Keyword-Driven Framework for Selenium and Protractor

Serguei Kouzmine
kouzmine_serguei@yahoo.com

Keyword-Driven Frameworks

The original Keyword Driven Framework few columns

* **Summary**: brief description of the step

* **Target**: name of the Web Page object/element, like "Link" or "Input"

* **Action**: name of the action, which will be performed on Target Element such as click, open browser, input text etc.

* **Data**: any value which is needed by the Object to perform any action, like text value for input field

http://toolsqa.com/selenium-webdriver/keyword-driven-framework/introduction/

Keyword-Driven Frameworks

Advantages from taking Keyword Driven Framework approach:

Less Technical Expertise: manual testers or non technical testers can easily write test scripts

Easy To Understand: With no coding is exposed, the test flow is easy to read and understand. Keywords & actions are descriptive. The implementation detail of the script is hidden from the users

Early Start: One can start building Keyword Driven test cases immediately defering technically challenging tasks to a later stage. Keyword steps are quick to map from requirements documentation or manual test steps.

Re-usability: Stable and powerful Execution Engine in Keyword Driven Framework encourage code re-usability.

Automation: Excel file is (a lot) easier to produce by a recording tool than a full blown program.

Keyword-Driven Frameworks

The original Keyword Driven Framework [suggests](http://toolsqa.com/selenium-webdriver/keyword-driven-framework/introduction/) identifying only four columns

* __Summary__: *a brief description of the step*

* __Target__: *the name of the Web Page object/element, like "Link" or "Input"*

* __Action__: *name of the action, which will be performed on Target Element such as click, open browser, input text etc.*

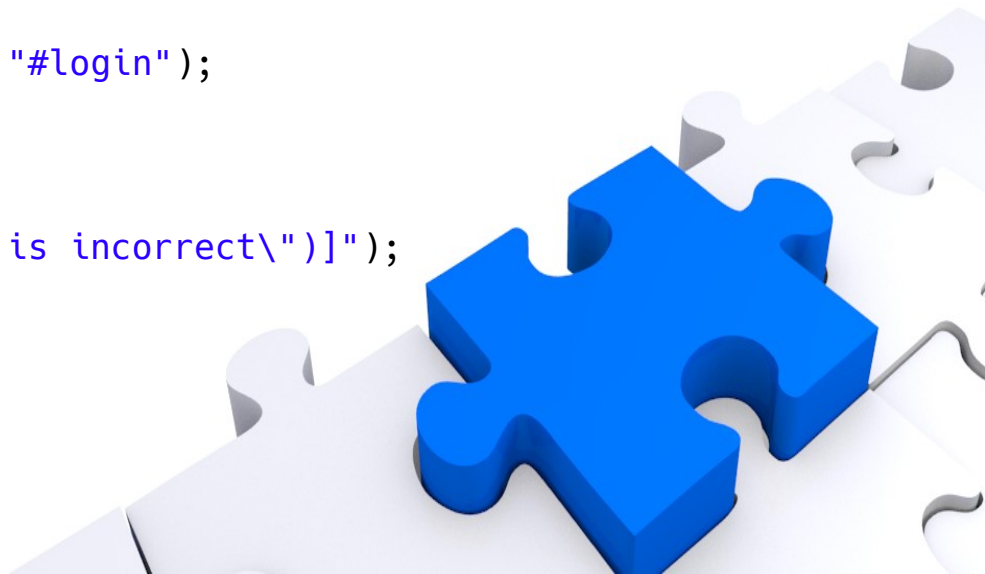
* __Data__: *any value which is needed by the Object to perform any action, like text value for input field

