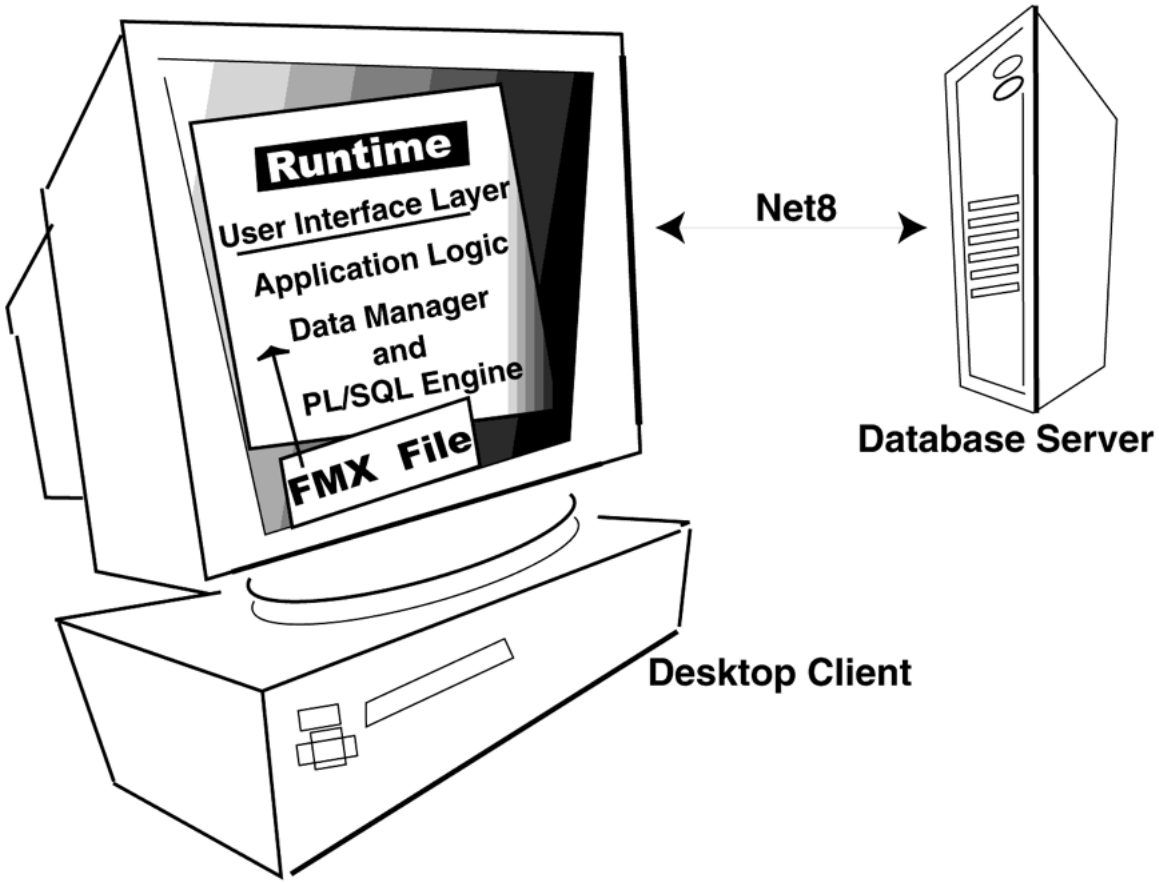
Web Acceptance Testing Revisited

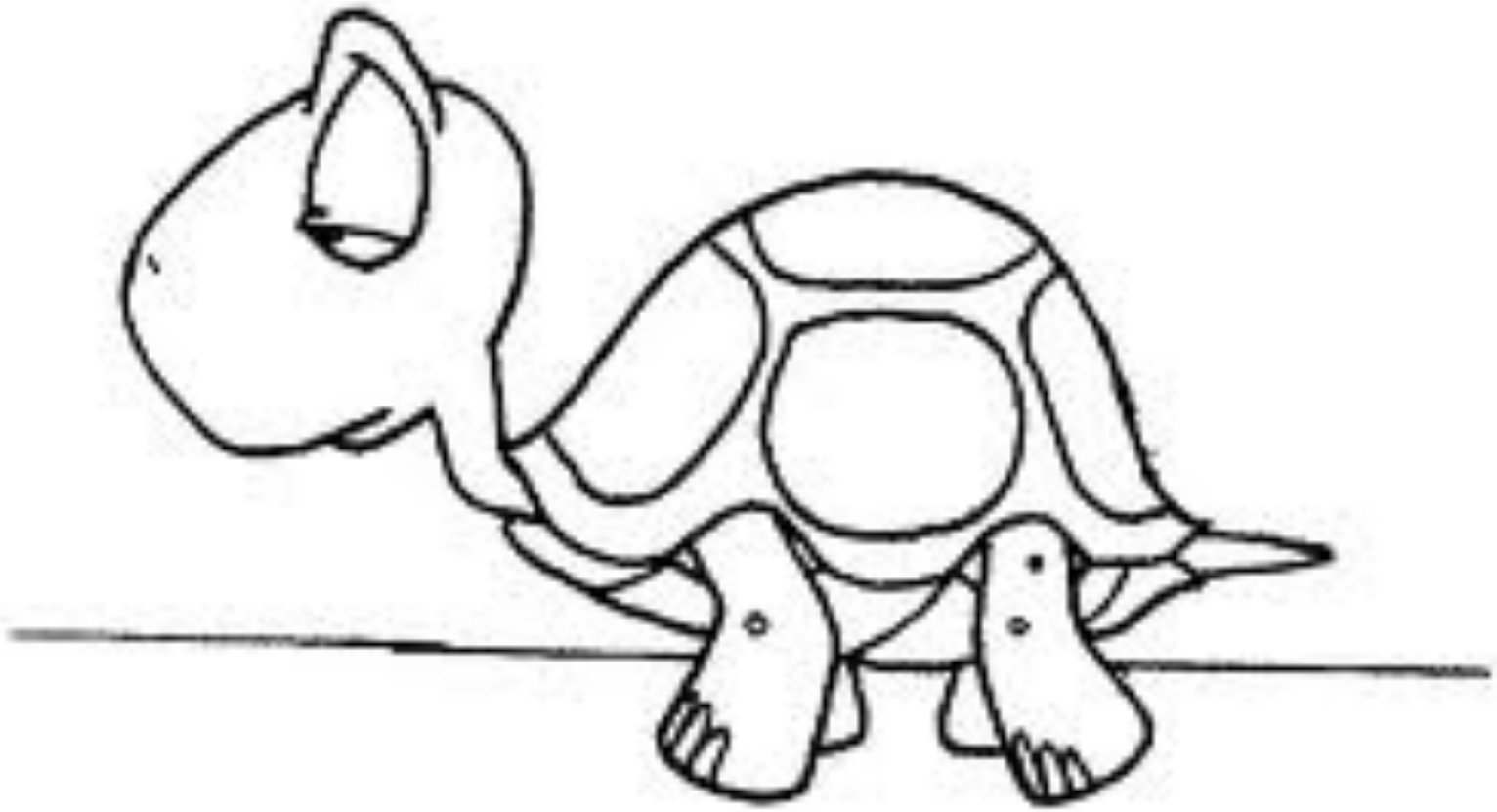
Björn Beskow| bjorn.beskow@callistaenterprise.se | 2012-01-18

