem a necessidade de nenhum software ou dispositivo de middleware. Veja abaixo uma lista de amostra de navegadores suportados:



* **Suporte multilíngue** - Nem todos os testadores são versados em um idioma específico. Como o Selenium fornece suporte para vários idiomas, um testador pode usar qualquer idioma fora dos idiomas suportados e, em seguida, usar o WebDriver para automação. Isso dá a liberdade de escrever código na linguagem com a qual as pessoas se sentem confortáveis.
* **Execução mais rápida** - Ao contrário do Selenium RC, o WebDriver não depende de um servidor middleware para se comunicar com o navegador. O WebDriver direciona as comunicações com os navegadores usando um protocolo definido (JSON Wire), que permite que ele se comunique mais rapidamente do que a maioria das ferramentas do Selenium. Além disso, como o próprio JSON Wire usa JSON, que é muito leve, a quantidade de transferência de dados por chamada é mínima. A figura abaixo mostra claramente como o WebDriver se comunica com o Browser:



* **Localizando elementos da Web** - Para executar ações como clicar, digitar, arrastar e soltar, primeiro precisamos identificar em qual elemento da Web (como botão, caixa de seleção, menu suspenso, área de texto) precisamos executar uma ação. Para facilitar isso, o WebDriver fornece métodos para identificar elementos da Web usando vários atributos HTML - como id, nome, classe, CSS, nome da tag, XPath, linktext etc
* **Lidando com elementos dinâmicos da web** - Há momentos em que há elementos da web na página, que mudam a cada recarregamento da página. Como os atributos HTML mudam, torna-se um desafio identificar esses elementos. O Selenium fornece vários métodos para lidar com essas situações -
* **Xpath absoluto** - contém o caminho XML completo do elemento em questão.
* **Contens( )** - o uso desses elementos funcionais pode ser pesquisado com texto parcial ou completo e pode ser usado para lidar com elementos dinâmicos.
* **Starts-With( )** - esta função é baseada na localização de elementos usando o texto inicial do atributo em questão.
* **Lidando com a espera de elementos** - Nem todas as páginas têm a mesma estrutura. Alguns são leves, enquanto outros têm uma quantidade considerável de manipulação de dados ou chamadas AJAX. Muitas vezes, os elementos da web levam algum tempo para carregar. Para dar conta desse WebDriver, forneceu vários mecanismos de espera que podem ser usados para pausar a execução do script por um período de tempo necessário com base em determinadas condições e, em seguida, continuar assim que a condição for preenchida. A figura a seguir mostra uma lista de amostra que mostra os recursos do WebDriver, que ajuda a lidar com o comportamento dinâmico das páginas da web.



