


GEB FOR BROWSER AUTOMATION

Jacob Aae Mikkelsen

AGENDA

- Functional testing
- Geb - cudos and history
- How Geb works
- Geb and Ratpack, Grails 2 and 3
- Browser support
- Tips and Tricks

JACOB AAE MIKKELSEN

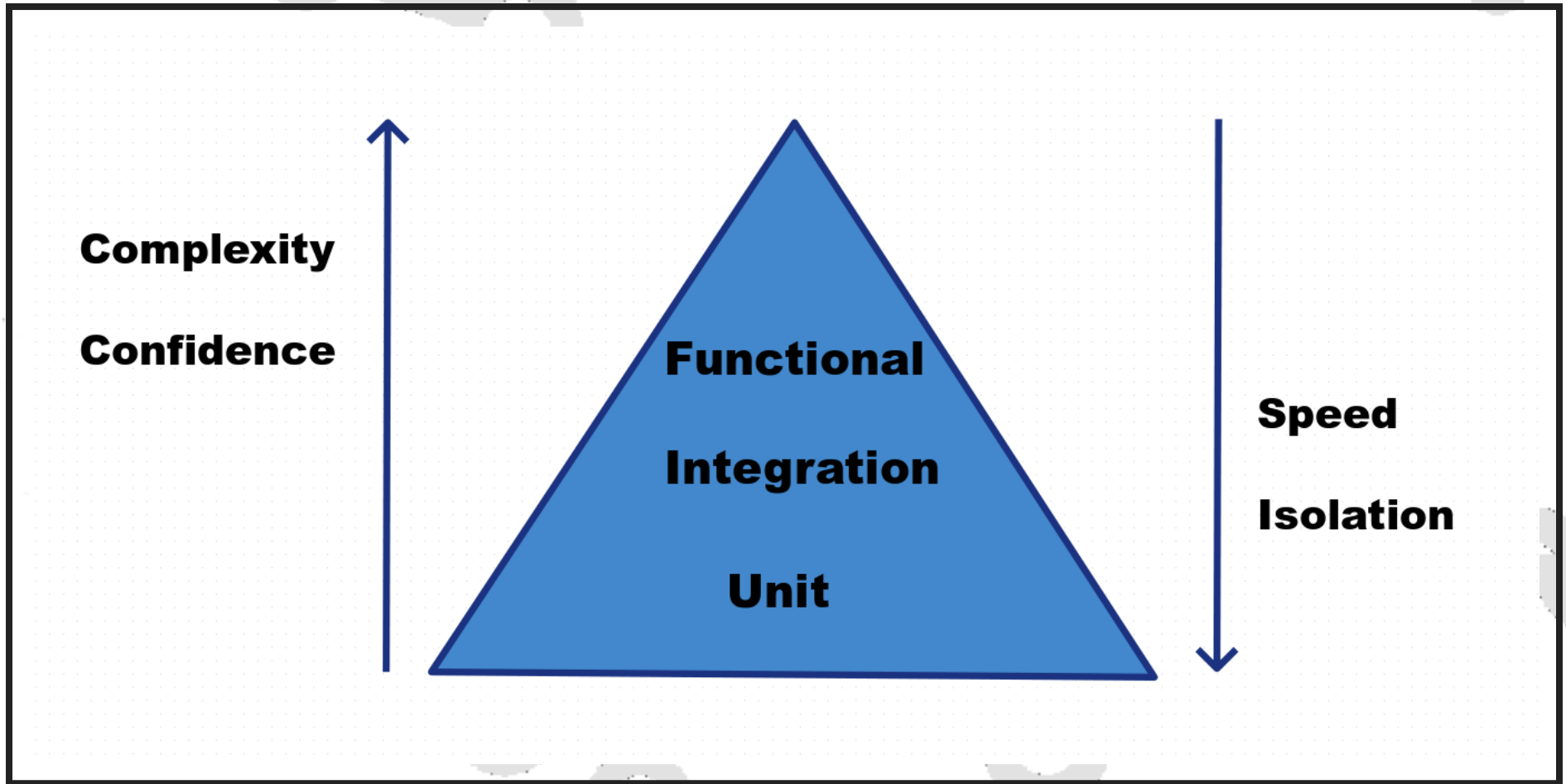
- Senior Engineer at LEGO
 - Microservice based architecture on JVM
- Previously 4 years at Gennemtænkt IT
 - Consultant on Groovy and Grails
- External Associate Professor - University of Southern Denmark
-  @JacobAae
- Blogs [The Grails Diary](#)



FUNCTIONAL TESTING



WHY?