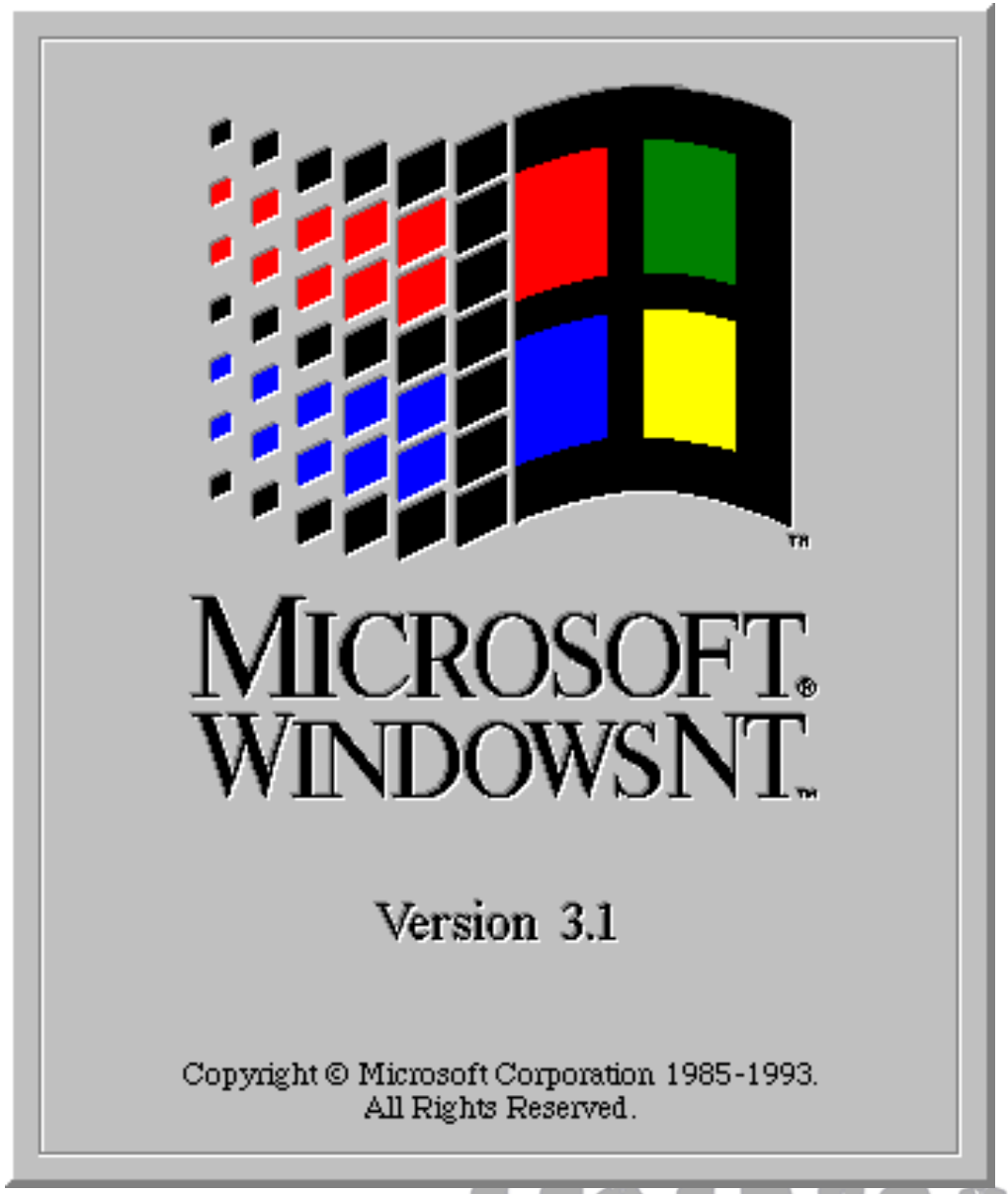
CALLISTA Enterprise

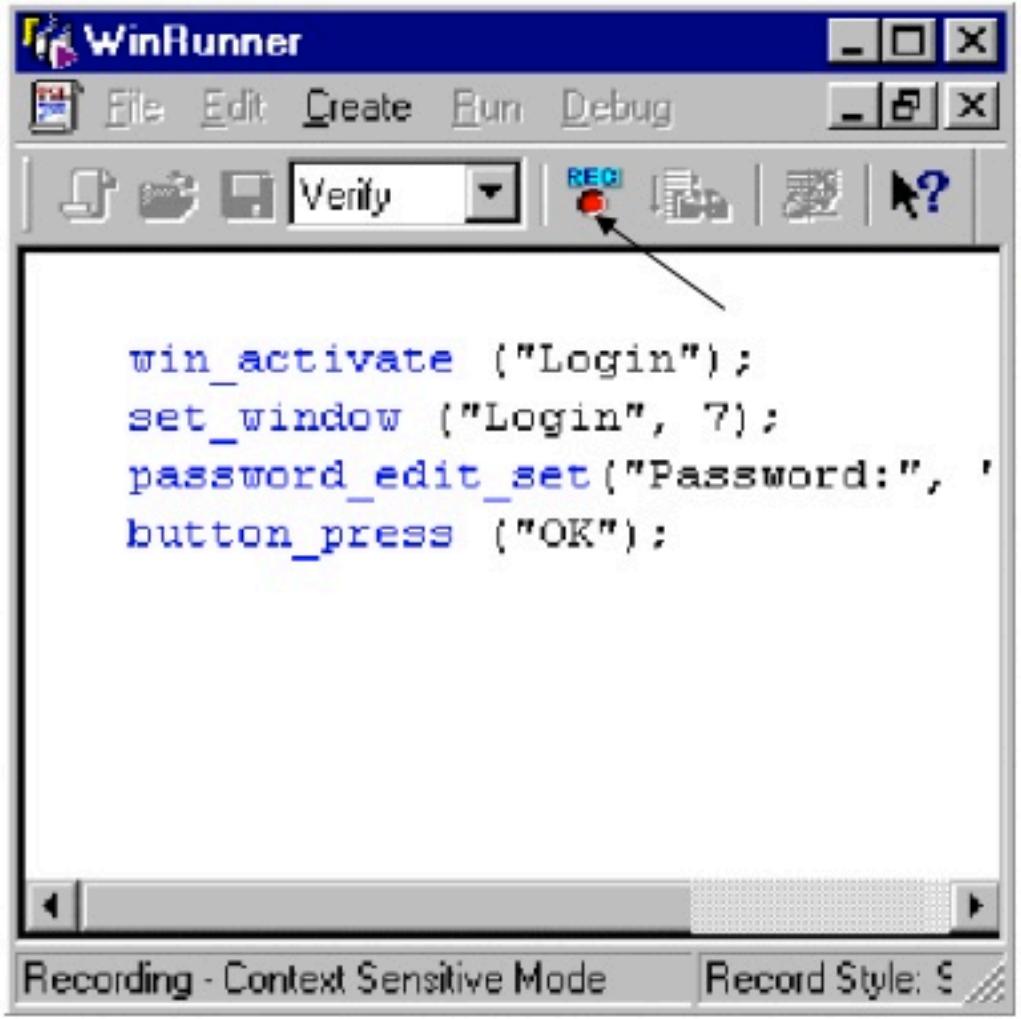
Testing Graphical User Interfaces is ...