**Quais são as desvantagens do Selenium WebDriver?**

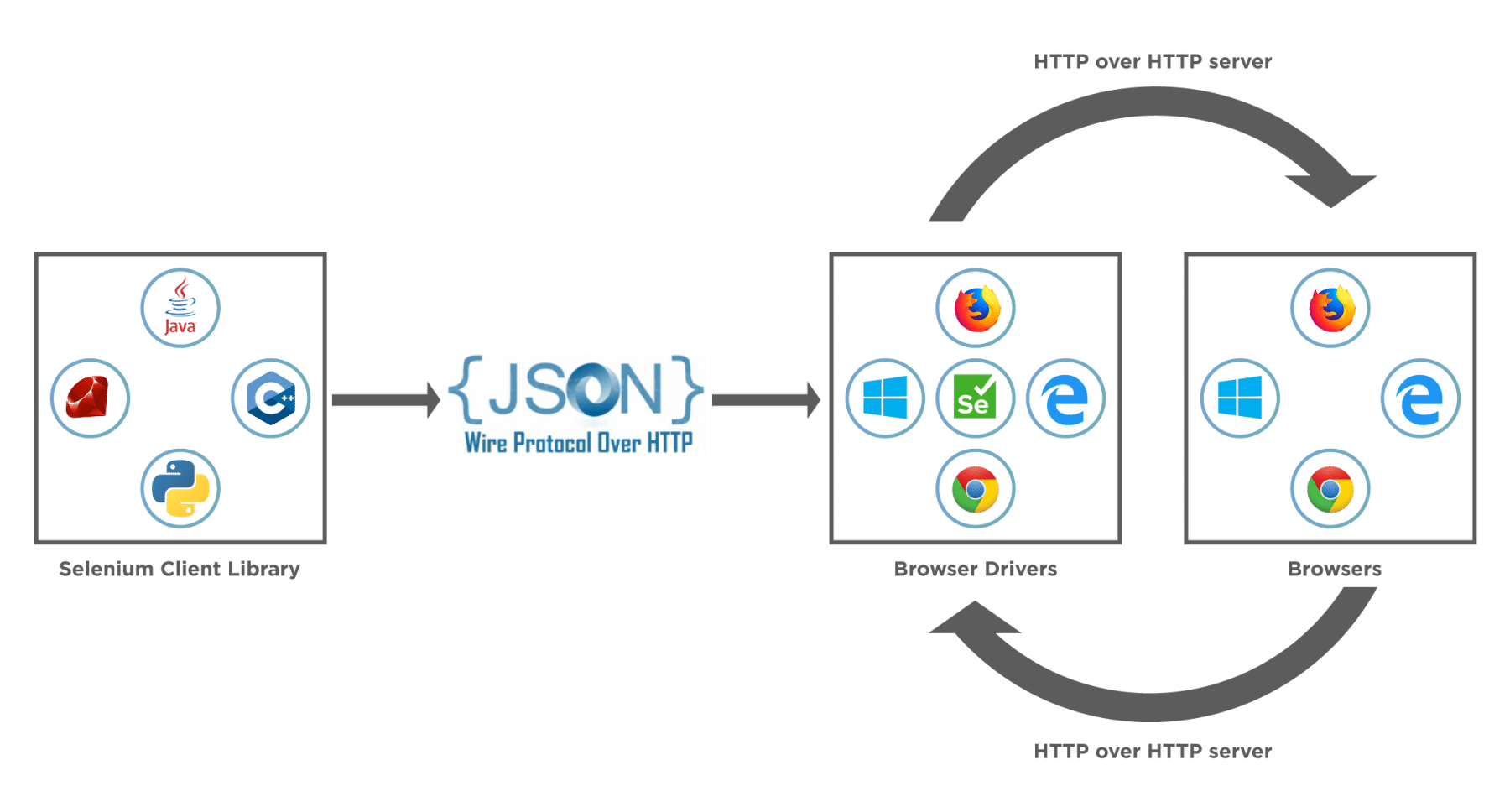
Embora o Selenium trabalhe muito na solução da \*IU\* e automação funcional de aplicativos da web, ele tem suas desvantagens. Vejamos algumas das deficiências/desvantagens:

* **Requer conhecimento e experiência em programação** - Como o WebDriver permite automatizar as ações do usuário usando código escrito em uma determinada linguagem de programação, qualquer pessoa que queira usá-lo deve ter um entendimento básico de como funciona a codificação no idioma. As pessoas que não entendem de codificação em uma linguagem de programação acharão difícil usar o WebDriver.
* **Sem suporte para aplicativos de desktop.** - Ecossistema Selenium, incluindo WebDriver, foi construído para a automação de aplicações web. Portanto, se você deseja automatizar aplicativos baseados no Windows, não poderá fazê-lo.
* **Sem suporte ao cliente** - o ecossistema Selenium, incluindo o WebDriver, é totalmente de código aberto, o que significa que é conduzido por indivíduos e não por qualquer empresa. Por causa disso, ***não há uma equipe de suporte dedicada para analisar seus problemas***. Se uma pessoa está presa em algum lugar, existem muitas comunidades, fóruns nos quais uma pessoa pode confiar, mas isso é tudo.
* **Nenhum Repositório de Objetos Integrado** - Ferramentas pagas como UFT/QTP fornecem um local centralizado para armazenar objetos/elementos, chamado de Repositório de Objetos. Essa capacidade não é fornecida por padrão no Selenium WebDriver. Isso pode ser superado usando abordagens como o Page Object Model, mas requer conhecimentos e habilidades de codificação consideráveis.
* **Falta de relatórios integrados** - Selenium WebDriver pode ajudá-lo a executar seus testes de automação, mas para fornecer um recurso de relatórios, você precisa integrá-lo a uma estrutura de teste como Junit, TestNG, PyTest, Allure, etc.
* **Gerenciando dependências do navegador - Selenium** - Como o Selenium depende da compatibilidade entre os drivers do navegador e o próprio navegador, muitas vezes devido a incompatibilidade ou bugs no driver do navegador ou no navegador, quebras de funcionalidade e os usuários precisam contar com o suporte da comunidade para consertá-lo.