Programming quiz, part I

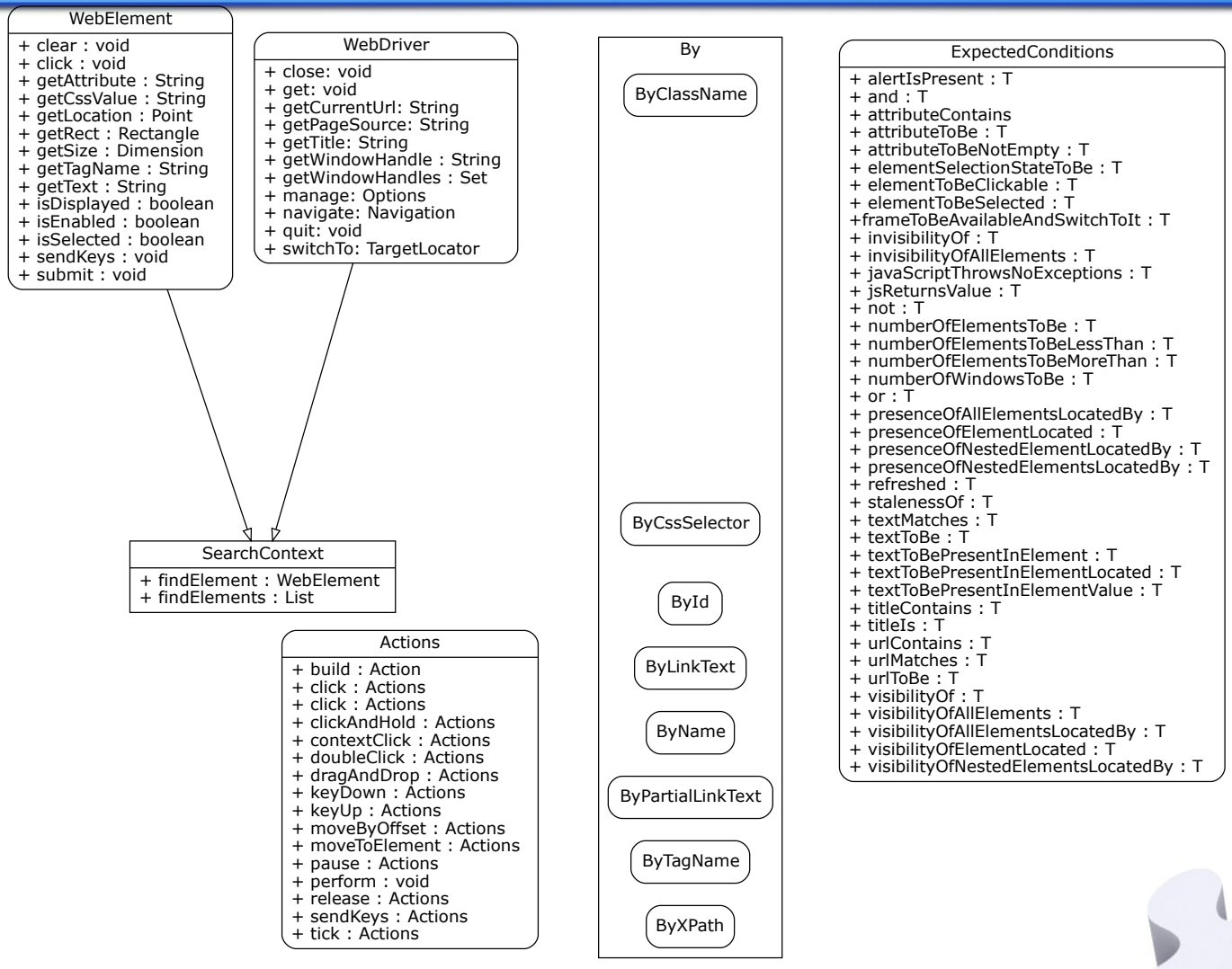
```
 import org.openqa.selenium.WebDriver;

WebDriver driver = new ChromeDriver();
WebDriverWait wait = new WebDriverWait(driver, 10);

 driver.get("http://www.store.demoqa.com"); // open the start page

 driver.findElement(By.xpath("//*[@id='account']/a")).click();  // go to login page
 wait.until( driver ->
 driver.getCurrentUrl().matches("http://store.demoqa.com/products-page/your-account/"));

 driver.findElement(By.id("log")).sendKeys("testuser_3"); // log in
 driver.findElement(By.id("pwd")).sendKeys("Test@123");
 driver.findElement(By.id("login")).click();

 // confirm the error message is displayed
 assertThat(wait.until(ExpectedConditions.visibilityOf(driver.findElement(
     By.cssSelector("#ajax_loginform > p.response")))).getText(),
       containsString("The password is incorrect"));
 driver.quit();
```