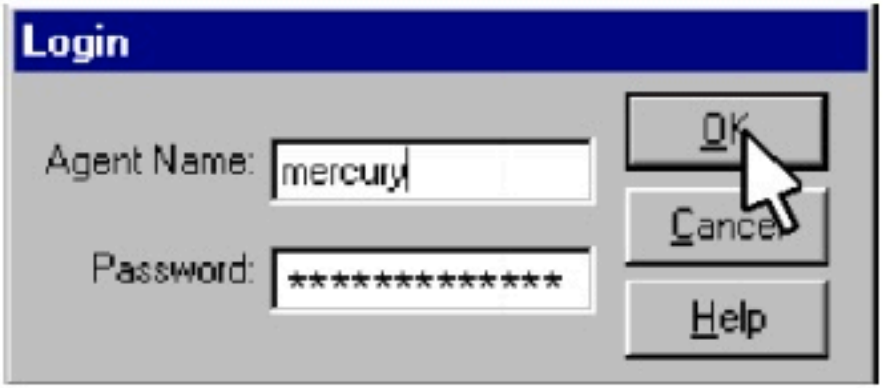
Client Server era



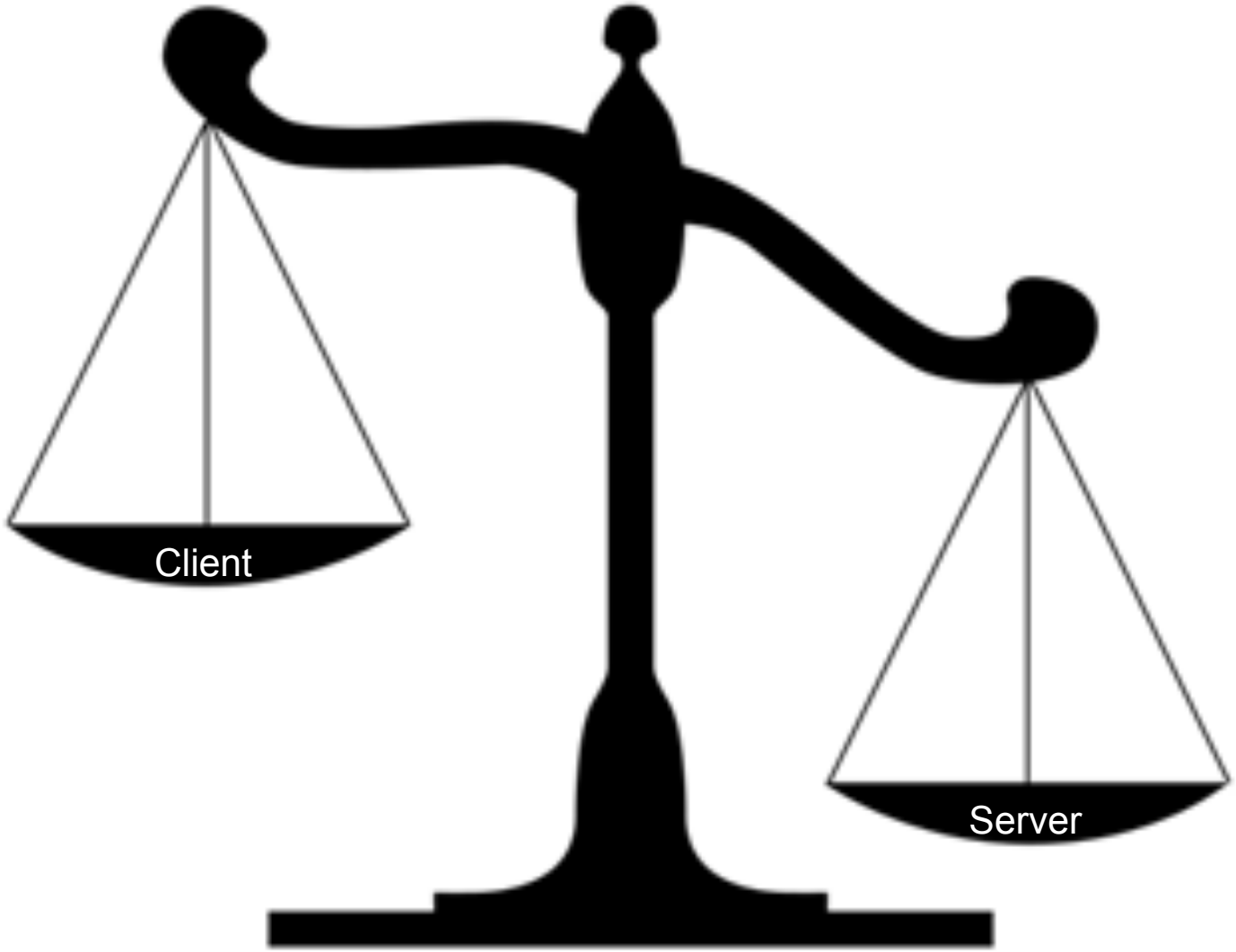








www



ALL CODE IS GUILTY
UNTIL PROVEN INNOCENT

CODESMACK

```
'_____
_____

'get login details

Call LoginInfo()

strComputerName = Environment.Value("envComputerName")

strUserName  = Environment.Value("envUserName")

strUserDomain = Environment.Value("envUserDomain")

'_____
_____


Set objBrwsr=Description.Create

objBrwsr("micclass").value="Browser"

objBrwsr("name").value=".*"


Set objLink=Description.Create

objLink("html tag").value="A"

objLink("micclass").value="Link"
```

```
public void testSearch() {
    beginAt("/");
    assertFormElementPresent("q");
    setFormElement("q", "HttpUnit");
    submit("btnG");
    assertLinkPresentWithText(searchLink);
    clickLinkWithText(searchLink);
}
```

