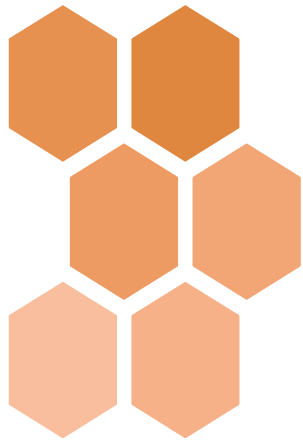


# Providing Protractor to Java / .Net

Serguei Kouzmine  
kouzmine\_serguei@yahoo.com



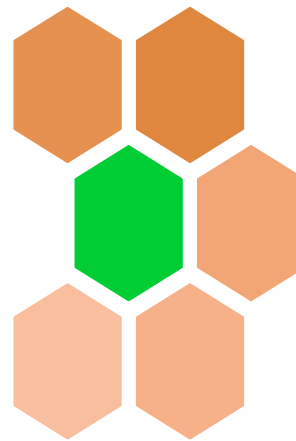
# Adding Key Client-side Protractor Methods to Java or .Net



Java



Node.js



jProractor,  
Protractor-  
net



# Projects Timeline

- [angular/protractor](#) Jan 2013, 247 contributors
- [bbaia/protractor-net](#) Oct 2013, 10 contributors
- [paul-hammant/ngWebDriver](#) Sep 2013, 6 contributors
- [greengerong/maven-ng-protractor](#) Mar 2014, 1 contributor
- [caarlos0/jProtractor](#) Jul 2014, 2 contributors
- [GRITAG/Protractor-jvm](#) Jun 2015, 2 contributors
- [henrrich/jpagefactory](#) May 2016, 2 contributors



# Angular Protractor in Javascript

Many of technologies considered part of Protractor Angular application "end to end" integration testing framework have same or better counterparts in Java and .Net stack outside of Selenium:

- mocha
- jasmine
- webdriverJs
- cucumber
- Junit / testNg
- Mockito
- Selenium / selenium Grid
- cucumber

From Selenium perspective, two outstanding features of Protractor are Angular semantics integration with locator and delegating the wait during element detection to Angular runtime.



# Bare-bones Angular page

```
<html ng-app="myApp">
<head>
<script src="angular.js"></script>
<script type="text/javascript">
var app = angular.module('myApp', []);
app.controller('myCtrl',
  function($scope){
    $scope.myChoice = 1;
    $scope.options = [
      {name: 'one', value: 1},
      {name: 'two', value: 2},
    ]
  });
</script>
</head>
```

```
<body ng-controller="myCtrl">
  <select ng-model="myChoice">
    <option ng-repeat="option in
options" value="{{option.value}}"
ng-selected=
  "option.value == myChoice"
>{{option.name}}</option>
  </select>
  <div>Value = {{myChoice}}</div>
</body>
</html>
```

Protractor 'client side' Javascript snippets injected in the browser enable test operate same language as page developer by offering element finder methods "by" binding, model, repeater etc. It also provides Angular agnostic methods like buttonText, cssContainingText



# Integration with QA / CI

- Framework (module) layers offered for integrating Selenium Testing with CI by Java, .Net and Node.js offer similar functionality but are not easy to combine (examples exist)
- Angular MVC Clientside API were only available through Protractor



# WebDriver Wire Protocol

Command	Method	URI Template
Find Element	POST	<i>/session/{session id}/element</i>
Find Element from Another	POST	<i>/session/{session id}/element/{element id}/element</i>
Get Element Attribute	GET	<i>/session/{session id}/element/{element id}/attribute/{name}</i>
Get Element Property	GET	<i>/session/{session id}/element/{element id}/property/{name}</i>
Get Element CSS Value	GET	<i>/session/{session id}/element/{element id}/css/{property name}</i>
Get Element Text	GET	<i>/session/{session id}/element/{element id}/text</i>
Get Element Tag Name	GET	<i>/session/{session id}/element/{element id}/name</i>
Get Element Rectangle position	GET	<i>/session/{session id}/element/{element id}/rect</i>
Execute Script	POST	<i>/session/{session id}/execute/sync</i>
Execute Async script	POST	<i>/session/{session id}/execute/async</i>