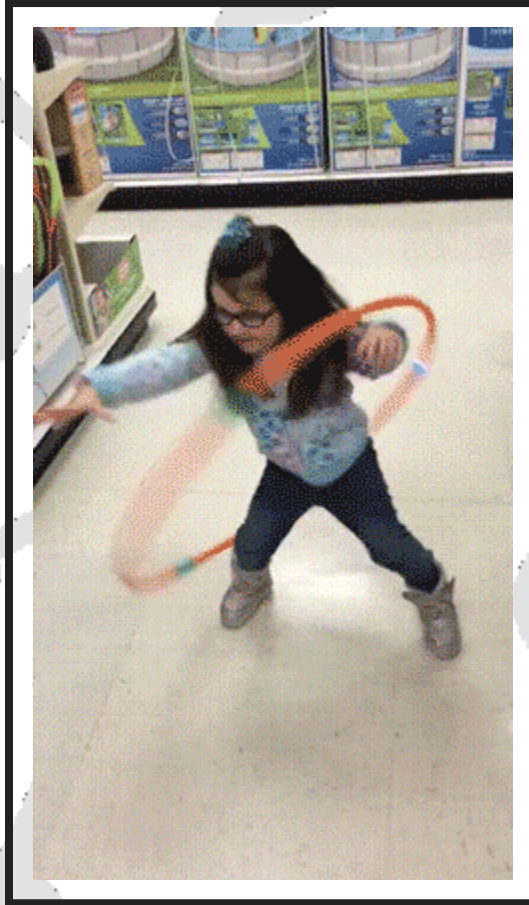




FUNCTIONAL TESTING

- Ignores the specifics of the underlying software component under test.
 - Whitebox / Greybox
- Merely asserts that providing certain input results in certain output.
- **Web-application:** Programmatically controlling a web browser to simulate the actions of a user on a web page.

BROWSER AUTOMATION





GEB HISTORY

- Started in November 2009
- Created by Luke Daley
- Current project lead Marcin Erdman

GEB IMPLEMENTATION

- Build on top of the WebDriver browser automation library
 - successor to the Selenium Remote Control (RC) testing framework.
 - Selenium RC → JavaScript to interact
 - WebDriver → native browser drivers
- Use JUnit or Spock

WEBDRIVER

- Very active development
- Stable API and feature set
- Verbose
- Low level
- Not a complete solution

WHY GEB

- jQuery like selector syntax
- Power of WebDriver (Easier api)
- Robustness of Page Object modeling
- Expressiveness of the Groovy language
- Integrates well with build systems (Gradle/Maven)
- Excellent user manual/documentation

Test Framework
Spock - JUnit - Cucumber etc.



Geb



WebDriver



Browser



Application



USING GEB

NAVIGATOR

The `$ ()` method returns a Navigator object

General format - JQuery like selector syntax

```
$( <css selector>, <index/range>, <attribute/text matchers> )
```

GEB SELECTORS (1)

```
// match all 'p' elements on page  
$("p")
```

```
// match the first 'p' element on the page  
$("p", 0)
```

```
// All 'div' elements with a title value 'section'  
$("div", title: "section")
```

```
// match the first 'div' element text 'section'  
$("div", 0, text: "section")
```

```
// match the first 'div' element with the class 'main'  
$("div.main", 0)
```

GEB SELECTORS (2)

Text attribute supports regex

```
// Any div with the text starting with GR8  
$("div", text: ~/GR8.+/)  
$("p", text: startsWith("GR8"))
```

```
// And other handy predicates  
$("div", class: contains("ui-"))
```

GEB SELECTORS (3)

Selecting returns Navigator objects

```
// The parent of the first div  
$("div", 0).parent()
```

```
// All tables with a cellspacing  
// attribute value of 0 that are nested in a paragraph  
$("p").find("table", cellspacing: '0')
```