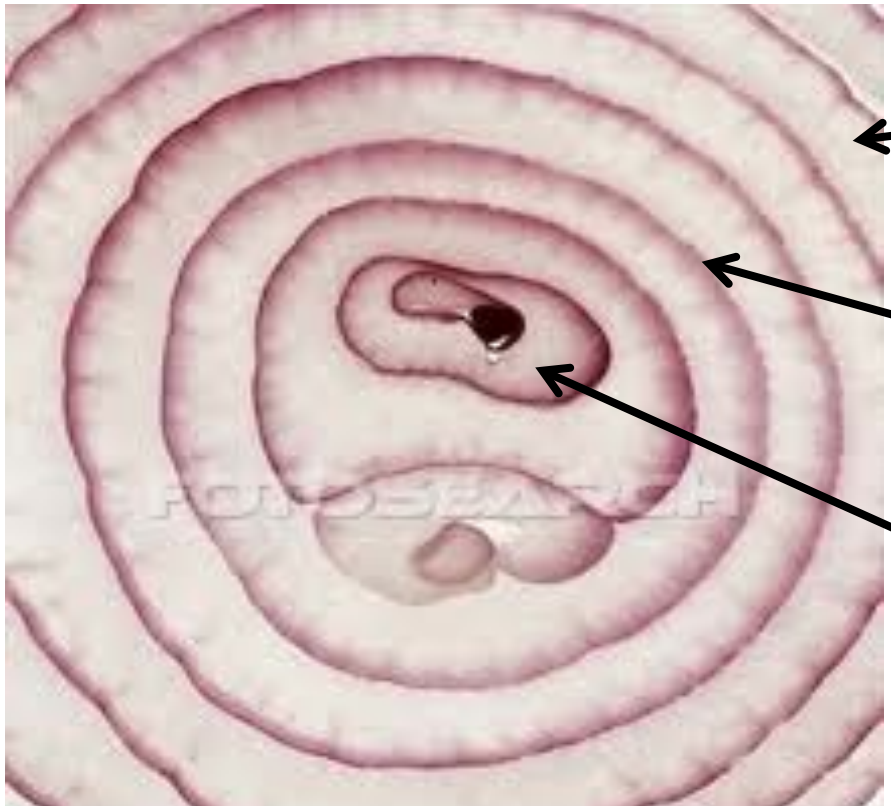
… except for the GUI …

Web 2.0

"Flickr deploys 10 times a day - why don't you?"

Huston, we have a problem ...

… and hence we need yet some layers of abstraction!

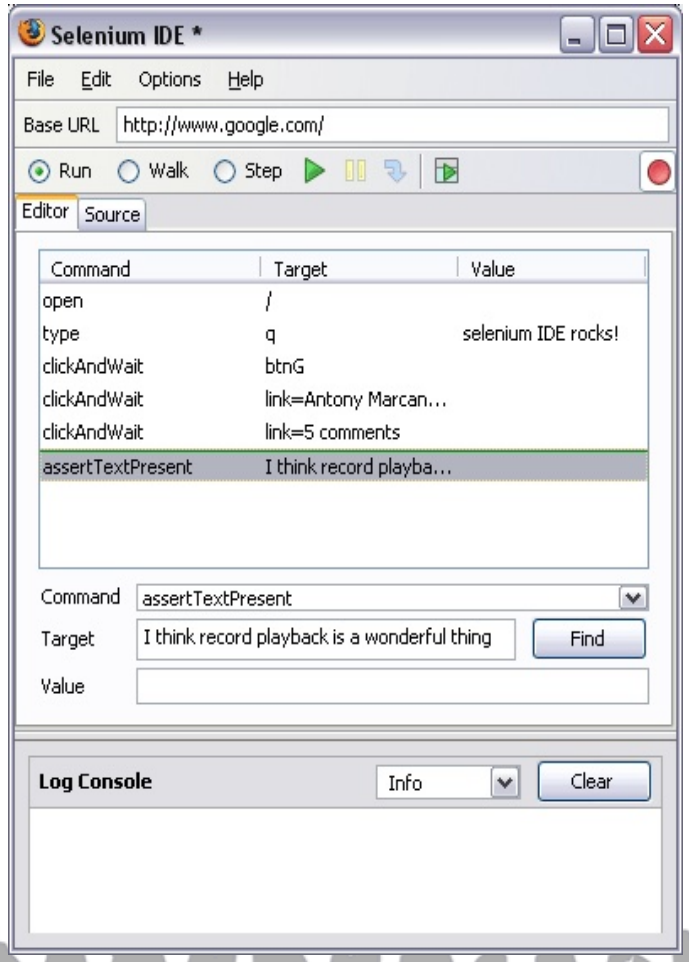


- Specifications
- DOM Abstraction
- Browser Automation

CALLISTA Enterprise

Selenium: A framework for Browser Automation

- The Selenium engine executes tests **directly in a browser**, just as real users do.
- The Selenium 1.0 test engine is implemented in JavaScript, and thus run in most browser (Internet Explorer, Mozilla, Firefox, Opera, Safari …) on most platforms.



Selenium + WebDriver = Selenium 2.0

- First production-ready version in July 2011
- Greatly enhanced model compared to Selenium 1.0
  - Object-based API for DOM Interaction
  - Browser Driver Implementations
  - Advanced User Interactions

At the heart of Selenium: *Selenese*

click	getHtmlSource	isVisible
close	getTitle	keyPress
createCookie	getValue	mouseOver
dragdrop	goBack	open
fireEvent	isElementPresent	refresh
getEval	isTextPresent	type

