



An AngularJS testing framework for the rest of us:

porting Protractor to Python

Ed Manlove
QA Automation Supervisor
Dealertrack Technologies

Follow along at

github.com/emanlove/SeConf2015







Protractor

end to end testing for AngularJS



element(locator)

then

clone

locator

getWebElement

all

element

\$\$

\$

isPresent

isElementPresent

evaluate

allowAnimations

ExpectedConditions

not

and

or

alertIsPresent

elementToBeClickable

textToBePresentInElement

textToBePresentInElementValue

titleContains

titleIs

presenceOf

stalenessOf

visibilityOf

invisibilityOf

elementToBeSelected

locators

webdriver.Locator

checkLocator

toString

webdriver.Key.chord

webdriver.WebElementPromise

cancel

isPending

then

thenCatch

thenFinally

getId

webdriver.AlertPromise

cancel

isPending

then

thenCatch

thenFinally

getText

accept

dismiss

sendKeys

webdriver.UnhandledAlertError

getAlertText

getAlert

webdriver.FileDetector



locators

by.addLocator

by.binding

by.exactBinding

by.model

by.buttonText

by.partialButtonText

by.repeater

by.exactRepeater

by.cssContainingText

by.options

by.deepCss

browser

getProcessedConfig

forkNewDriverInstance

waitForAngular

findElement

findElements

isElementPresent

addMockModule

clearMockModules

removeMockModule

getRegisteredMockModules

get

refresh

navigate

setLocation

getLocationAbsUrl

debugger

enterRepl

pause



locators

by.addLocator

by.binding

by.exactBinding

by.model

by.buttonText

by.partialButtonText

by.repeater

by.exactRepeater

by.cssContainingText

by.options

by.deepCss

browser

getProcessedConfig

forkNewDriverInstance

waitForAngular

findElement

findElements

isElementPresent

addMockModule

clearMockModules

removeMockModule

getRegisteredMockModules

get

refresh

navigate

setLocation

getLocationAbsUrl

debugger

enterRepl

pause

locators

by.binding

by.model

by.repeater

browser

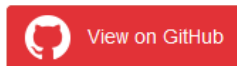
waitForAngular





Protractor

end to end testing for AngularJS



Protractor is an end-to-end test framework for AngularJS applications. Protractor runs tests against your application running in a real browser, interacting with it as a user would.

Test Like a User

Protractor is built on top of WebDriverJS, which uses native events and browser-specific drivers to interact with your application as a user would.

For AngularJS Apps

Protractor supports Angular-specific locator strategies, which allows you to test Angular-specific elements without any setup effort on your part.

Automatic Waiting

You no longer need to add waits and sleeps to your test. Protractor can automatically execute the next step in your test the moment the webpage finishes pending tasks, so you don't have to worry about waiting for your test and webpage to sync.



locators

by.binding

by.model

by.repeater

browser

waitForAngular

by Binding

```
functions.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');
        if (dataBinding) {
            var bindingName = dataBinding.exp || dataBinding[0].exp || dataBinding;
            if (exactMatch) {
                ...
            } else {
                if (bindingName.indexOf(binding) !== -1) {
                    matches.push(bindings[i]);
                }
            }
        }
    }
    return matches; /* Return the whole array for webdriver.findElements. */
};
```



by Binding

```
def _find_by_binding(self, browser, criteria, tag, constraints):
    return self._filter_elements(
        browser.execute_script(
            "var binding = '%s';" % criteria
            "var bindings = document.getElementsByClassName('ng-binding');"
            "var matches = [];"
            "for (var i = 0; i < bindings.length; ++i) {"
            "    var dataBinding = angular.element(bindings[i]).data('$binding');"
            "    if(dataBinding) {"
            "        var bindingName = dataBinding.exp || dataBinding[0].exp || dataBinding;"
            "        if (bindingName.indexOf(binding) != -1) {"
            "            matches.push(bindings[i]);"
            "        }"
            "    }"
            "}"
            "}"
            "return matches;"), tag, constraints)
```



by Binding



```
private String makeJsBy(String oneOrAll) {
    return
        "var using = arguments[0] || document;\n" +
        "var binding = '" + binding + "';\n" +
        "var bindings = using.getElementsByClassName('ng-binding');\n" +
        "var matches = [];\n" +
        "for (var i = 0; i < bindings.length; ++i) {\n" +
        "    var bindingName = angular.element(bindings[i]).data().$binding[0].exp ||\n" +
        "        angular.element(bindings[i]).data().$binding;\n" +
        "    if (bindingName.indexOf(binding) != -1) {\n" +
        "        matches.push(bindings[i]);\n" +
        "    }\n" +
        "}\n" +
        "return matches" + oneOrAll + ";;";
}
```