CSS SUPPORT

```
$("table tr:nth-child(2n+1) td")
```

RETRIVING INFORMATION

```
<p id="sample" class="class-a class-b" title="Sample p element">  
Sample text  
</p>
```

```
$("p").text() == "Sample text"  
$("#sample").tag() == "p"  
$("p").@title == "Sample p element"  
$("p").classes() == ["class-a", "class-b"]
```

INTERACTION WITH CONTENT

- `click()`
- `isDisplayed()`
- `withConfirm{}`
- `withAlert{}`

```
$("a.btn").click()  
$("div").isDisplayed()  
  
withConfirm {  
    $("button.delete").click()  
}
```

SENDING INPUT

```
import org.openqa.selenium.Keys

// Shorthand for sendKeys() method of WebDriver.
$("div") << "abc"

$("input", name: "foo") << Keys.chord(Keys.CONTROL, "c")
```




INTERACTION

Using `Actions` API from `WebDriver` is possible.

But `Geb` provides the `interact` closure

CONTROL-CLICKING

```
import org.openqa.selenium.Keys

interact {
    keyDown Keys.CTRL
    click $("a.myLink")
    keyUp Keys.CTRL
}
```

SIMULATE DRAG-N-DROP

```
interact {  
  clickAndHold($('#draggable'))  
  moveByOffset(150, 200)  
  release()  
}
```

Or easier

```
interact {  
  dragAndDropBy($('#draggable'), 150, 200)  
}
```

MORE INTERACTION WITH FORMS

```
<form>  
  <input type="text" name="geb" value="Functional" />  
</form>
```

The value can be read and written via property notation...

```
$("form").geb == "Functional"  
$("form").geb = "Testing"
```

These are literally shortcuts for...

```
$("form").find("input", name: "geb").value() == "Functional"  
$("form").find("input", name: "geb").value("Testing")
```

VARIABLES AVAILABLE

- `title`
- `browser`
- `currentUrl`
- `currentWindow`

MORE POSSIBILITIES

- Uploading files
- Downloading files
- Interacting with javascript
 - js object (Example later)



STANDALONE GEB SCRIPT

GEB STANDALONE EXAMPLE

Lets try to automate:

- Searching for Greach Conference
- Click the first link
- Hopefully end up on the right homepage

GEB STANDALONE EXAMPLE

```
go "http://duckduckgo.com"

$('input', name: 'q').value("Greach Conference")
$('input', name: 'q') << Keys.ENTER

waitFor(10, 1) { $("#links").displayed }
    sleep(3000) // For demo reasons

$("#h2.result__title").first().click()

waitFor { title.startsWith "Greach" }
```



STRUCTURING GEB TESTS

SCENARIO

Lets test a small todo application

Lets test the following

1. Goto list of todos
2. Create new item
3. Delete item again

GEB SPEC BASICS

```
import geb.spock.GebSpec

@Stepwise // Ensures the tests are run sequentially
class TodoSpec extends GebSpec {

    // Spock specs here
}
```

GEB SPEC (1)

The naive inmaintainable way!

```
def "Go to index page"() {  
  when: 'Go to index url'  
  go '/'  
  
  then: 'Verify we are there'  
  title == "Todo List"  
}
```

GEB SPEC (2)

The naive inmaintainable way!

```
def "Create new todo"() {  
  when: 'Input text and submit'  
  $("#new-todo") << "Do this"  
  $("#create-btn").click()  
  
  then: 'Verify new item present in list'  
  waitFor { $("#count").text() == '4' }  
  $('li.todo-item').any { it.text().contains 'Do this' }  
  
  and: 'Verify input field empty'  
  !$("#new-todo").text()  
}
```

GEB SPEC (3)

The naive inmaintainable way!

```
def "Delete todo item"() {  
  when: 'Click delete and accept'  
  withConfirm {  
    $('button', 4).click()  
  }  
  
  then: 'Verify item deleted'  
  waitFor { $("#count").text() == '3' }  
  $('li.todo-item').every{ !( it.text().contains('Do this')) }  
}
```

GEB SPEC - THE BETTER WAY

If we make a few scenarios, there will be

- Much duplication
- Many places to correct if we change the layout / DOM

File Edit View Insert Format Window Help
Invert Selection Layout Colors Data Spy Test Favorites
[Icons]