Programming quiz, part II

```
import io.webfolder.cdp.*;

SessionFactory factory = new Launcher().launch();
Session session = factory.create();

session.navigate("http://www.store.demoqa.com"); // navigate to start screen

session.click("#account > a"); // go to login page
session.waitUntil(s -> s.getLocation().matches(
"http://store.demoqa.com/products-page/your-account/"));

session.focus("#log"); session.sendKeys("testuser_3"); // log in
session.focus("#pwd").sendKeys("Test@123");
executeScript(session, "function(){this.click();}", "#login");

// confirm the error message is displayed
List<String> errMsgs = session.getObjectIds(
  "//*[contains (text(), \"The password you entered is incorrect\")]");
assertTrue(errMsgs.size() > 0);

session.stop(); session.close();
```

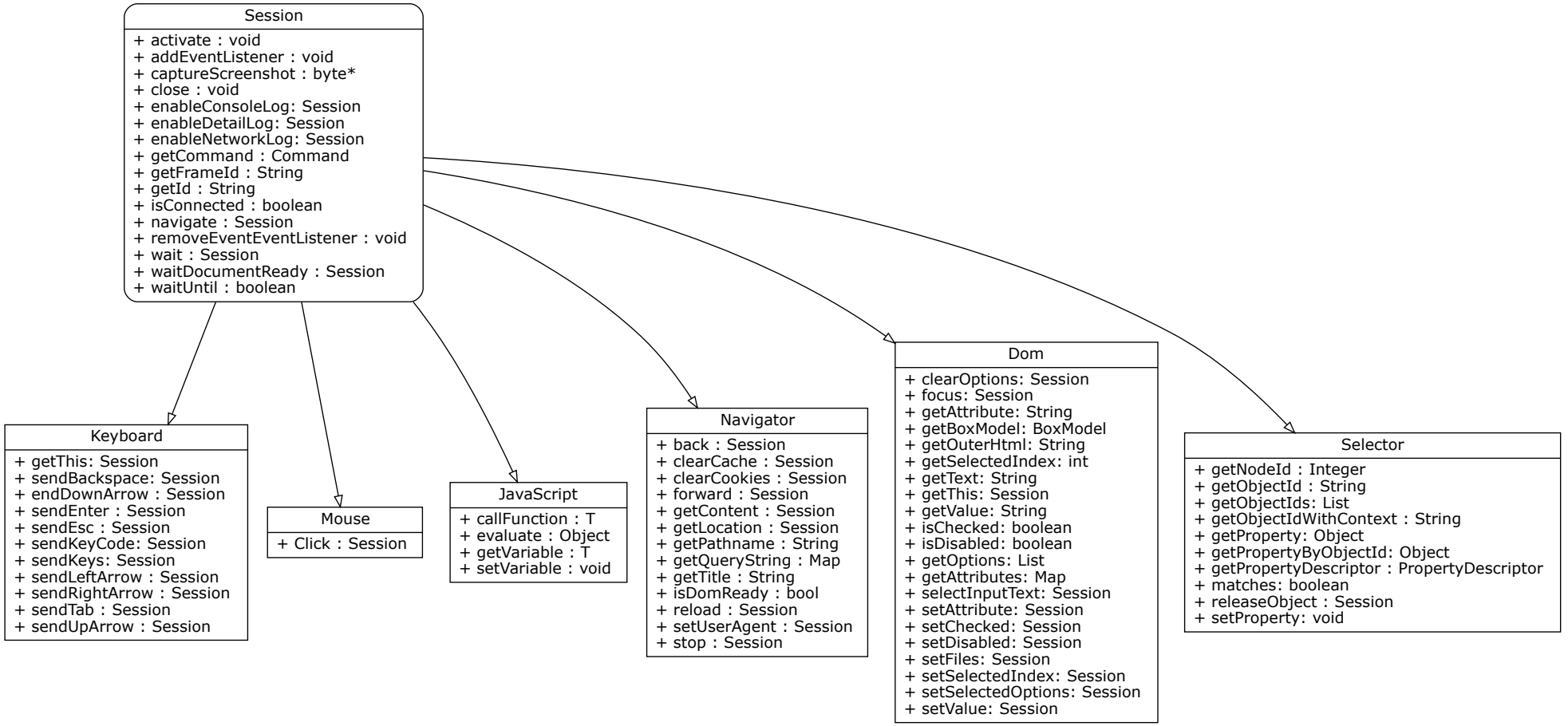


Core Selenium Class Hierarchy



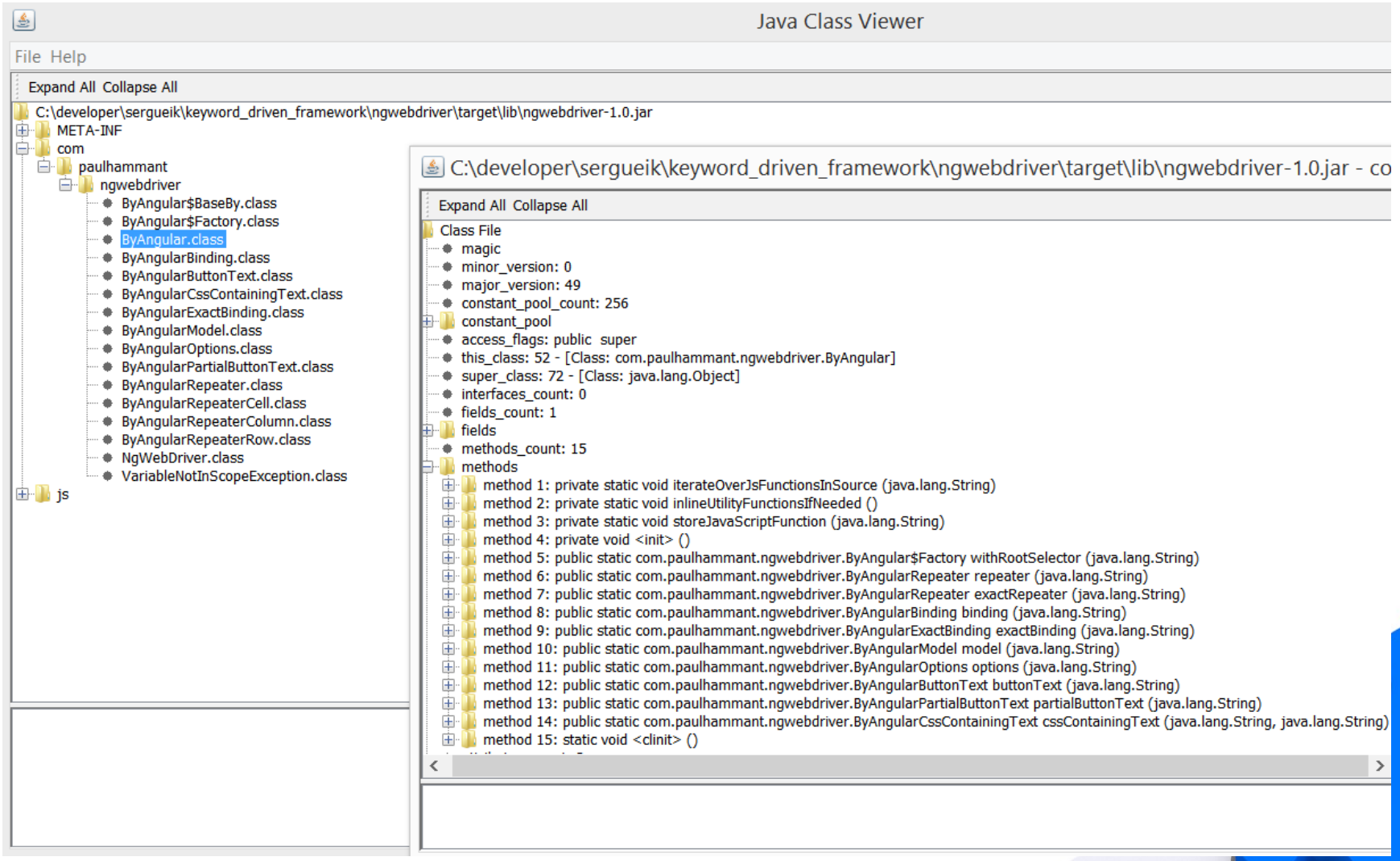
https://seleniumhq.github.io/selenium/docs/api/java/