**Compreensão da arquitetura Selenium WebDriver**

Sendo parte do sistema de componentes geral, deduzimos que o Selenium WebDriver não é uma ferramenta de teste independente. É composto por vários componentes que são necessários para executar testes. Estes são os componentes arquiteturais do Selenium.

Então, primeiro vamos dar uma olhada nesta imagem abaixo

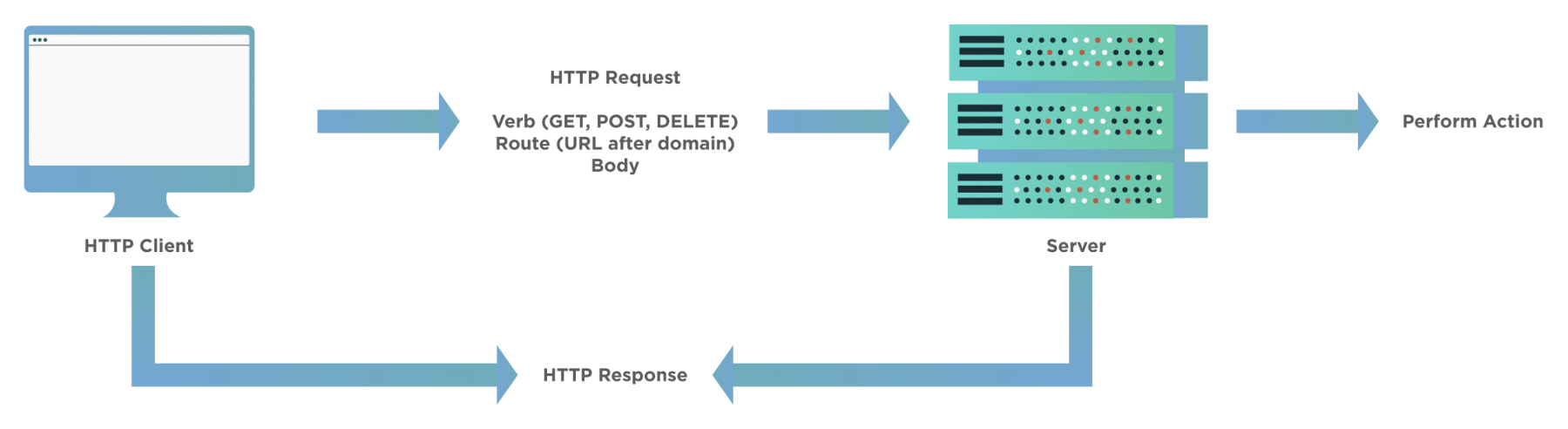


Esta imagem nos fala sobre a arquitetura central do Selenium webdriver e os ***principais componentes do Selenium*** que compõem o WebDriver.

* ***Bibliotecas do cliente Selenium WebDriver / Ligações de idioma*** – Os testadores de software desejam selecionar idiomas com os quais se sintam confortáveis. Como a arquitetura do WebDriver oferece suporte a diferentes linguagens, há vinculações disponíveis para uma variedade de linguagens, como **Java, C#, Python, Ruby, PHP** etc. desligado. É assim que o Selenium Achitecture fornece flexibilidade aos testadores para fazer a automação em sua zona de conforto
* **JSON WIRE PROTOCOL** - Conforme a Selenium Architecture acima, o JSON Wire Protocol facilita toda a comunicação que está acontecendo no **Selenium entre o navegador e o código**. Este é o coração do selênium. JSON Wire Protocol fornece um meio para transferência de dados usando uma API RESTful (Representational State Transfer) que fornece um mecanismo de transporte e define um serviço da Web RESTful usando JSON sobre HTTP.
* **Drivers de navegador** – Como existem vários navegadores suportados pelo Selenium, cada navegador tem sua própria implementação do padrão W3C fornecido pelo Selenium. Como tal, estão disponíveis binários específicos do navegador que são específicos do navegador e ocultam a lógica de implementação do usuário final. O protocolo JSONWire estabelece uma conexão entre os binários do navegador e as bibliotecas do cliente.
* ***Navegadores*** *– O Selenium só poderá executar testes nos navegadores se eles estiverem instalados localmente, seja na máquina local ou nas máquinas servidoras. Portanto, a instalação do navegador é necessária.*