<https://w3c.github.io/webdriver/webdriver-spec.html#dfn-command>

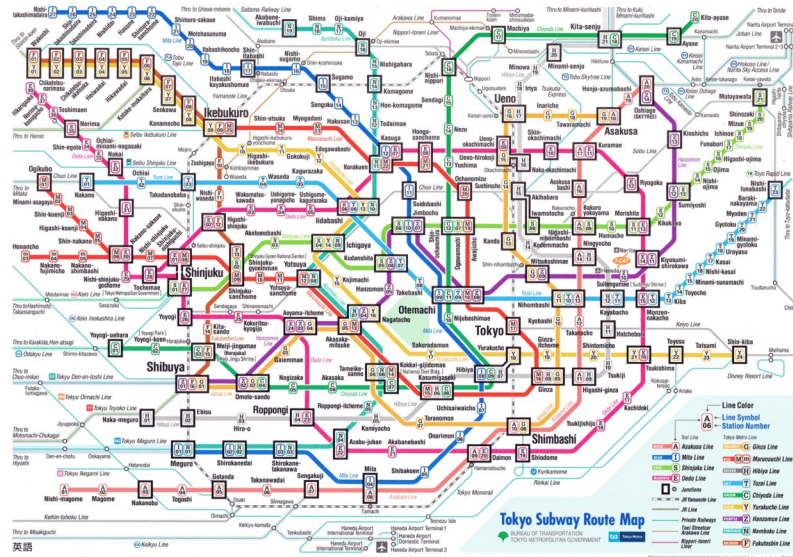
# Russian Nesting Doll Architecture

Java program launches Selenium to be the carrier of Javascript snippet that commands Protractor to find and return Page Elements





# Internal Design



Swisswatch complexity vs. Tokyo Subway map complexity



# Protract methods

Method
<code>model</code>
<code>binding</code>
<code>options</code>
<code>selectedOption, selectedRepeaterOption</code>
<code>repeater</code>
<code>RepeaterRow, repeaterColumn, repeaterElement</code>
<code>buttonText, partialButtonText</code>
<code>cssContainingText</code>
<code>testForAngular, waitForAngular</code>
<code>evaluate</code>



# WebDriver extended with Protractor

- ByClassName, ByCssSelector, ById, ByLinkText, ByName, ByPartialLinkText, ByTagName, ByXPath (nested classes of By)
- NgBy (JavaScriptBy): binding, buttonText, cssContainingText, model, options, repeater, repeaterRows, repeaterElement, selectedOption, selectedRepeaterOption
- ExpectedConditions
- WebElement.findElements
- WebElement.evaluate
- Wait
- WebDriver.waitForAngular
- SendKeys
- Actions



# Selenium JavascriptExecutor usage

Core Selenim 'By' classes support *a generic* DOM node therefore need annotated :

```
// finding selected customer
WebElement element = driver.findElement(By.xpath(
    "/div[13]/ul/li[4]/select"));

WebElement element = driver.findElement(By.cssSelector(
    "div#outer div.inner"));
```

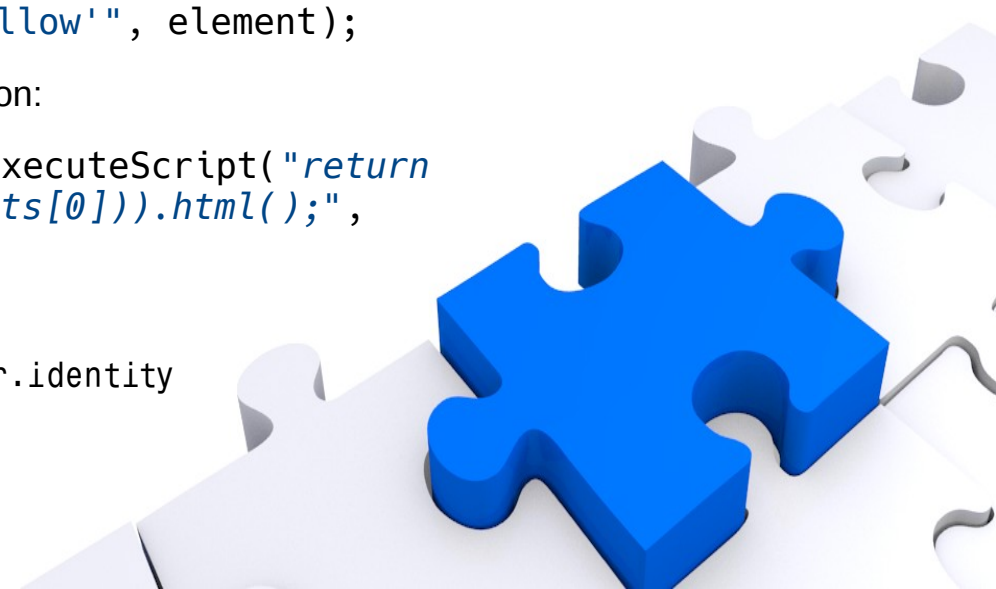
Javascript frequently used for visual cue:

```
// highlight
(JavascriptExecutor) driver.executeScript(
    "arguments[0].style.border='3px solid yellow'", element);
```