CDP Class Hierarchy



https://webfolder.io/cdp4j/javadoc/index.html

Java Protractor Wrapper Methods



Alernative Java Protractor Wrapper Methods

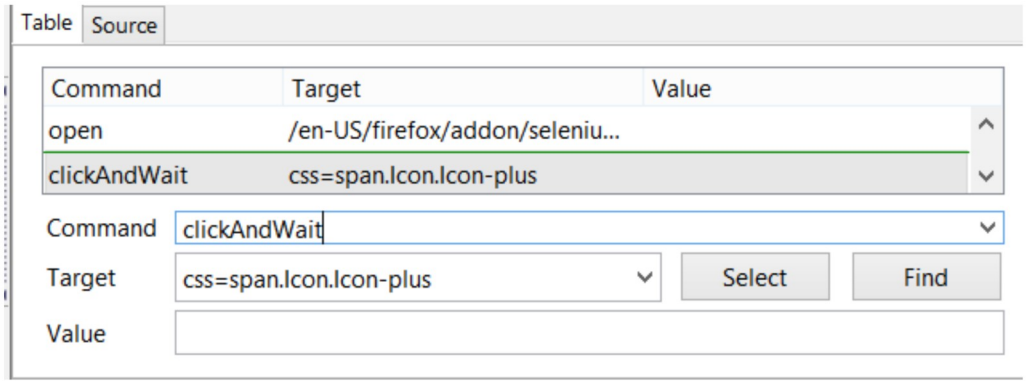
Expand All Collapse All

- Class File
  - magic
  - minor_version: 0
  - major_version: 52
  - constant_pool_count: 81
  - constant_pool
  - access_flags: public final super
  - this_class: 29 - [Class: com.jprotractor.NgBy]
  - super_class: 5 - [Class: java.lang.Object]
  - interfaces_count: 0
  - fields_count: 0
  - methods_count: 16
  - methods
    - method 1: private void <init> ()
    - method 2: public static org.openqa.selenium.By binding (java.lang.String)
    - method 3: public static org.openqa.selenium.By binding (java.lang.String, java.lang.String)
    - method 4: public static org.openqa.selenium.By buttonText (java.lang.String)
    - method 5: public static org.openqa.selenium.By cssContainingText (java.lang.String, java.lang.String)
    - method 6: public static org.openqa.selenium.By input (java.lang.String)
    - method 7: public static org.openqa.selenium.By model (java.lang.String)
    - method 8: public static org.openqa.selenium.By model (java.lang.String, java.lang.String)
    - method 9: public static org.openqa.selenium.By partialButtonText (java.lang.String)
    - method 10: public static org.openqa.selenium.By repeater (java.lang.String)
    - method 11: public static org.openqa.selenium.By repeaterColumn (java.lang.String, java.lang.String)
    - method 12: public static org.openqa.selenium.By repeaterElement (java.lang.String, java.lang.Integer, java.lang.String)
    - method 13: public static org.openqa.selenium.By repeaterRows (java.lang.String, java.lang.Integer)
    - method 14: public static org.openqa.selenium.By options (java.lang.String)
    - method 15: public static org.openqa.selenium.By selectedOption (java.lang.String)
    - method 16: public static org.openqa.selenium.By selectedRepeaterOption (java.lang.String)
  - attributes_count: 1
  - attributes

TestCase, core Selenium

	A	B	C	D	E	F	G	H	I
1	Keyword	param1	param2	param3	param4	param5	param6	param7	status
2	CREATE_BROWSER								Passed
3	GOTO_URL	http://www.seleniumeasy.com/test/						1	Passed
4	clickLink	css	#navbar-brand-centered > ul:nth-child(1) > li:nth-child(1) > a						Passed
5	CLICK_LINK	text	Simple Form Demo						Passed
6	SET_TEXT	css	#user-message			hello			Passed
7	GET_TEXT	css	#user-message						Passed
8	GET_ATTR	css	#user-message			outerHTML			Passed
9	getElementAttribute	css	#user-message			value			Passed
10	CLOSE_BROWSER								Passed

Each core Selenium locator accepts a *single* input argument of a specific *type.* Inspired by Selenium IDE Command, Target, Value columns, but with narrow choice of commands.