by Model

```
functions.findByModel = function(model, using, rootSelector) {
  var root = document.querySelector(rootSelector || 'body');
  using = using || document;
  ...
  var prefixes = ['ng-', 'ng_', 'data-ng-', 'x-ng-', 'ng\\\:'];
  for (var p = 0; p < prefixes.length; ++p) {
    var selector = '[' + prefixes[p] + 'model="' + model + '"']';
    var elements = using.querySelectorAll(selector);
    if (elements.length) {
      return elements;
    }
  }
};
```



by Model



```
def _find_by_model(self, browser, criteria, tag, constraints):
    prefixes = ['ng-', 'ng_', 'data-ng-', 'x-ng-', 'ng\\:']
    for prefix in prefixes:
        selector = '[%smodel="%s"]' % (prefix, criteria)
        elements = browser.execute_script("""return document.querySelectorAll('%s');""" % selector);
        if len(elements):
            return self._filter_elements(elements, tag, constraints)
```



angular.getTestability

```
functions.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');  
        if (dataBinding) {  
            var bindingName = dataBinding.exp || dataBinding[0].exp || dataBinding;  
            if (exactMatch) {  
                ...  
            } else {  
                if (bindingName.indexOf(binding) != -1) {  
                    matches.push(bindings[i]);  
                }  
            }  
        }  
    }  
    return matches; /* Return the whole array for webdriver.findElements. */  
};
```

angular.getTestability

```
functions.findBindings = function(binding, exactMatch, using, rootSelector) {
  var root = document.querySelector(rootSelector || 'body');
  using = using || document;
  if (angular.getTestability) {
    return angular.getTestability(root).
      findBindings(using, binding, exactMatch);
  }
  var bindings = using.getElementsByClassName('ng-binding');
  var matches = [];
  for (var i = 0; i < bindings.length; ++i) {
    var dataBinding = angular.element(bindings[i]).data('$binding');
    if (dataBinding) {
      var bindingName = dataBinding.exp || dataBinding[0].exp || dataBinding;
      if (exactMatch) {
        var matcher = new RegExp('{{|\\s|^|\\|}}' +
          /* See http://stackoverflow.com/q/3561711 */
          binding.replace(/[-\[\\]\/\{\}\|\(\)\*\+\?\.\^\$\\|]/g, '\\$&') +
          '{{|\\s|^|\\|}}');
        if (matcher.test(bindingName)) {
          matches.push(bindings[i]);
        }
      } else {
        if (bindingName.indexOf(binding) != -1) {

```

angular.getTestability

```
if (angular.getTestability) {  
    return angular.getTestability(root).  
        findBindings(using, binding, exactMatch);  
}
```



by Repeater

+ Add New Void ROS #										Set Notice Date Archive Download		
	▼ ROS #	Date Due	Customer	Deal ID	Stock #	VIN	Sale Date	Over / Under	Deal Status			
▶		06/25/2015	Test1C, QA			394964	06/07/2015		Saved			
▶		06/25/2015	Test3B, QA			394965	06/07/2015		Saved			
▶		06/25/2015	Test3C, QA		3388221	394926	06/07/2015		Saved			
▶		06/28/2015	GONZELES C	RR123	RR124	737C22		\$0.00	RDF			
▶		06/28/2015				821			Saved			
▶		06/28/2015	PAC BELL			516811		\$0.00	RDF			
▶		07/01/2015	Miller, David		JLC00001	881			Saved			
▶		06/29/2015	Smoke Level, Test1			387395	06/09/2015		Saved			
▶		06/29/2015	Smoke Level, Test1			387395	06/09/2015		Saved			
▶		06/29/2015	Smoke Level, T6 Auto			147038	06/09/2015		Saved			