Code

```
fnlsdnakdpajsdna psod; 'a+ '138  
}}}}}} 546r7t67t7  
{}}}  
[[[9099
```

```
ijungu1fn alfsldth suif sfhsui  
ia ladgh ajksdcd :043 u ss sdf  
asud  
aodyh1 a hda: a8ytaugf
```



SOLUTION



Use *pages* and *modules*

PAGE OBJECTS

Describes a web page

- Url
- How to check if we are at the correct place
- Content we wish to interact with
 - .. and how it is found
- Helper methods

PAGE OBJECTS

```
package net.grydeske.greach.pages

import geb.Page

class AboutPage extends Page {

    static url = "/about"

    static at = {
        title == "About"
    }

    static content = {
        header { $('h1', 0) }
    }
}
```

CONTENT CLOSURE

```
static content = {  
  info(required: false) { $("div.info") }  
  message(wait: false) { $("div.message") }  
}
```

MODULES

Describes repeated content

- Across pages
- Repeated content within the same page

MODULES ACROSS PAGES

```
package net.grydeske.greach.modules

import geb.Module

class MenubarModule extends Module {

    static base = { $("nav.navbar") }

    static content = {
        home { $('a', text: 'Todo List') }
        about { $('a', text: 'About') }
    }

}
```

MODULES FOR REPEATED CONTENT

```
package net.grydeske.greach.modules

import geb.Module

class TodoItemModule extends Module {

    static content = {
        checkbox { $('input', type: 'checkbox') }
        label { $('label').text() }
        deleteBtn { $('button') }
    }
}
```


USING MODULES

```
static content = {  
  menubar { module MenubarModule }  
  todos(required: false) {  
    $('li.todo-item').moduleList(TodoItemModule)  
  }  
}
```

A faint, light gray background pattern consisting of a network of interconnected circles and lines, resembling a molecular structure or a complex graph. The circles vary in size and are connected by thin lines, creating a web-like structure across the entire slide.

GEB SPEC - STRUCTURED

Lets try to restructure the ugly spec from before

GEB SPEC - STRUCTURED (1)

```
def "Go to index page"() {  
  when: 'Go to index url'  
  to IndexPage  
  
  then: 'Verify 3 items present'  
  at IndexPage  
  countValue == '3'  
}
```

GEB SPEC - STRUCTURED (2)

```
def "Create new todo"() {  
  when: 'Input text and submit'  
  todoInput = "Do this"  
  todoSubmit.click()  
  
  then: 'Verify new item present in list'  
  waitFor { countValue == '4' }  
  todos.any{ it.label == 'Do this' }  
  
  and: 'Verify input field empty'  
  !todoInput.text()  
}
```

GEB SPEC - STRUCTURED (3)

```
def "Delete todo item"() {  
  when: 'Click delete and accept'  
  withConfirm {  
    todos.find{ it.label == 'Do this' }.deleteBtn.click()  
  }  
  
  then: 'Verify item deleted'  
  waitFor {countValue == '3'}  
  todos.every{ it.label != 'Do this' }  
}
```

STANDALONE REVISITED

```
class DuckDuckGoPage extends geb.Page {  
  
  static url = "http://duckduckgo.com"  
  
  static at = { title ==~ /DuckDuckGo/ }  
  
  static content = {  
    inputField{ $('input', name: 'q') }  
  }  
  
  def submit() {  
    inputField << Keys.ENTER  
  }  
}
```

STANDALONE REVISITED

```
class DuckDuckGoResultPage extends geb.Page {  
  
  static url = "http://duckduckgo.com"  
  
  static at = { $("#links").displayed }  
  
  static content = {  
    links{ $("#h2.result__title") }  
  }  
  
  def clickLink(int linkNumber) {  
    links[linkNumber].click()  
  }  
  
}
```

STANDALONE REVISITED

```
class GreachPage extends geb.Page {  
    static at = { title.startsWith("Greach") }  
}
```


STANDALONE REVISITED

```
browser.with {  
  to DuckDuckGoPage  
  
  inputField << "Greach Conference"  
  submit()  
  
  waitFor(10, 0.5) {  
    at DuckDuckGoResultPage  
  }  
  
  sleep(3000) // For demo reasons  
  
  clickLink(0)  
  
  waitFor {  
    at GreachPage  
  }  
}
```