**Como o Selenium WebDriver funciona?**

Na seção acima, vimos a arquitetura do Selenium. Agora vamos ver como acontece toda a comunicação nos bastidores? Dê uma olhada na imagem abaixo - isso mostra uma visão de como o fluxo de trabalho real se parece.



Quando um usuário escreve um código WebDriver no Selenium e o executa, as seguintes ações acontecem em segundo plano:

* Uma **solicitação HTTP** é gerada e vai para o respectivo driver do navegador (Chrome, IE, Firefox). Há uma **solicitação individual para cada comando** do Selenium.
* O driver do navegador recebe a solicitação por meio de um servidor HTTP.
* O servidor HTTP decide quais ações/instruções precisam ser executadas no navegador.
* O navegador executa as instruções/etapas conforme decidido acima.
* O servidor HTTP recebe o status de execução e envia de volta o status para um script de automação, que mostra o resultado (conforme passado ou uma exceção ou erro)

**Como usar o Selenium WebDriver para automação da Web?**

O Selenium WebDriver fornece uma abordagem muito simples, amigável e amigável ao código para automação usando vários navegadores. Como ele oferece suporte à maioria dos principais fornecedores de navegadores, é apenas uma questão de usar o respectivo driver e navegador do navegador e configurar o Selenium para usar o mesmo.

Para qualquer script de teste do Selenium, geralmente existem as 7 etapas a seguir, que se aplicam a todos os casos de teste e a todos os aplicativos em teste ( AUT ):

* + 1. **Crie uma instância do WebDriver específica para o Browser:**
* Ex.: Para criar uma instância do driver Firefox, podemos utilizar os seguintes comandos:

|  |
| --- |
| import org.openqa.selenium.WebDriver;  import org.openqa.selenium.firefox.FirefoxDriver;  WebDriver driver = new FirefoxDriver(); |

* + 1. **Navegue até a página da Web desejada que precisa ser automatizada**:
* Ex.: Para navegar até o "https://demoqa.com/text-box", podemos utilizar o seguinte comando:

|  |
| --- |
| driver.get("https://demoqa.com/text-box") |

* + 1. **Localize um elemento HTML na página da Web**:
* Para interagir com uma página da web, precisamos localizar os elementos HTML na página da web. Podemos usar qualquer uma das estratégias de localizador de elementos mencionadas em "Localizadores de Selenium" . Ex.: se quisermos obter a caixa de texto " Nome Completo ", podemos usar os seguintes comandos:

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

WebElement usernameElement = driver.findElement(By.id("userName"));