by Repeater

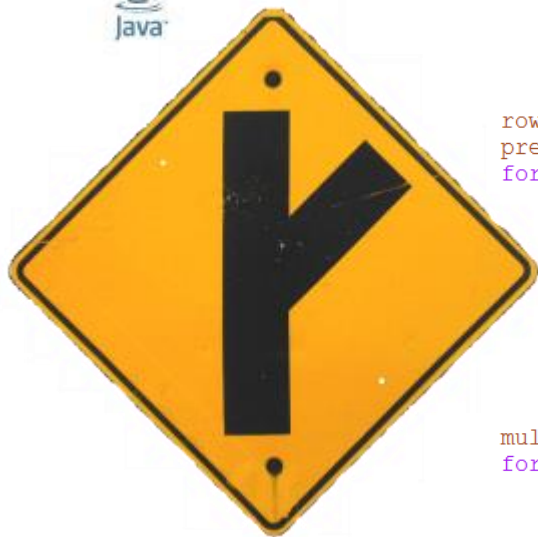
```
function findRepeaterRows(repeater, exact, index, using) {
    using = using || document;

    var prefixes = ['ng-', 'ng_', 'data-ng-', 'x-ng-', 'ng\\:'];
    var rows = [];
    for (var p = 0; p < prefixes.length; ++p) {
        var attr = prefixes[p] + 'repeat';
        var repeatElems = using.querySelectorAll(['' + attr + '']);
        attr = attr.replace(/\\/g, '');
        for (var i = 0; i < repeatElems.length; ++i) {
            if (repeaterMatch(repeatElems[i].getAttribute(attr), repeater, exact)) {
                rows.push(repeatElems[i]);
            }
        }
    }
    /* multiRows is an array of arrays, where each inner array contains
       one row of elements. */
    var multiRows = [];
    for (var p = 0; p < prefixes.length; ++p) {
        var attr = prefixes[p] + 'repeat-start';
        var repeatElems = using.querySelectorAll(['' + attr + '']);
        attr = attr.replace(/\\/g, '');
        for (var i = 0; i < repeatElems.length; ++i) {
            if (repeaterMatch(repeatElems[i].getAttribute(attr), repeater, exact)) {
                var elem = repeatElems[i];
                var row = [];
                while (elem.nodeType != 8 ||
                    !repeaterMatch(elem.nodeValue, repeater, exact)) {
                    if (elem.nodeType == 1) {
                        row.push(elem);
                    }
                }
            }
        }
    }
}
```





by Repeater



```
def _find_by_ng_repeater(self, browser, criteria, tag, constraints):
    matches=[]
    repeater_row_col = self._parse_ng_repeater_locator(criteria)
```

```
import collections
def get_iterable(x):
    if isinstance(x, collections.Iterable):
        return x
    else:
        return [x]
```

```
rows = []
prefixes = ['ng-', 'ng_', 'data-ng-', 'x-ng-', 'ng\\\\\\\\:']
for prefix in prefixes:
    repeatElems=[]
    attr = prefix + 'repeat'
    repeatElems = browser.execute_script("""return document.querySelectorAll('[%s]');""") % attr
    attr = attr.replace('\\\\', '\\')
    repeatElems = get_iterable(repeatElems)
    for elem in repeatElems:
        val = elem.get_attribute(attr)
        if val and repeater_row_col['repeater'] in elem.get_attribute(attr):
            rows.append(elem)

multiRows = []
for prefix in prefixes:
    attr = prefix + 'repeat-start'
    repeatElems = browser.execute_script("""return document.querySelectorAll('[%s]');""") % attr
    attr = attr.replace('\\\\', '\\')
    for elem in repeatElems:
```







by Repeater



```
def _find_by_ng_repeater(self, browser, criteria, tag, constraints):
    matches=[]
    repeater_row_col = self._parse_ng_repeater_locator(criteria)

    import collections
    def get_iterable(x):
        if isinstance(x, collections.Iterable):
            return x
        else:
            return [x]

    rows = []
    prefixes = ['ng-', 'ng_', 'data-ng-', 'x-ng-', 'ng\\\\:']
    for prefix in prefixes:
        repeatElems=[]
        attr = prefix + 'repeat'
        repeatElems = browser.execute_script("""return document.querySelectorAll('[%s]');""" % attr)
        attr = attr.replace('\\\\', '\\')
        repeatElems = get_iterable(repeatElems)
        for elem in repeatElems:
            val = elem.get_attribute(attr)
            if val and repeater_row_col['repeater'] in elem.get_attribute(attr):
                rows.append(elem)

    multiRows = []
    for prefix in prefixes:
        attr = prefix + 'repeat-start'
        repeatElems = browser.execute_script("""return document.querySelectorAll('[%s]');""" % attr)
        attr = attr.replace('\\\\', '\\')
        for elem in repeatElems:
```