GEB WITH RATPACK AND GRAILS

GEB WITH RATPACK

```
dependencies {  
    ...  
    // If using Spock, need to depend on geb-spock  
    testCompile "org.gebish:geb-spock:0.13.1"  
    testCompile "org.spockframework:spock-core:1.0-groovy-2.4"  
  
    // Need a driver implementation  
    testCompile(  
        "org.seleniumhq.selenium:selenium-firefox-driver:2.53.0"  
    )  
    testRuntime(  
        "org.seleniumhq.selenium:selenium-support:2.53.0"  
    )  
}
```

GEB WITH RATPACK

```
@Shared
def aut = new GroovyRatpackMainApplicationUnderTest()

def setup() {
    URI base = aut.address
    browser.baseUrl = base.toString()
}
```

INTERACTING WITH RATPACK APPLICATION

When some functionality is needed that is not exposed through the browser, it can be necessary to interact with the application under test.

INTERACTING WITH RATPACK

Include in build.gradle

```
dependencies {  
    ...  
    compile ratpack.dependency("remote")  
    testCompile ratpack.dependency("remote-test")  
    ...  
}
```

INTERACTING WITH RATPACK

```
@Shared
def aut = new GroovyRatpackMainApplicationUnderTest()

RemoteControl remoteControl

def setup() {
    URI base = aut.address
    browser.baseUrl = base.toString()
    remoteControl = new RemoteControl(aut)
}
```

INTERACTING WITH RATPACK

```
def "Complete item and check database"() {  
  when: 'Check item done'  
  todos.find{ it.label == 'Give Geb presentation' }  
    .checkbox.click()  
  
  then: 'Use remote control to check database'  
  remoteControl.exec {  
    Boolean completed  
    TodoItem.withNewSession {  
      TodoItem todoItem = TodoItem  
        .findByText('Give Geb presentation')  
      completed = todoItem.completed  
    }  
    completed  
  } == true  
}
```


GEB AND GRAILS 2.X

Must install plugin in BuildConfig.groovy

```
dependencies {  
    ...  
    test("org.seleniumhq.selenium:selenium-support:2.53.0")  
    test("org.seleniumhq.selenium:selenium-firefox-driver:2.53.0")  
    test "org.gebish:geb-spock:0.12.1"  
}  
plugins {  
    ...  
    test "org.grails.plugins:geb:0.12.1"  
}
```

GEB TESTS IN GRAILS 2.X

Tests placed in `test/functional` folder

Running the tests

```
grails test-app functional:
```

GEB AND GRAILS 3

Geb is default in build.gradle

```
dependencies {  
    ...  
    testCompile "org.grails.plugins:geb"  
  
    testRuntime(  
        "org.seleniumhq.selenium:selenium-htmlunit-driver:2.47.1"  
    )  
    testRuntime "net.sourceforge.htmlunit:htmlunit:2.18"  
}
```

GEB TESTS IN GRAILS 3

Creating Geb Spec

```
grails create-functional-test MyGebScenario
```

Placing the test in `src/integration-test/groovy`

Running the tests

```
grails test-app -integration
```

GENERATED CLASS

```
@Integration
@Rollback
class DemoSpec extends GebSpec {

    void "test something"() {
        when:"The home page is visited"
            go '/'

        then:"The title is correct"
            $('title').text() == "Welcome to Grails"
    }
}
```

INTERACTING WITH GRAILS 2.5

Tests not running in same JVM

Done with remote-control plugin

Send a closure for execution in application

```
compile ":remote-control:2.0"
```

INTERACTING WITH GRAILS 3.0

Application is in same jvm

Interaction is possible directly

Tests run as integration tests

INTERACTING WITH GRAILS 3.0

```
setup:
Attendee.withNewTransaction {
  15.times {
    new Attendee(name: "N$it", email: "m$it@t.dk").save()
  }
}
```




CONFIGURATION AND BROWSER SUPPORT

GEBCONFIG

Configuration for Geb is placed in `GebConfig.groovy`

It must be placed in default package scope

💡 <http://www.gebish.org/manual/current/configuration.html>

A faint, light gray background pattern consisting of a network of interconnected circles and lines, resembling a molecular structure or a web graph. The circles vary in size, and the lines are thin and gray.