Test Case, Protractor

A	B	C	D	E	F	G	H	I
Keyword	param1	param2	param3	param4	param5	param6	param7	status
CREATE_BROWSER								Passed
GOTO_URL	http://juliemr.github.io/protractor-demo/							Passed
SET_TEXT	model	first			40			Passed
SET_TEXT	model	second			2			Passed
GET_ATTR	options	value for (key, value) in operators			innerHTML			Passed
CLICK_BUTTON	buttontext	Go!						Passed
GET_TEXT	binding	latest						Passed
CLOSE_BROWSER								

Protractor alows test operate the very same language (attibutes) as Angular Web developer used when designing the page by finding elements by their *binding*, *model* , *repeater* etc.
It also provides Angular agnostic methods like *buttonText*, *cssContainingText*

Test Case, Protractor

	A	B	C	D	E	F	G	H	I
1	Keyword	param1	param2	param3	param4		param6	param7	status
2	CREATE_BROWSER								
3	GOTO_URL	http://amitava82.github.io/angular-multiselect/							
4	VERIFY_TAG	model	selectedCar			am-multiselect			
5	CLICK_BUTTON	buttontext	Select Some Cars						
6	WAIT							1000	
7	VERIFY_TEXT	repeaterElement	i in items	2	i.label	Honda			
8	CLICK_LINK	repeaterElement	i in items	1	i.label				
9	CLICK_LINK	repeaterElement	i in items	0	i.label				
10	WAIT							1000	
11	VERIFY_TEXT	css	am-multiselect > div > button			There are 2 car(s) selected			
12	CLOSE_BROWSER								

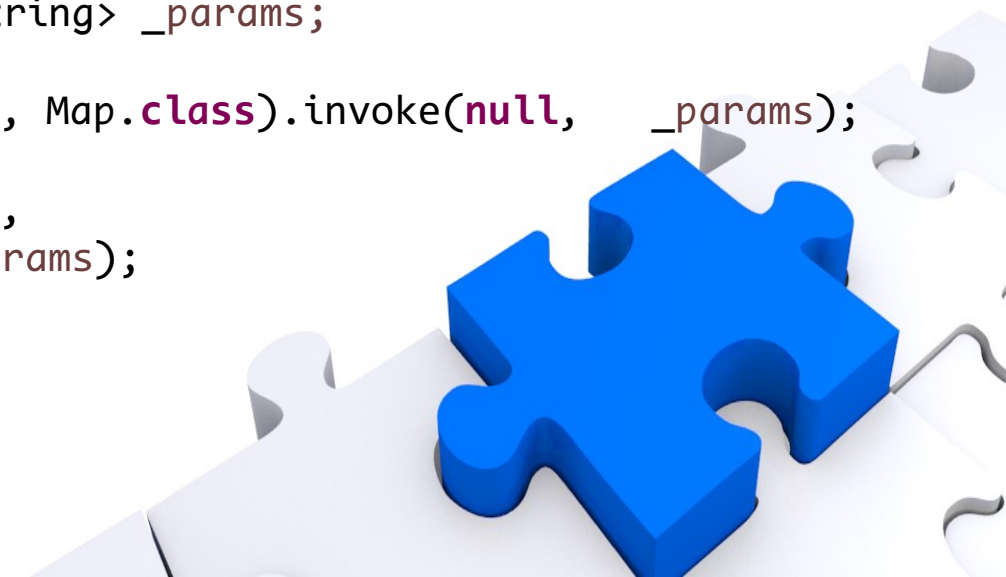
Protractor locates repeated elements relying on Angular attributes the individual elements and groups of elements were created on the web page by MVC. When the target element belongs to a set, additional coordinates are required to identify the set and the position of the member

The Keyword Table becomes *sparse*

Implementation

Keyword proceccing through Java Reflection

```
 private static Map<String, String> methodTable = new HashMap<>();
   static {
     methodTable.put("CLICK_BUTTON", "clickButton");
     ...
   }
 private static final Class<?> _class = Class.forName("Keyword Library class");

public static void callMethod(String keyword,
                              Map<String, String> _params;
   // for static methods
   _class.getMethod(methodTable.get(keyword), Map.class).invoke(null,   _params);
   // for instance methods
   _class.getMethod(methodTable.get(keyword),
Map.class).invoke( _class.newInstance(), _params);
```



Loading Steps from Excel or OpenOffice spreadsheet

```
List<Object[]> steps = utils.createDataFromExcel2003(testCase);
   // utils.createDataFromExcel2007(testCase);
   // utils.createDataFromOpenOffice(testCase);

for (int step = 0; step < steps.size(); step++) {
  Object[] row = steps.get(step);
  Map<String, String> data = new HashMap<>();
  String keyword = (String) row[0];
  for (int col = 1; col < row.length; col++) {
   data.put(String.format("param%d", col), row[col].toString());
  }
  KeywordLibrary.callMethod(keyword, data);
}
```