And can be equally used as well to return element information:

```
Object result = (JavascriptExecutor)driver.executeScript("return
    angular.identity(angular.element(arguments[0])).html();",
    element);
```

<https://docs.angularjs.org/api/ng/function/angular.identity>



# Convenience of Protractor Locators

Choosing the same language the page and the Test makes test more readable:  
locators reflect the context

```
NgWebElement cell = ngDriver.findElement(  
    NgBy.repeaterElement("row in rows", rowNum, "cellName"));
```

```
ngDriver.findElement(ByAngular.repeater("car in Cars")  
    .row(2)  
    .column("item.isSelected")).click();
```

```
List<WebElement> customers = ngDriver.findElements(NgBy.repeater("cust in  
Customers"));
```

```
NgWebElement selectedCustomer =  
ngDriver.findElement(NgBy.model("selectedCustomer"));
```

```
WebElement customerName = selectedCustomer.findElement(NgBy.binding("Name"));
```

```
Object customerCountry = selectedCustomer.evaluate("customer.Country");
```

```
ArrayList<Long> accounts = (ArrayList<Long>)  
    new NgWebElement(ngDriver, selectedCustomer).evaluate(  
"customer.accountNo");
```



# Sample Scenario

```
// When I proceed to customer login
NgWebElement login = ngDriver.findElement(
    NgBy.buttonText("Customer Login"));
highlight(login);
login.click();

// I find myself among existing customers
NgWebElement custId = ngDriver.findElement(NgBy.input("custId"));
highlight(custId);
custId.click();

WebElement customer = custId.findElements(
    NgBy.repeater("cust in Customers")).stream().filter(cust -> cust.getText()
        .containsString("Hermoine Granger"))
    .findFirst().get();

customer.sendKeys(Keys.SPACE); customer.click();
ngDriver.waitForAngular();
```



# Sample Scenario, contd.

```
// And I log in
login = ngDriver.findElement( NgBy.buttonText("Login"));
assertTrue(login.isEnabled());
highlight(login);
login.click();

// Then I am greeted by my name
assertThat("greeting", ngDriver.findElement( NgBy.binding("user")).getText(),
containsString("Hermoine Granger"));

// And I see balance on one of my accounts

NgWebElement accountNo = ngDriver.findElement(NgBy.binding("accountNo"));

assertThat("account number", accountNo.getText(),
anyOf(is("1001"), is("1002"), is("1003")));
highlight(accountNo);
```





# Sample Scenario, contd.

// And I open account

```
ngDriver.findElements(NgBy.options("account for account in Accounts"))  
.stream().filter(account -> account.getText().matches("1001"))  
.collect(Collectors.toList()).get(0).click();
```

// And I inspect transactions

```
NgWebElement transactionsButton = ngDriver  
.findElement(NgBy.partialButtonText("Transactions"));  
highlight(transactionsButton);  
transactionsButton.click();
```

```
wait.until(ExpectedConditions.visibilityOf(  
ngDriver.findElement(  
NgBy.repeater("tx in transactions")).getWrappedElement())));
```





# Sample Scenario, contd.

```
// Then for every transaction
```

```
for (WebElement tx : ngDriver.findElements(  
    NgBy.repeater("tx in transactions"))) {
```

```
    NgWebElement ngTx = new NgWebElement(ngDriver, tx);
```

```
    // it would be either Debit or Credit
```

```
    assertThat("transaction type", ngTx.evaluate("tx.type").toString(),  
        anyOf(containsString("Debit"), containsString("Credit")));
```