DRIVER

It is possible to configure the browser used.

SUPPORTED DRIVERS

An abstract background graphic consisting of a network of interconnected circles and lines. The circles vary in size and are connected by thin, light gray lines, creating a complex, organic pattern that resembles a molecular structure or a network diagram. The overall color palette is light gray and white.

- Firefox
- Chrome
- InternetExplorer
- Safari
- HtmlUnit
- PhantomJS

DRIVER

It is possible to configure the browser used.

build.gradle

```
compile 'org.seleniumhq.selenium:selenium-chrome-driver:2.49.0'  
compile 'org.seleniumhq.selenium:selenium-firefox-driver:2.49.0'
```

FIREFOX

GebConfig.groovy

```
import org.openqa.selenium.firefox.FirefoxProfile
import org.openqa.selenium.firefox.FirefoxDriver

driver = {
    FirefoxProfile profile = new FirefoxProfile()
    profile.setPreference("browser.download.folderList", 2)
    profile.setPreference("browser.download.dir", "/tmp")
    profile.setPreference(
        "browser.helperApps.neverAsk.saveToDisk", "text/csv")

    def driverInstance = new FirefoxDriver(profile)
    driverInstance.manage().window().maximize()
    driverInstance
}
```

CHROME

- Needs ChromeDriver downloaded
- Pretty fast and stable

CHROME (1)

GebConfig.groovy

```
private String driverLocationDependingOnOperatingSystem() {  
    String os = System.getProperty("os.name").toLowerCase();  
    def loc = "http://chromedriver.storage.googleapis.com/2.20"  
    if( os.contains('mac')) {  
        return "${loc}/chromedriver_mac32.zip"  
    }  
    if( os.contains('win')) {  
        return "${loc}/chromedriver_win32.zip"  
    }  
    return "${loc}/chromedriver_linux64.zip"  
}
```


CHROME (2)

GebConfig.groovy

```
private void downloadDriver(File file, String path) {  
    if (!file.exists()) {  
        def ant = new AntBuilder()  
        ant.get(src: path, dest: 'driver.zip')  
        ant.unzip(src: 'driver.zip', dest: file.parent)  
        ant.delete(file: 'driver.zip')  
        ant.chmod(file: file, perm: '700')  
    }  
}
```

CHROME (3)

GebConfig.groovy

```
def chromeDriver = new File('build/drivers/chrome/chromedriver')
downloadDriver(chromeDriver,
    driverLocationDependingOnOperatingSystem())
System.setProperty('webdriver.chrome.driver',
    chromeDriver.absolutePath)

driver = {
    def driverInstance = new ChromeDriver()

    def browserWindow = driverInstance.manage().window()
    // width, height
    browserWindow.size = new Dimension(1000, 2500)
    browserWindow.position = new Point(0, 0)

    driverInstance
}
```

CLOUD BASED TESTING

- SauceLabs
- BrowserStack

SAUCELABS

GebConfig.groovy

```
import geb.driver.SauceLabsDriverFactory

def slBrowser = System.getProperty("geb.saucelabs.browser")
if (slBrowser) {
    driver = {
        def username = System.getenv(
            "GEB_SAUCE_LABS_USER")
        assert username
        def accessKey = System.getenv(
            "GEB_SAUCE_LABS_ACCESS_PASSWORD")
        assert accessKey
        new SauceLabsDriverFactory().create(
            saucelabsBrowser, username, accessKey)
    }
}
```

SAUCELABS

build.gradle

```
apply plugin: "geb-saucelabs"

repositories {
    ...
    mavenCentral()
}

dependencies {
    ...
    sauceConnect "com.saucelabs:ci-sauce:1.113"
}
```

SAUCELABS

```
sauceLabs {
  browsers {
    firefox_linux_19
    delegate."internet explorer_vista_9"
    nexus4 {
      capabilities(
        browserName: "android", platform: "Linux",
        version: "4.4", deviceName: "LG Nexus 4"
      )
    }
  }
  task {
    testClassesDir = test.testClassesDir
    testSrcDirs = test.testSrcDirs
    classpath = test.classpath
  }
  account {
    username = "username"
    accessKey = "api-key"
  }
  connect {
    //      port = 4444
    //      additionalOptions = ['--proxy', 'proxy.example.com:8080']
  } }
```

The background of the slide features a complex, abstract network diagram. It consists of numerous circular nodes of varying sizes, some of which are solid gray, while others are hollow with gray outlines. These nodes are interconnected by a web of thin, light gray lines, creating a sense of connectivity and structure. The overall aesthetic is clean and modern, typical of a technical or educational presentation.