Keyword method implementanion

```
 provate static WebDriver driver;
 private static NgWebDriver ngDriver;
 private static WebElement element;
 private static String status;
 private static String result;

public static void clickButton(Map<String, String> params) {
   element = _findElement(params);
   if (element != null) {
     highlight(element);
     element.click();
     status = "Passed";
   } else {
     status = "Failed";
   }
 }
```

Extracting arguments from params map

```
public static void verifyText(Map<String, String> params) {
  boolean flag = false;
  expectedText = params.get("param5");
  element = _findElement(params);
  if (element != null) {
    highlight(element);
    flag = element.getText().equals(expectedText);
  }
  if (flag)
    status = "Passed";
  else
    status = "Failed";
}
```

Find element using Selenium and Protractor

```
private static WebElement _findElement(Map<String, String> params) {
  strategy = params.get("param1");
  if (!strategies.containsKey(strategy)) {
   throw new RuntimeException("Unknown strategy: " + strategy);
  }
  value = params.get("param2");
  if (params.containsKey("param3")) {
    row = params.get("param3");
  }
  if (params.containsKey("param4")) {
    column = params.get("param4");
  }

  if (params.containsKey("param5")) {
    containedText = params.get("param5");
  }
  if (params.containsKey("param6")) {
    tagName = params.get("param6");
  }
```

Supported Locator Strategies

```
private static Map<String, Method> strategies = new HashMap<>();
  strategies.put("css",
    By.class.getMethod("cssSelector", String.class));
  strategies.put("id",
    By.class.getMethod("id", String.class));

  strategies.put("binding",
    NgBy.class.getMethod("binding", String.class));
  strategies.put("repeaterColumn",
    NgBy.class.getMethod("repeaterColumn", String.class,
    String.class));
  strategies.put("repeaterElement",
    NgBy.class.getMethod("repeaterElement", String.class,
    Integer.class, String.class));

  strategies.put("repeaterRow",
                              methodMissing);
```

Phony method

```
// phony method
Method methodMissing = null;
try {
  @SuppressWarnings("rawtypes")
  Constructor<Method> methodConstructor = Method.class
   .getDeclaredConstructor(new Class[{ String.class,
      String.class });
  methodConstructor.setAccessible(true);
  methodMissing = (Method) methodConstructor.newInstance();
} catch (NoSuchMethodException    |
         IllegalAccessException   |
         InstantiationException   |
         IllegalArgumentException |
        InvocationTargetException e) {
    System.out.println(e.toString()); // slurp
}
```

Find element using Selenium and Protractor, contd.

```
WebElement _element = null;

  switch (strategy) {
    case "binding":
      _element = ngDriver.findElement(NgBy.binding(value));
      break;

    case "css":
      _element = driver.findElement(By.cssSelector(value));
      break;

   case "repeaterRow":
    case "repeaterRows":
      _element = ngDriver.findElement(NgBy.repeaterRows(
                 value, row);
      break;
   }
   return _element;
}
```

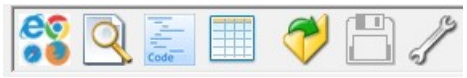
Find element extending Selenium and Protractor, contd.

```
case "text":

 // Option 1: construct XPath selector
 Map<String, String> newParams = new HashMap<>();
   newParams.put("param1", "xpath");
   newParams.put("param2", String.format(
     "//%s[contains(normalize-space(text()),'%s')]",
          (tagName != null) ? tagName : "*", value));

 _element = _findElement(newParams);

 // Option 2: use Java streams/filter
 if (tagName != null) {
   _element = driver.findElements(
      By.tagName(tagName)).stream().filter(
       o -> o.getText().contains(value))
       .findFirst().get();
 }
 break;
```

