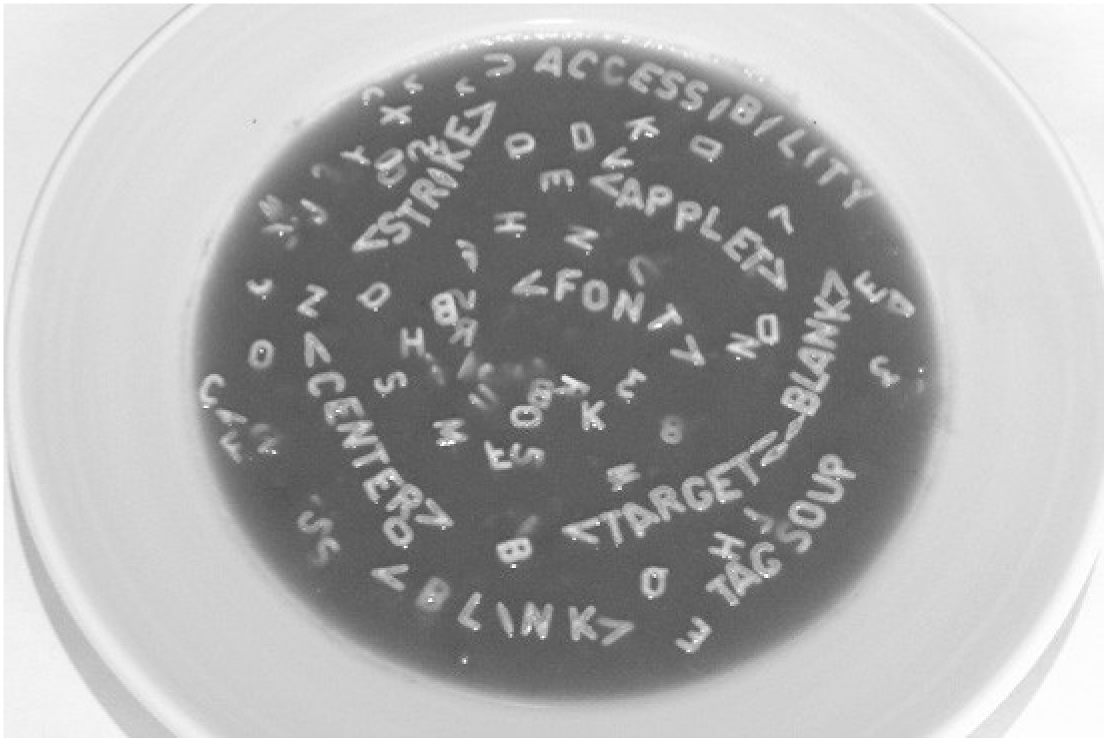
## Integration Testing: Why?

> Sanity checks
> Top-to-bottom integration
> Find errors early
- HTTP errors
- HTML errors
- JavaScript errors
- Backend errors

> **Not** proving the application is bug-free!



# HtmlUnit: An Efficient Approach to Testing Web Applications

Daniel Gredler

DHL Global Mail

daniel.gredler@gmail.com

http://daniel.gredler.net

## Agenda

> Integration Testing: Why?

> Browser Driving: Pros and Cons

> Browser Simulation: HtmlUnit

> Browser Simulation: Pros and Cons

> Wrappers Around HtmlUnit

> HtmlUnit Future Plans

> Q&A

## Browser Driving: Pros

- Feedback / visualization
- Fidelity to the user experience
- Leverages browser configuration
  - Browser plugins
  - Flash
  - Google Gears
  - …
- Easy to create tests (recorders)

## Simulation: HtmlUnit

- 100% Java-based headless browser
- Open Source (Apache 2 License)
  - 7 committers (3 active)
  - Very mature
- Useful for:
  - Integration Testing
  - Web Scraping
  - System Monitoring

## Browser Driving: Cons

> Feedback / visualization

> Platform dependence

> Hard to test multiple browsers

> Limited extensibility

> Performance

> Scalability

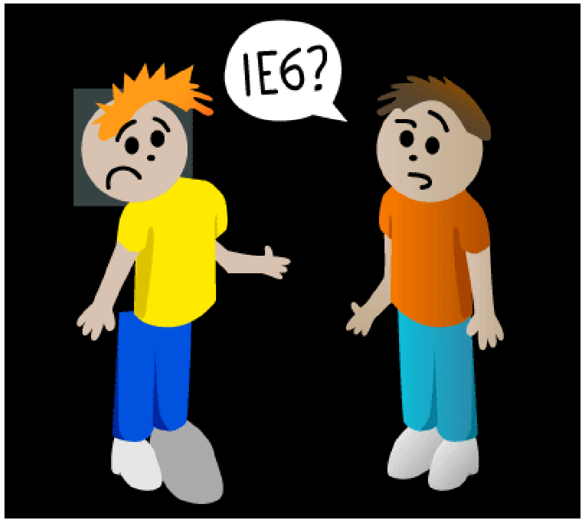
> Recorders encourage limited, brittle tests