JAVASCRIPT

In case you need to interact using javascript

EXECUTING JAVASCRIPT

Clicking a button that is hidden will create a
`ElementNotVisibleException`

```
<fieldset class="well" style="display: none">  
  <a href="/list" class="btn">List</a>  
</fieldset>
```


EXECUTING JAVASCRIPT

```
JavascriptExecutor executor = (JavascriptExecutor) driver  
executor.executeScript('jQuery(".well").show();')
```

WRAPPING JAVASCRIPT

```
def js( String script ){  
    (driver as JavascriptExecutor).executeScript( script )  
}
```

```
js('jQuery(".well").show();')
```

JQUERY SHORTHAND

```
$("#div#a").jquery.mouseover()  
$("#a").jquery.trigger('mouseover')
```



WAITING AND PAUSING

WAITING

GebConfig.groovy

```
waiting {  
    timeout = 10  
    retryInterval = 0.5  
}  
  
baseNavigatorWaiting = true  
atCheckWaiting = true
```

USING WAITING

```
<div class="fade-me-in" style="display: none">
  Hi - are yo waiting for me?
</div>
<script>
  $('div.fade-me-in').delay(3000).slideDown();
</script>
```

```
static content = {
  fadeInMessage{ $('div.fade-me-in') }
}
```

```
then:
waitFor {
  fadeInMessage.text() == 'Hi - are yo waiting for me?'
}
```

PAUSING GEB

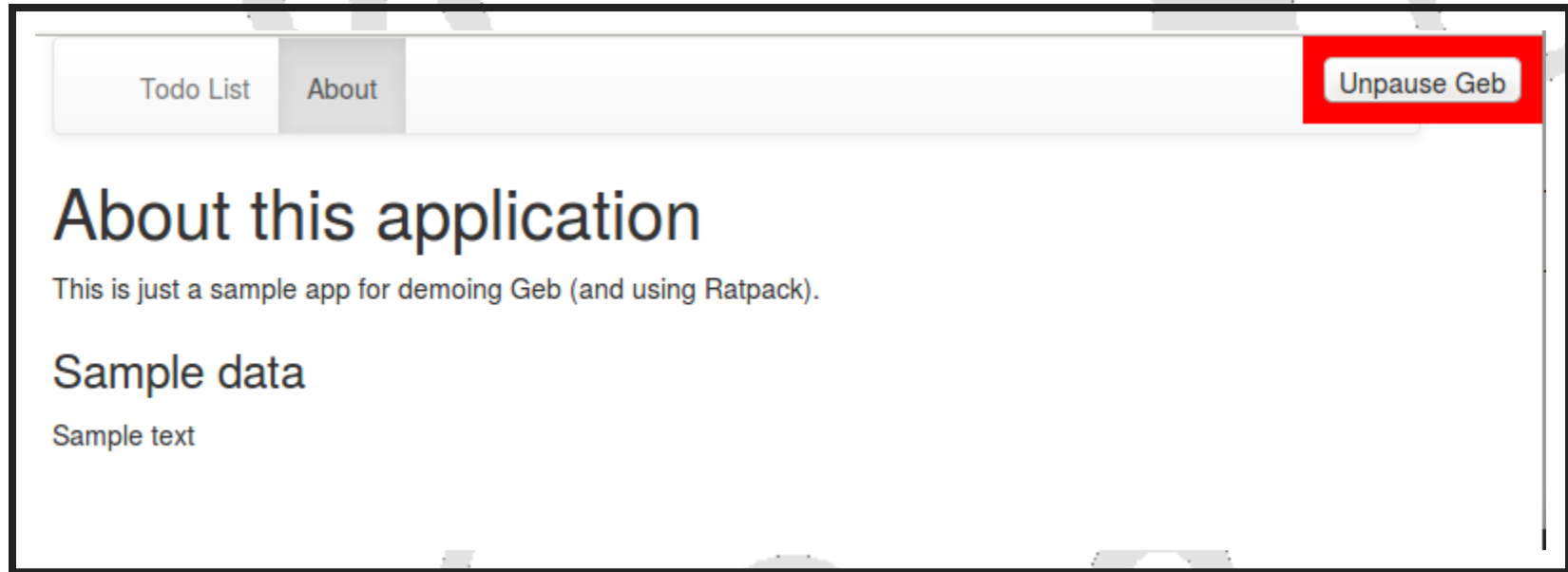
```
private void pause() {  
    js.exec """(function() {  
        window.__gebPaused = true;  
        var div = document.createElement("div");  
        div.setAttribute('style',  
            "position: absolute; top:0px;right: 0px;z-index: 3000;\\  
            padding: 10px; background-color: red;");  
        var button = document.createElement("button");  
        button.innerHTML = "Unpause Geb";  
        button.onclick = function() {  
            window.__gebPaused = false;  
        }  
        div.appendChild(button);  
        document.getElementsByTagName("body")[0].appendChild(div);  
    })();"""  
    waitFor(300) { !js.__gebPaused }  
}
```