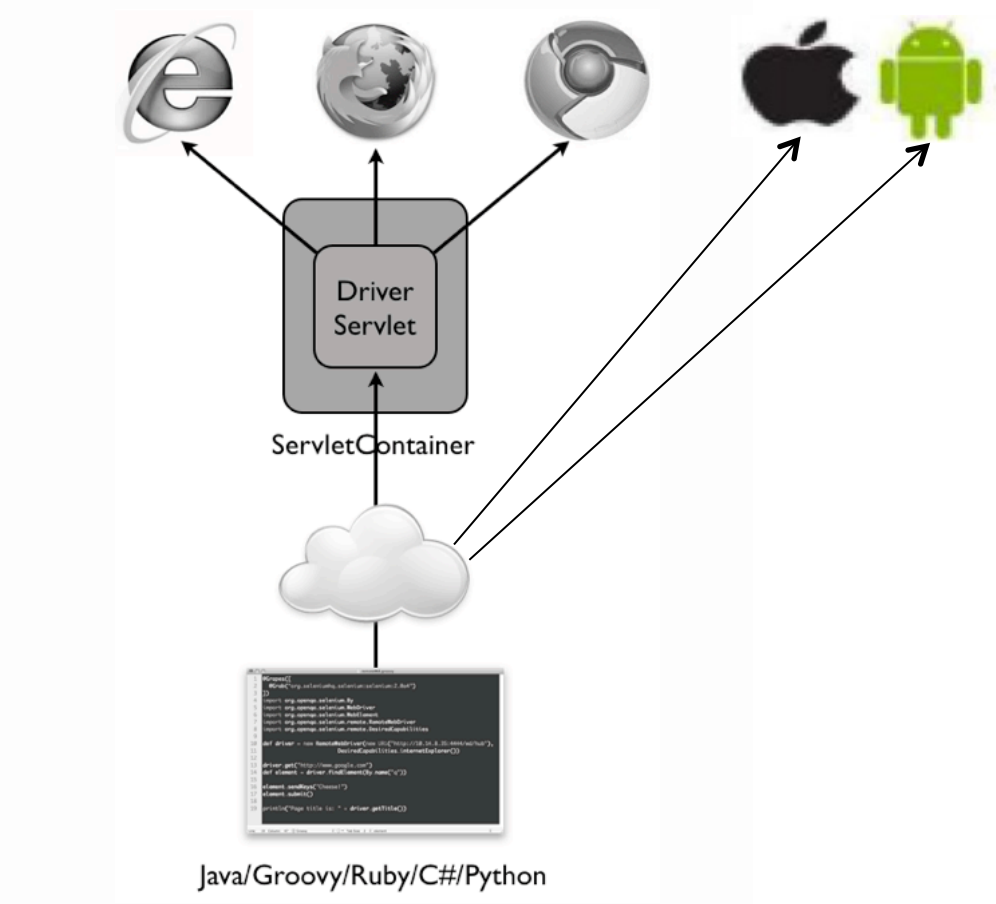
Selenium 2.0 Tool Chain

- Selenium IDE
- Browser Drivers
- Remote WebDriver
- Selenium Server (Selenium RC/ WebDriver)
- Selenium Grid

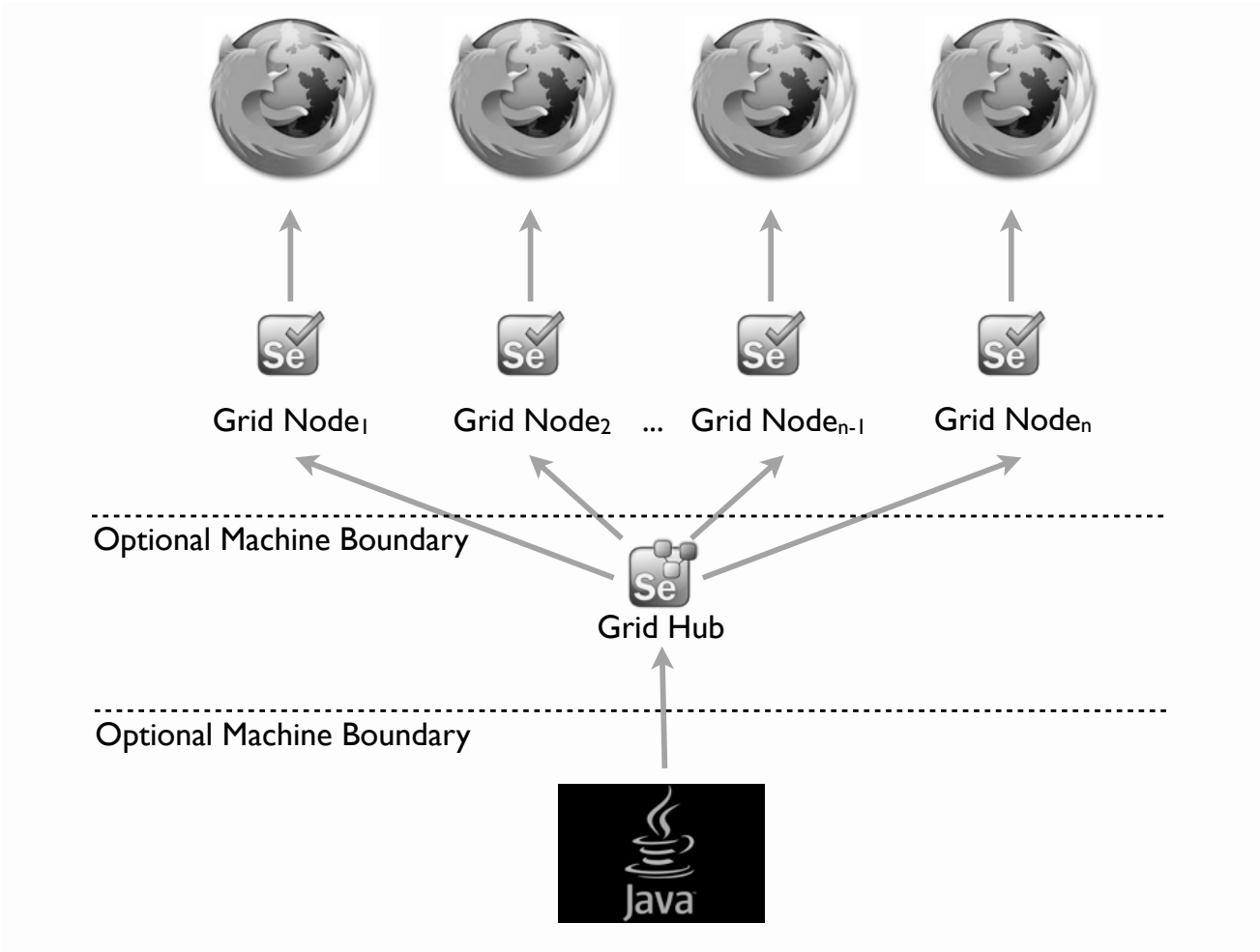
Browser Drivers

- Native implementations of the WebDriver API
  - Firefox
  - Chrome
  - Internet Explorer
  - Opera
  - Headless (HtmlUnit)
  - iOS
  - Android
  - …