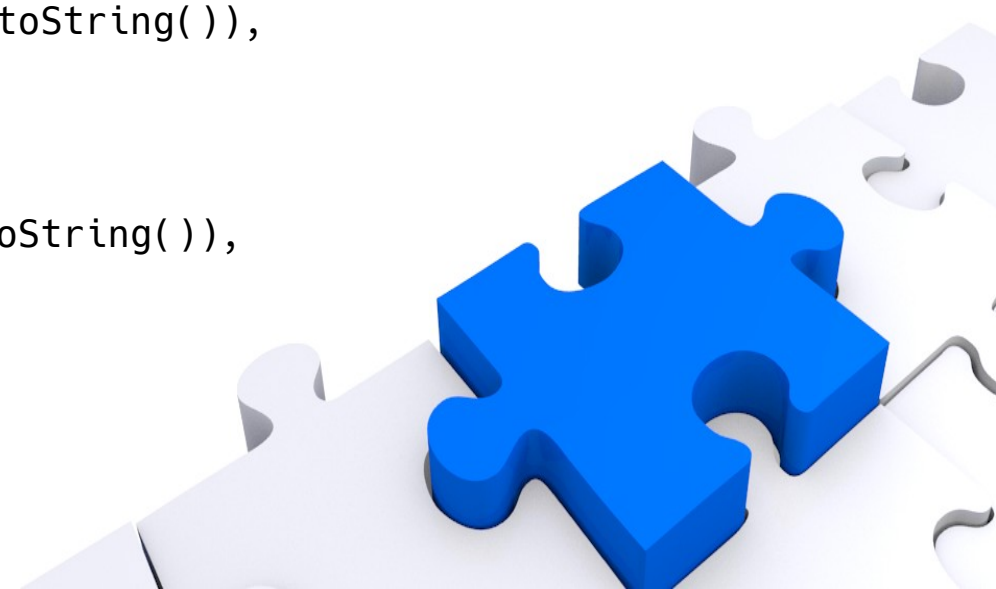
```
    // it will have non zero amount
```

```
    assertThat("it will have non zero amount",  
        Integer.parseInt(ngTx.evaluate("tx.amount").toString()),  
        greaterThan(0));
```

```
    // it date should be the past
```

```
    assertThat("transaction date",  
        dateFormat.parse(ngTx.evaluate("tx.date").toString()),  
        sameOrBefore(new Date()));
```

```
}
```



# Protractor and Cucumber

@test **Scenario Outline:** Existing Customer Login

**Given** I am at the Home Page

**When** I continue as "Customer Login"

**And** I am existing customer "<FirstName>", "<LastName>"

**Then** I am greeted by "<FirstName>", "<LastName>"

**And** I see balance on one of my accounts  
"<AccountNumbers>"

**And** I can switch to any of accounts "<AccountNumbers>"  
I have

**And** I can not see any other accounts...

**Examples:**

AccountNumbers	FirstName	LastName
1004,1005,1006	Harry	Potter



# Protractor Annotations

```
NgWebElement firstoperand = ngDriver.findElement(NgBy.model("first"));
```

```
NgWebElement element = ngDriver.findElement(NgBy.options(  
"value for (key, value) in operators"));
```

```
@FindBy(how = How.MODEL, using = "first")
```

```
private WebElement element;
```

```
@FindBy(how = How.OPTIONS, using = "value for (key, value) in  
operators")
```



# Clientsidescripts.js

18 methods

3 to > 100 lines of JavaScript / method

```
var element = arguments[0];  
var expression = arguments[1];  
return angular.element(element).scope().$eval(expression);
```

<https://github.com/angular/protractor/blob/master/lib/clientsidescripts.js#L623>



# Clientsidescripts.js

```
var findBindings = function(binding, exactMatch, using, rootSelector) {
  var root = document.querySelector(rootSelector || 'body');
  using = using || document;
  var bindings = using.getElementsByClassName('ng-binding');
  var matches = [];
  for (var i = 0; i < bindings.length; ++i) {
    var dataBinding = angular.element(bindings[i]).data('$binding');
    var bindingName = dataBinding.exp || dataBinding[0].exp || dataBinding;
    var matcher = new RegExp('{{|\\s|^|\\|}}' +
      binding.replace(/[\-\\[\]\/\{\}\(\)\*\+\?\\.\\\\^\$\\|]/g,
        '\\\$&') +
      '({|\\s|\\$|\\|})');
    if (matcher.test(bindingName)) {
      matches.push(bindings[i]);
    }
  }
  return matches; /* Return the array for webdriver.findElement. */
};

var using = arguments[0] || document;
var binding = arguments[1];
var rootSelector = arguments[2];
return findBindings(binding, using, rootSelector);
```



# Questions, Demos



# Resources

- <https://github.com/angular/protractor>
- <https://github.com/sergueik/jProtractor>
- <https://github.com/henrrich/jpagefactory>
- <https://github.com/paul-hammant/ngWebDriver>
- <https://github.com/bbaia/protractor-net>



# Resources

- <http://www.java2s.com/Tutorials/AngularJS/>
- <http://juliemr.github.io/protractor-demo/>
- <http://qualityshepherd.com/angular/friends/>
- <http://www.way2automation.com/angularjs-protractor/banking>
- <https://htmlpreview.github.io/?https://github.com/angular-ui/>
- <http://amitava82.github.io/angular-multiselect/>
- <http://www.codeproject.com/Articles/1066968/Developing-Protractor-tests-in-Csharp-or-Java>