PAUSING GEB

```
when: 'Demoing pause'
```

```
pause() // Pause Geb until button pressed
```


PAUSING GEB



An abstract network diagram with various sized nodes (circles) connected by lines, forming a complex web. The nodes are light gray with white centers, and the lines are thin gray. The word "REPORTING" is centered over the diagram.

REPORTING

TEST REPORTS

- Nicely formatted
- Spock power-assert format

SPOCK REPORTS

Take a look at [Spock Reports](#) plugin

```
testCompile( 'com.athaydes:spock-reports:1.2.10' ) {  
    // this avoids affecting your version of Groovy/Spock  
    transitive = false  
}
```

SPOCK REPORTS

Report for net.grydeske.greach.TODOListFunctionalSpec

Summary:

Created on Wed Apr 06 18:28:48 CEST 2016 by jacob

Executed features	Fallures	Errors	Skipped	Success rate	Time
5	0	0	0	100.0%	3.598 seconds

Features:

- [Go to index page](#)
- [Create new todo](#)
- [Delete todo item](#)
- [Complete item and check database](#)
- [Create batch items](#)

Go to index page

[Return](#)

When: Go to index url

Then: Verify 3 items present

Create new todo

[Return](#)

When: Input text and submit

Then: Verify new item present in list

And: Verify input field empty

SPOCK REPORTS

Report for net.grydeske.greach.TODOListFunctionalSpec

Summary:

Created on Thu Apr 07 10:00:12 CEST 2016 by jacob

Executed features	Failures	Errors	Skipped	Success rate	Time
1	1	0	5	0.0%	0.931 seconds

Features:

- [Go to index page](#)
- [Create new todo](#)
- [Delete todo item](#)
- [Complete item and check database](#)
- [Create batch items](#)

Go to index page

[Return](#)

When: Go to index url

Then: Verify 3 items present

The following problems occurred:

- Condition not satisfied:

```
countValue == 'Create an error for report'
|           |
3           false
           26 differences (0% similarity)
           (3-----)
           (Create an error for report)
```

SCREENSHOTS

Screenshots and HTML from end of each test:

Extend from `GebReportingSpec`

```
class AttendeeFunctionalSpec extends GebReportingSpec
```

GebConfig.groovy

```
reportsDir = new File("build/geb-reports")  
reportOnTestFailureOnly = true
```

AD-HOC SCREENSHOTS

```
report "When-form-is-just-filled"
```

- Saves a report in reportsDir
- Numbered in increasing order

OTHER USAGES

- Screenscraping of a site
- Solving complex problems like 2048

RESOURCES

- <http://gebish.org>
- <https://groups.google.com/forum/#!forum/geb-user>
- <https://github.com/geb>
- <https://gist.github.com/melix/9619800>
- <https://github.com/tomaslin/grails-test-recipes>
- <https://fbflex.wordpress.com/2010/08/25/geb-and-grails-tips-tricks-and-gotchas/>
- <https://github.com/JacobAae/gr8conf-in-2016-geb-for-grails>



QUESTIONS?

some text...