Selenium Remote Driver

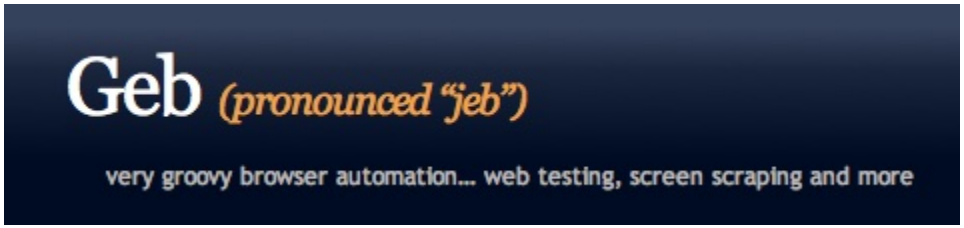


Selenium Grid



DOM Abstraction: Geb

- Brings together
  - The power of Selenium / Web Driver
  - The elegance of JQuery
  - The robustness of the Page Object pattern
  - The flexibility and pure joy of Groovy



Navigator API

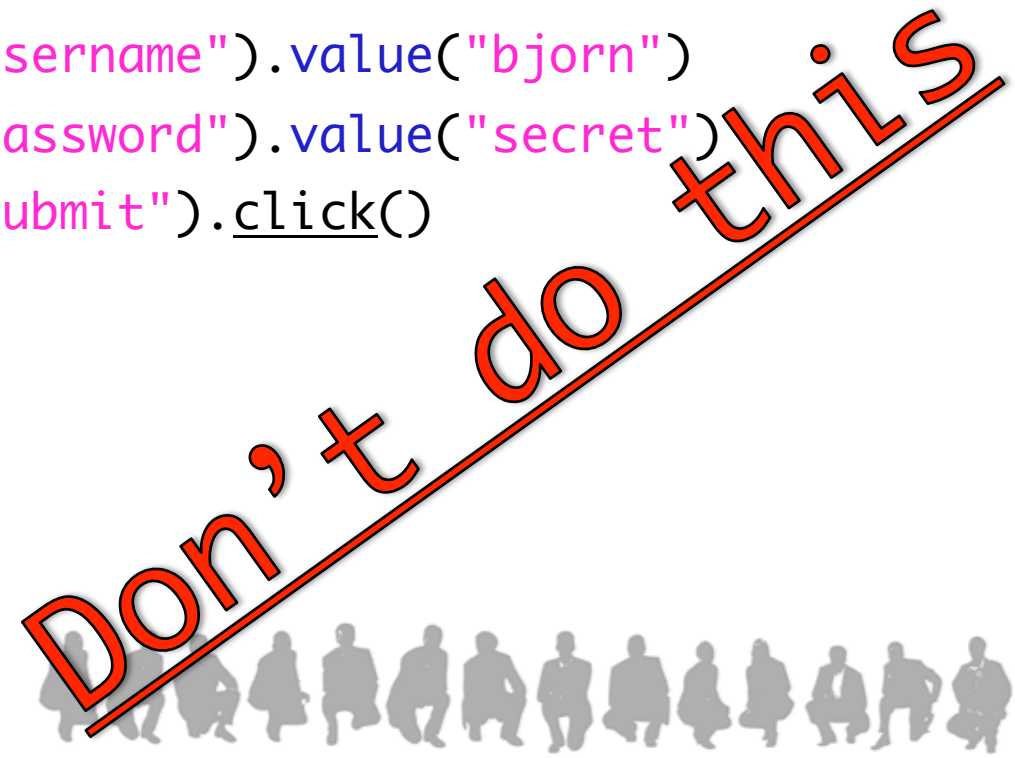
- JQuery-like expressions provide a concise and effective way to navigate the DOM

```
$("p", text: contains(”Timecard"))
$("input", name: "username").value("bjorn")
$("div.message").text()
```

Page Objects

- Use proper object-orientation techniques to avoid brittleness and duplication

```
$("input", name: "username").value("bjorn")
$("input", name: "password").value("secret")
$("input", name: "submit").click()
```