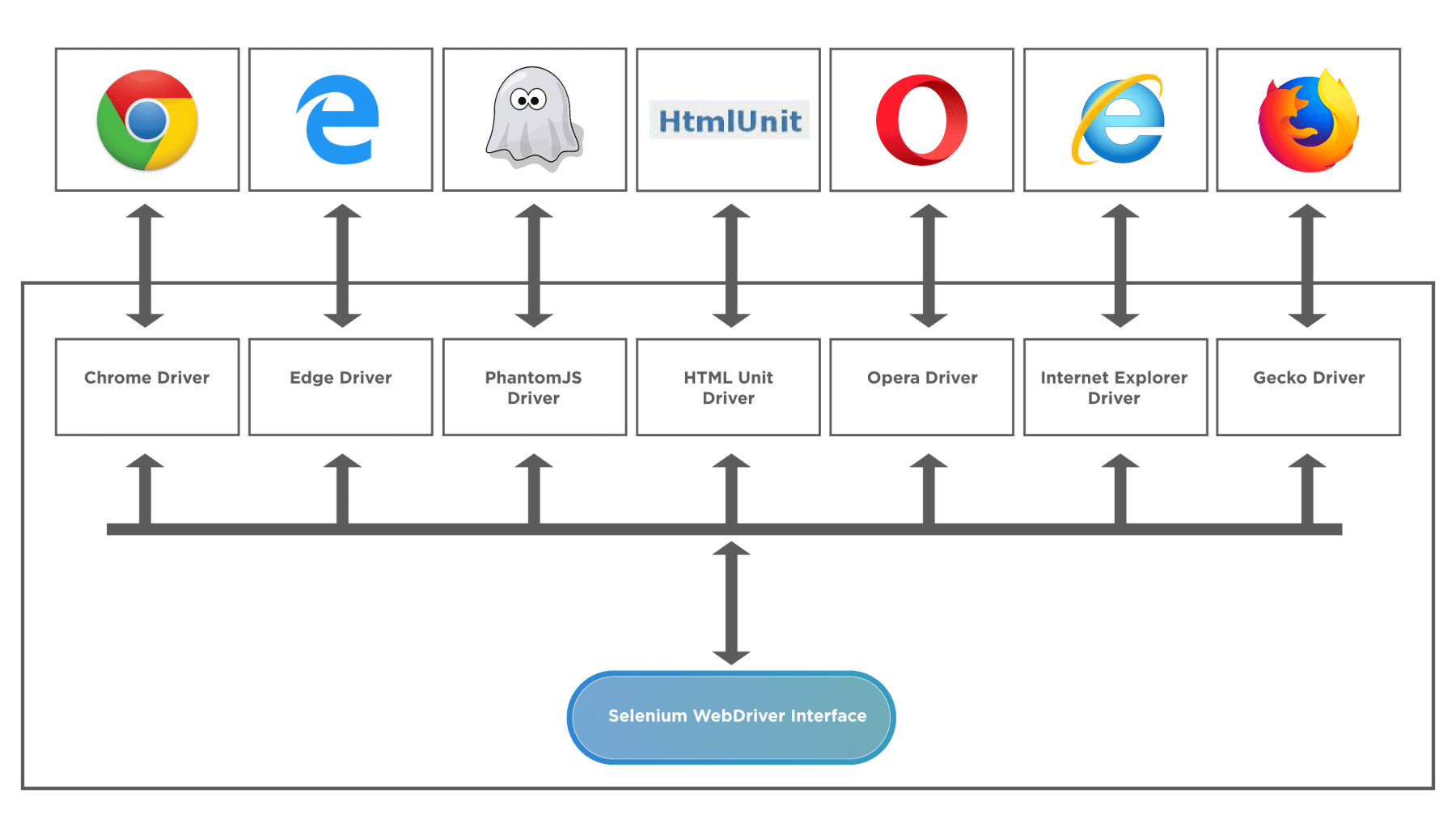
* + 1. Execute uma ação em um elemento HTML:
* Podemos realizar certas ações nos elementos HTML, como digitar algo usando o método SendKeys , clicar no elemento se for um botão. Ex.: se quisermos digitar o nome na caixa de texto identificada, podemos utilizar os seguintes comandos

usernameElement.sendKeys("Ravinder Singh");

* + 1. Execute testes e registre os resultados dos testes usando uma estrutura de teste.

E terminamos de usar o WebDriver para identificar e executar as ações necessárias no aplicativo da Web. Dependendo do navegador, no qual precisamos testar nosso aplicativo, podemos usar o WebDriver correspondente.

Aqui está uma lista de vários navegadores e seus respectivos drivers de navegador:



Recentemente, a Microsoft mudou seu navegador Edge para a mesma plataforma que o Chromium ( que é o pai do Chrome ) e, devido a isso, o ChromeDriver agora também pode oferecer suporte ao Microsoft Edge Chromium.

**Perguntas comuns**

* **Tipos de Dados e Variáveis**
* **Operators**
* **Tomando uma decisão**
* **Arrays**
* **Loops**
* **Classes and Objects**
* **Class Constructors**
* **String Class**
* **Set Up Java**
* **Download and Start Eclipse**
* **Download Selenium WebDriver**
* **Configure Selenium WebDriver with Eclipse**
* **First Selenium Test Case (refazer)**

* **Browser Commands**
* **Selenium Navigation Commands**
* **WebElement Commands**

**Find Element and Find Elements in Selenium**

* **Handle CheckBox in Selenium WebDriver**