Testing the Testing Framework

```
*** Settings ***
Test Setup      Start Angular Test
Resource        ../resource.txt

*** Test Cases ***
Should Find An Element By Binding
    ${text} = Get Text  {{greeting}}
    Should Be Equal  '${text}'  'Hiya'

Should Find A Binding By Partial Match
    ${text} = Get Text  binding=greet
    Should Be Equal  '${text}'  'Hiya'

Should Find An Element By Binding With Ng-Bind ?
    ${text} = Get Text  binding=username
    Should Be Equal  '${text}'  'Anon'

Should Find An Element By Binding With Ng-Bind-?
    ${text} = Get Text  {{nickname|uppercase}}
    Should Be Equal  '${text}'  '(ANNIE)'

Should Find An Element By Text Input Model
    Clear Text  model=username
    ${text} = Get Text  binding=username
    Should Be Equal  '${text}'  ''

    Input Text  model=username  Jane Doe
    ${text} = Get Text  binding=username
    Should Be Equal  '${text}'  'Jane Doe'
```



locators

by.binding

by.model

by.repeater

element find

xpath

`//*[@ng-click="openMenu()"]`



locators

by.binding

by.model

by.repeater

element find

xpath `//*[@ng-click="openMenu()"]`

model

mutate / retrieve

Model mutation and query

```
public void mutate(WebElement element, final String variable, final String value) {  
    driver.executeScript("angular.element(arguments[0]).scope()." + variable + " = " + value + ";" +  
        "angular.element(document.body).injector().get('$rootScope').$apply();", element);  
}  
  
public Object retrieve(WebElement element, final String variable) {  
    return check(variable, driver.executeScript(  
        "return angular.element(arguments[0]).scope()." + variable + ";", element));  
}
```