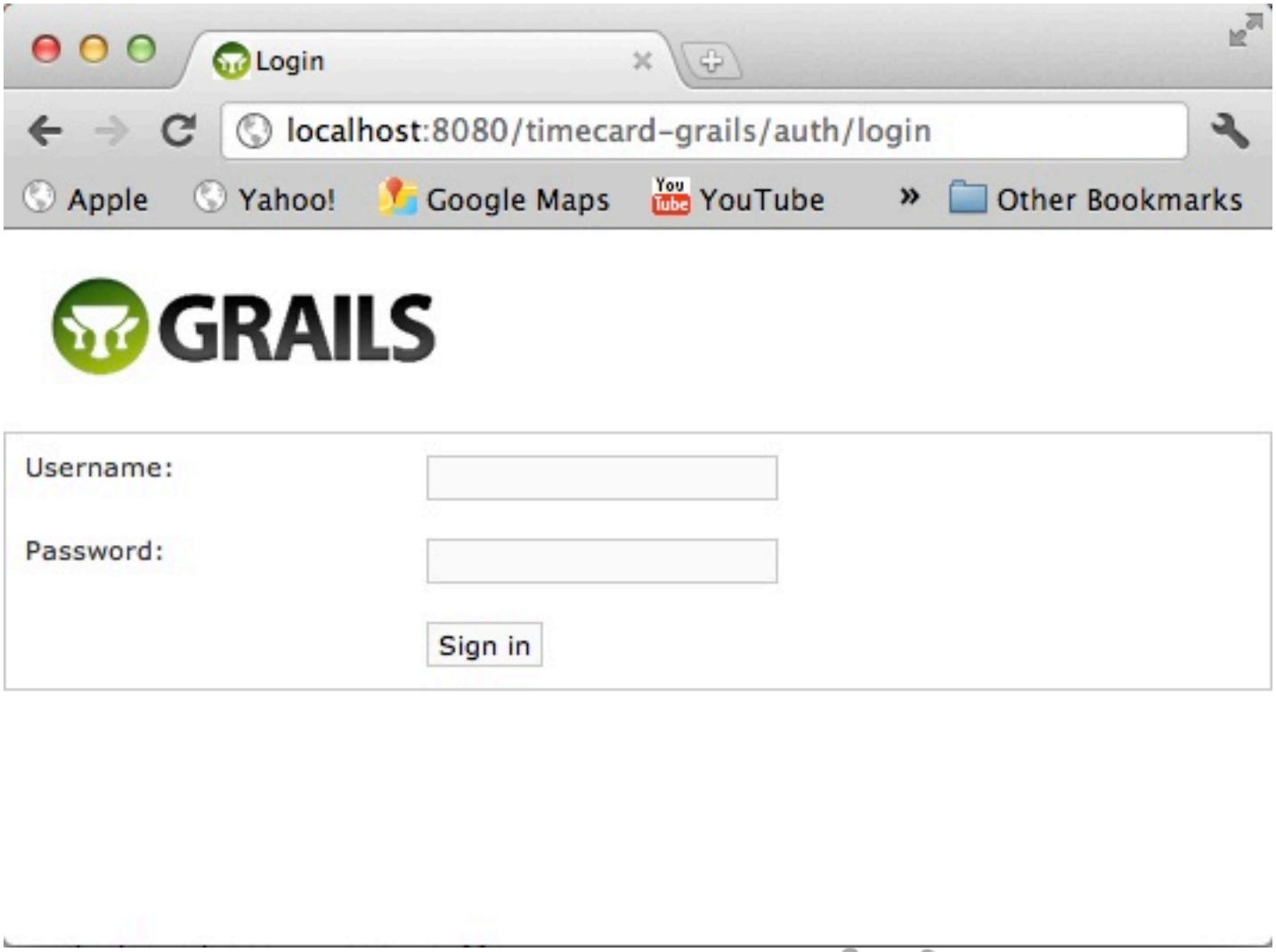


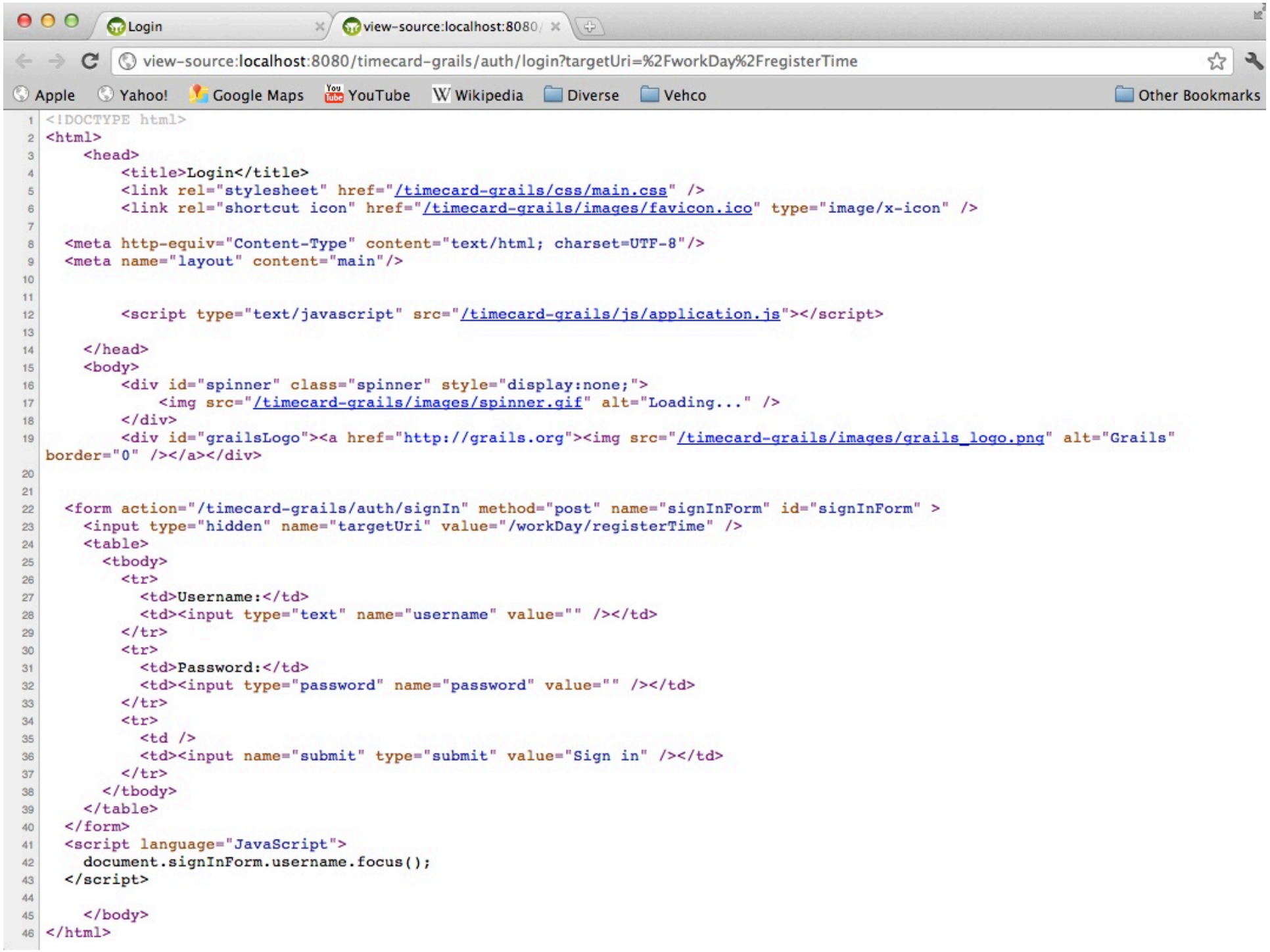
Page Objects

- Use proper object-orientation techniques to avoid brittleness and duplication

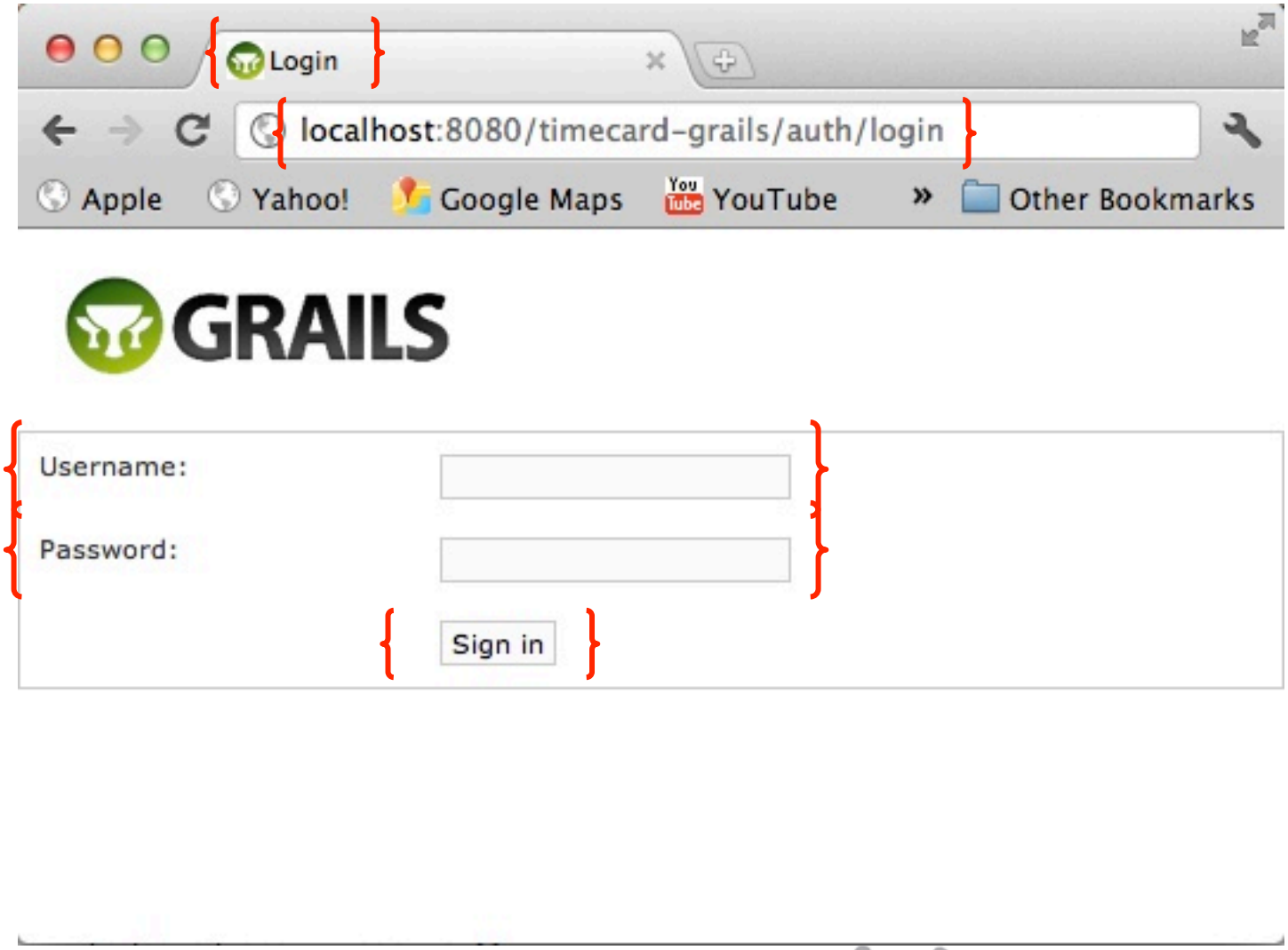
```
page.login("bjorn", "secret")
```

Do this instead




Login
view-source:localhost:8080/
view-source:localhost:8080/timecard-grails/auth/login?targetUri=%2FworkDay%2FregisterTime
Apple
Yahoo!
Google Maps
YouTube
Wikipedia
Diverse
Vehco
Other Bookmarks
<!DOCTYPE html>
<html>
<head>
<title>Login</title>
<link rel="stylesheet" href="/timecard-grails/css/main.css" />
<link rel="shortcut icon" href="/timecard-grails/images/favicon.ico" type="image/x-icon" />
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
<meta name="layout" content="main"/>
<script type="text/javascript" src="/timecard-grails/js/application.js"></script>
</head>
<body>

<form action="/timecard-grails/auth/signIn" method="post" name="signInForm" id="signInForm" >
<input type="hidden" name="targetUri" value="/workDay/registerTime" />
<table>
<tbody>
<tr>
<td>Username:</td>
<td><input type="text" name="username" value="" /></td>
</tr>
<tr>
<td>Password:</td>
<td><input type="password" name="password" value="" /></td>
</tr>
<tr>
<td />
<td><input name="submit" type="submit" value="Sign in" /></td>
</tr>
</tbody>
</table>
</form>
<script language="JavaScript">
document.signInForm.username.focus();
</script>
</body>
</html>



Geb Page definition

```
class LoginPage extends Page {
        static url = "/timecard-grails/auth/login”
        static at = { title == "Login" }
        static content = {
                username { $("input", name: "username") }
                password { $("input", name: "password") }
                login { $("input", name: "submit") }
                message(required: false) { $("div.message").text() }
        }
        def login(name, passwd) {
                username.value(name)
                password.value(passwd)
                login.click()
        }
}
```

Geb Page definition: navigation

```
class LoginPage extends Page {
        static url = "/timecard-grails/auth/login”
        static at = { title == "Login" }
        static content = {
                username { $("input", name: "username") }
                password { $("input", name: "password") }
                login { $("input", name: "submit") }
                message(required: false) { $("div.message").text() }
        }
        def login(name, passwd) {
                username.value(name)
                password.value(passwd)
                login.click()
        }
}
```

Geb Page definition: *at* predicate

```
class LoginPage extends Page {
        static url = "/timecard-grails/auth/login"