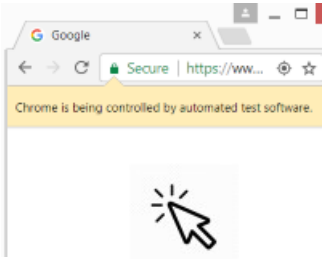
Finding elements with explicit wait

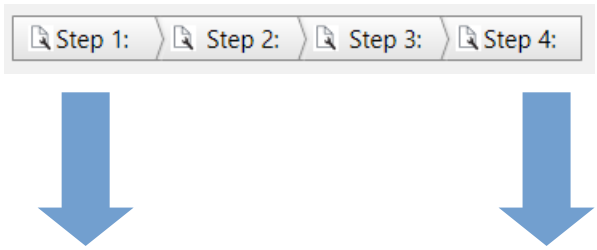
```
 switch (strategy) {
   case "css":
     locator = By.cssSelector(selectorValue);
     break;
 }
timeout = (long) (Float.parseFloat(params.get("param7")));
_wait = new WebDriverWait(driver, timeout);
_element = _wait.until(new ExpectedCondition<WebElement>() {
     @Override
     public WebElement apply(WebDriver d) {
       Optional<WebElement> e = d.findElements(locator)
                                 .stream()
                                 .findFirst();
       if (e.isPresent()) // log some debugging info
           logger.debug("find using strategy: " + strategy +
           " => " + e.get().getAttribute("outerHTML"));
       return (e.isPresent()) ? e.get() :
                          (WebElement) null;
     }
  });
```

Element Recorder Flow



```
// https://chromium.googlesource.com/chromium/src.git
getCssSelectorOf = function(element) {
  hello('getCssSelectorOf ' + element.tagName);
  var specialAttributesArray = ['href', 'src', 'title', 'alt'];
  if (!(element instanceof Element))
    return;
  var path = [];
  while (element.nodeType === ELEMENT_NODE) {
    var selector = element.nodeName.toLowerCase();
    if (element.id && path.length != 0) {
      if (element.id.indexOf(' ') > -1) {
        selector += '[id = "' + element.id + '"]';
      } else {
        selector += '#' + element.id;
      }
      path.unshift(selector);
      break;
    } else if (element.className) {
      var attr = element.className;
      // ignore className attributes with special characters
```





```
{
    ElementCssSelector: 'a[id = "js-link-box-ru"] > strong.highlight',
    ElementSelectedBy: 'ElementCssSelector',
    ElementCodeName: 'example',
    Command: 'AddElement',
    ElementPageURL: 'https://www.wikipedia.org/',
    ElementTagName: 'STRONG',
    CommandId: '8c8f535d-907a-4b3c-8d90-df2fd4be088a',
    ElementStepNumber: '2',
    ElementText: 'Русский',
    ElementXPath: 'id("js-link-box-ru")/strong[1]'
}
```

Element	Action Keyword	Selector Choice	Selector Value
step 1	CLICK_LINK	cssSelector	div#gs_lc0 > input.g
step 2	VERIFY_TEXT	xpath	div#rso > div._NId >
step 3	VERIFY_ATTR	id	a[id = "js-link-box-ru
step 4	SEND_KEYS	text	div[id = "mw-conten

```
@Test
public void invalS

  // go to login p
  driver.findEleme
  wait.until( driv
```

Flow explained

- The **Toolbar** launches the **Browser** and lets it navigate to the target **page**
- The **Tooolbar** injects into the browser the Javascript **Snippet**, which attaches itself to **Mouse Events** (like **Selenium IDE**)
- After a M**ouse Event** is fired, helper functions are generate the CSS, Xpath locators, extracts various attibutes of the **Target Element** - in a similar fashion **Firebug** works.
- The **Toolbar** continueously checks if the **Target Element** information is ready for pickup and when so, loads it, similar to how Selenium Wait works.
- The process repeats for the next Element in question.
- After the information about all elements was collected,

  the **Toolbar** generates the **Excel Flow** and/or the

  QA Test Program utilizing a template(s)

Resources

- https://github.com/angular/protractor
- https://github.com/bbaia/protractor-net
- https://github.com/paul-hammant/ngWebDriver
- https://github.com/caarlos0/jProtractor
- https://github.com/henrrich/jpagefactory
- https://github.com/sergueik/SKDF
- https://github.com/dzharii/swd-recorder
- https://github.com/sergueik/SWET
- https://github.com/ealves/protractor-recorder
- https://github.com/hanthomas/protractor-recorder
- https://github.com/dzharii/swd-recorder
- https://github.com/sergueik/powershell_selenium_plugin