locators

by.binding

by.model

by.repeater

element find

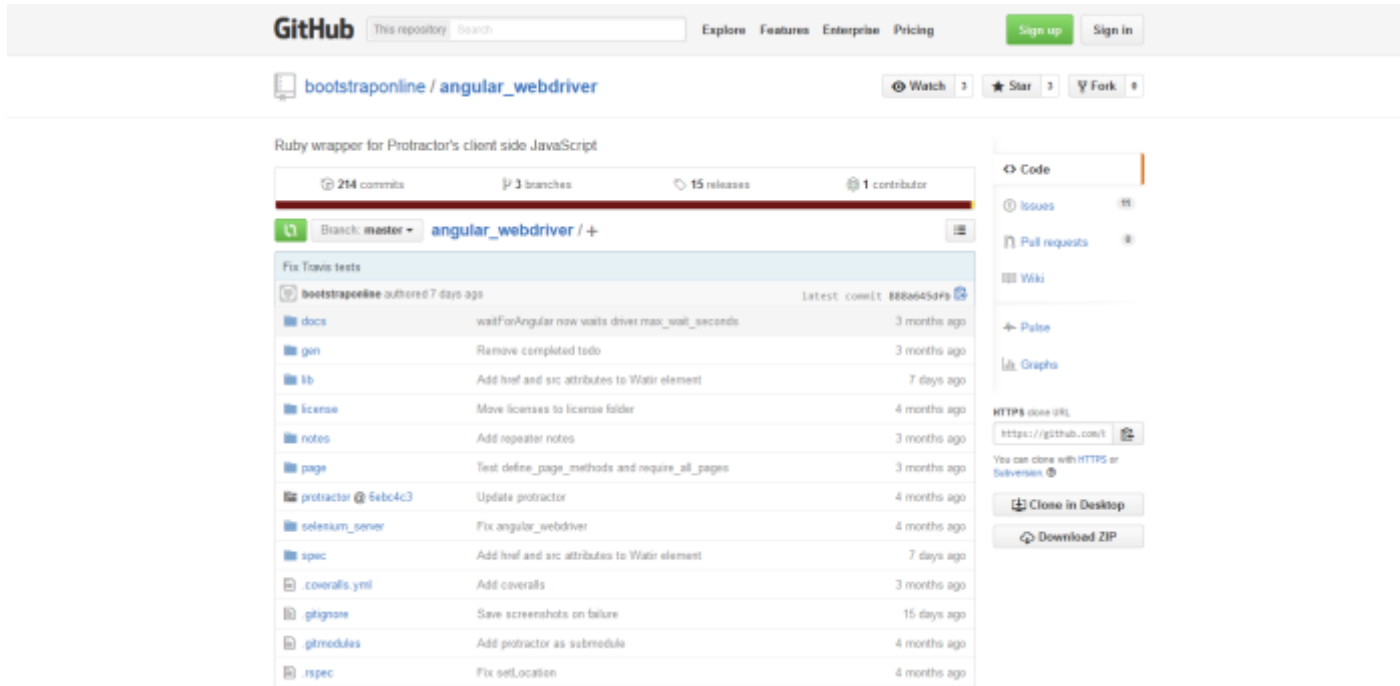
xpath `//*[@ng-click="openMenu()"]`

model

mutate / retrieve

Ruby -

<https://github.com/bootstraponline>



The screenshot shows the GitHub repository page for `bootstraponline / angular_webdriver`. The repository is a Ruby wrapper for Protractor's client side JavaScript. It has 214 commits, 3 branches, 15 releases, and 1 contributor. The current branch is `master`. The repository is public and has 3 stars and 0 forks.

The commit history table is as follows:

Commit	Message	Time
<code>bootstraponline</code>	outdated 7 days ago	latest commit: <code>888a645d9a</code>
<code>docs</code>	waitForAngular now waits driver.max_wait_seconds	3 months ago
<code>gin</code>	Remove completed todo	3 months ago
<code>lib</code>	Add href and src attributes to Water element	7 days ago
<code>license</code>	Move licenses to license folder	4 months ago
<code>notes</code>	Add repeater notes	3 months ago
<code>page</code>	Test define_page_methods and require_all_pages	3 months ago
<code>protractor @ 5ebc4c3</code>	Update protractor	4 months ago
<code>selenium_server</code>	Fix angular_webdriver	4 months ago
<code>spec</code>	Add href and src attributes to Water element	7 days ago
<code>.coveralls.yml</code>	Add coveralls	3 months ago
<code>gitignore</code>	Save screenshots on failure	15 days ago
<code>gitmodules</code>	Add protractor as submodule	4 months ago
<code>.rspec</code>	Fix setLocation	4 months ago

On the right side, there are links for `Code`, `Issues`, `Pull requests`, `Wiki`, `Pages`, and `Graphs`. Below these links, there is a section for cloning the repository with HTTPS or SSH, and buttons for `Clone in Desktop` and `Download ZIP`.

browser

waitForAngular



waitForAngular

```
public static void waitForAngularRequestsToFinish(JavascriptExecutor driver) {  
    driver.executeAsyncScript("var callback = arguments[arguments.length - 1];" +  
        "angular.element(document.body).injector().get('$browser').  
        notifyWhenNoOutstandingRequests(callback);");  
}
```



waitForAngular

```
def wait_for_angular(self, timeout=None, error=None):
    """
    """

    if not error:
        error = "Wait For Angular did not become true in <TIMEOUT>"

    script = """
    var el = document.querySelector('body');
    try {
        angular.element(el).injector().get('$browser').
            notifyWhenNoOutstandingRequests();
    } catch (e) {
        return true;
    }

    return false;
    """

    self._wait_until(timeout, error,
                     lambda: self._current_browser().execute_script(script) == True)
```



waitForAngular

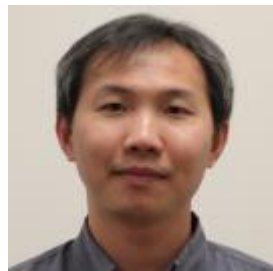
```

NG_WRAPPER = '%(prefix)s' \
              'angular.element(document.querySelector(\'[data-ng-app]\') || document).\' \
              'injector().get(\'$browser\').notifyWhenNoOutstandingRequests(%(handler)s)' \
              '%(suffix)s'

def wait_until_angular_ready(self, timeout=None, error=None):
    ...
    script = self.NG_WRAPPER % {'prefix': 'var cb=arguments[arguments.length-1];',
                                'if(window.angular){',
                                'handler': 'function(){cb(true)}',
                                'suffix': '}' else {cb(true)}}

    browser = self._current_browser()
    browser.set_script_timeout(timeout)
    try:
        WebDriverWait(browser, timeout, self._poll_frequency).\
            until(lambda driver: driver.execute_async_script(script), error)
    except TimeoutException:
        # prevent double wait
        pass
    except:
        # still inflight, second chance. let the browser take a deep breath...
        sleep(self._browser_breath_delay)
    ...
    finally:
        browser.set_script_timeout(self._timeout_in_secs)