        static at = { title == "Login" }
        static content = {
                username { $("input", name: "username") }
                password { $("input", name: "password") }
                login { $("input", name: "submit") }
                message(required: false) { $("div.message").text() }
        }
        def login(name, passwd) {
                username.value(name)
                password.value(passwd)
                login.click()
        }
}
```

Geb Page definition: logical *content*

```
class LoginPage extends Page {
        static url = "/timecard-grails/auth/login"
        static at = { title == "Login" }
        static content = {
                username { $("input", name: "username") }
                password { $("input", name: "password") }
                login { $("input", name: "submit") }
                message(required: false) { $("div.message").text() }
        }
        def login(name, passwd) {
                username.value(name)
                password.value(passwd)
                login.click()
        }
}
```

Geb Page definition: *actions*

```
class LoginPage extends Page {
        static url = "/timecard-grails/auth/login”
        static at = { title == "Login" }
        static content = {
                username { $("input", name: "username") }
                password { $("input", name: "password") }
                login { $("input", name: "submit") }
                message(required: false) { $("div.message").text() }
        }
        def login(name, passwd) {
                username.value(name)
                password.value(passwd)
                login.click()
        }
}
```

Specifications

Business people wants to express required behaviour, not necessarily write tests

CALLISTA Enterprise

Spock

- Another BDD framework for the Java platform
- Based on Groovy

CALLISTA Enterprise

Example Task

Task: Login is required for time reporting

Given a user

When he navigates to the time reporting page

Then he has to log in

Example Task

Task: Regular days

Given a white-collar employee

When he arrives at 8:00 and leaves at 17:00

Then his result for that day is 0.

Spock specification

```
class TimeReportTest extends GebSpec {
  def "White-collar time reporting, regular day"() {
        given:
                to HomePage
                registerTime.click()
                waitFor { at LoginPage }
                login("bjorn", "secret")
        expect:
                at TimeReportPage
        when:
                arriveAt("08:00")
                leaveAt("17:00")
        then:
                at ResultsPage
                flex == "0"
  }
}
```

Demo!

Time for Questions!