```



waitForAngular

```
functions.waitForAngular = function(rootSelector, callback) {
  var el = document.querySelector(rootSelector);

  try {
    if (!window.angular) {
      throw new Error('angular could not be found on the window');
    }
    if (angular.getTestability) {
      angular.getTestability(el).whenStable(callback);
    } else {
      if (!angular.element(el).injector()) {
        throw new Error('root element (' + rootSelector + ') has no injector.' +
          ' this may mean it is not inside ng-app.');
```



waitForAngular

```
ElementArrayFinder.prototype.all = function(locator) {  
  var self = this;  
  var ptor = this.ptor_  
  var getWebElements = function() {  
    if (self.getWebElements === null) {  
      // This is the first time we are looking for an element  
      return ptor.waitForAngular().then(function() {  
        if (locator.findElementsOverride) {  
          return locator.findElementsOverride(ptor.driver, null, ptor.rootEl);  
        } else {  
          return ptor.driver.findElements(locator);  
        }  
      });  
    } else {  
      return self.getWebElements().then(function(parentWebElements) {
```

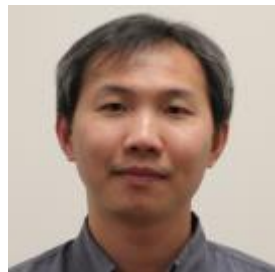
waitForAngular

```
def click_button(self, locator):
    self._scroll_into_view(locator)
    super(ExtendedSelenium2Library, self).click_button(locator)
    self._wait_until_page_ready()
    self.wait_until_angular_ready()

def click_element(self, locator):
    self._scroll_into_view(locator)
    super(ExtendedSelenium2Library, self).click_element(locator)
    self._wait_until_page_ready()
    self.wait_until_angular_ready()

def click_element_at_coordinates(self, locator, xoffset, yoffset):
    self._scroll_into_view(locator)
    super(ExtendedSelenium2Library, self). \
        click_element_at_coordinates(locator, xoffset, yoffset)
    self._wait_until_page_ready()
    self.wait_until_angular_ready()

def click_image(self, locator):
    self._scroll_into_view(locator)
    super(ExtendedSelenium2Library, self).click_image(locator)
    self._wait_until_page_ready()
    self.wait_until_angular_ready()
```



Al Scott, Ken Fox, et al. @ Atomic Object

A screenshot of a blog post from Atomic Object. The post is titled "Testing Asynchronous Behavior in JavaScript with Selenium" and is written by Al Scott, dated May 12, 2015. The author's bio states he is a developer at Atomic Object in Ann Arbor, spends most of his time doing full stack web development, and enjoys board and video games, hockey, and cats. The post content discusses Selenium for testing web applications, noting its limitations with asynchronous JavaScript and mentioning a "Flawed Workaround" involving "sleeps". A sidebar on the right contains navigation links like "Services", "Credentials", "Portfolio", "Approach", "Culture", "Blog", "Contact", and "We're Hiring". A red banner at the bottom right says "We're hiring software developers. learn more >".

ATOMIC OBJECT

May 12, 2015 by Al Scott

Testing Asynchronous Behavior in JavaScript with Selenium

Full stack browser testing of web applications is awesome. It validates that your application works end-to-end and allows you to check actual user workflows. For the last year and a half, I've been using Selenium to test a JavaScript web application we're developing using Backbone.js.

Unfortunately, web apps that use JavaScript a lot can be challenging to test with Selenium.

Selenium was designed for testing traditional websites and isn't well equipped out of the box to handle asynchronous behavior in JavaScript apps that might change the page. Even something simple like looking for an element that's put on the page in response to an AJAX request will fail with Selenium because it has no idea about the internals of your web app. Selenium just knows that the content it's looking for is not on the page *right now*.

A Flawed Workaround

The poor man's solution to this is just to add lots of sleeps is

< Atomic Spin

- Services
- Credentials
- Portfolio
- Approach
- Culture
- Blog
- Contact

We're Hiring

We're hiring software developers. [learn more >](#)



Future Development

- wrap WebDriver functions/methods/... with waitForAngular
- error reporting on waitForAngular failures
- Ruby, dotNet,...
- Protractor debugger???
- Stop Animations



Community, Community, Community



Thank You



Ed Manlove
QA Automation Supervisor
Dealertrack Technologies



Acknowledgements

Road signs courtesy of Jill C. and Donald Knuth

Python Logo used under the PSF Trademark Usage Policy

Java and “Java Powered” logo used under the Java Licensing Logo Guidelines