## Sample Usage

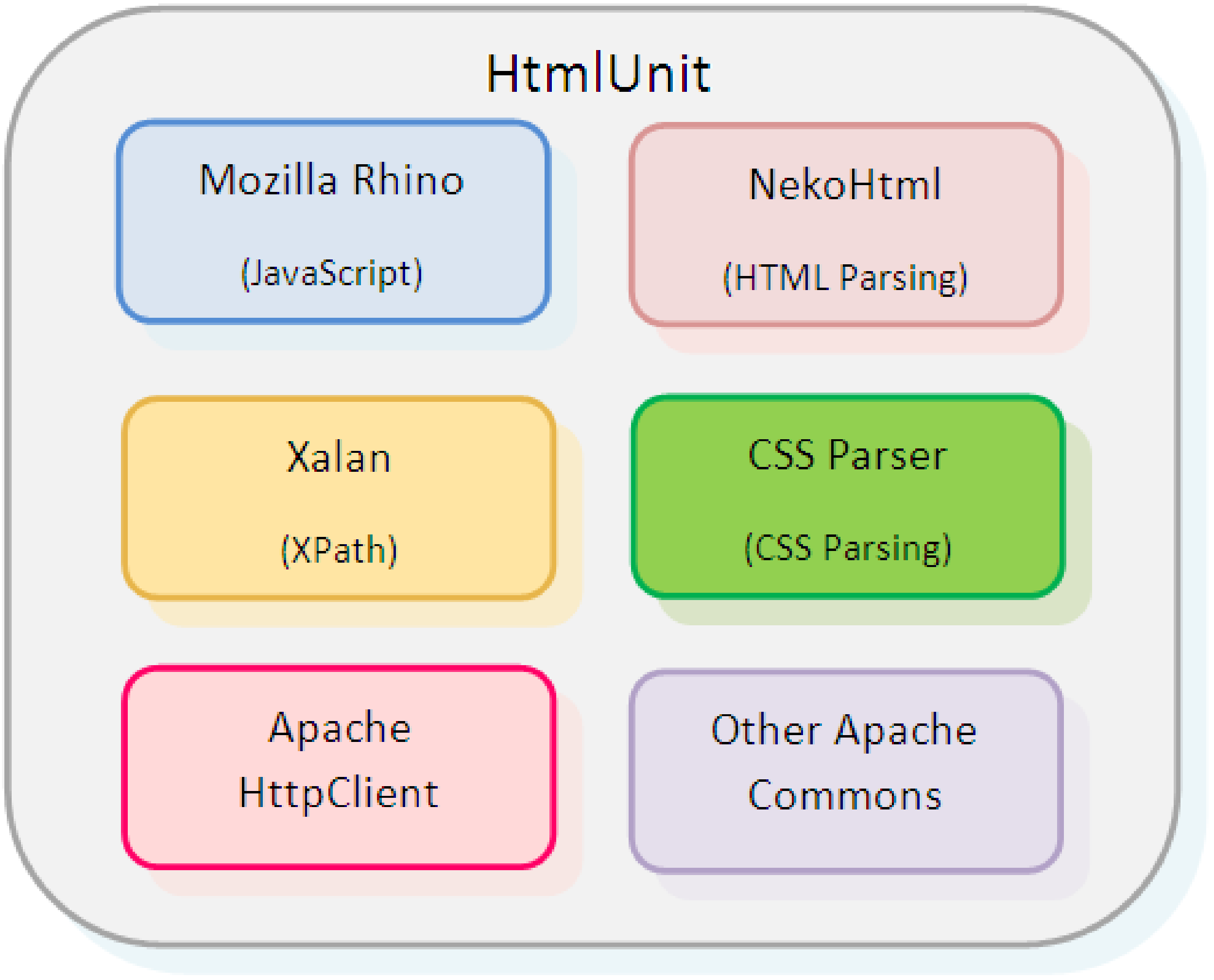
```
@Test
public void google() throws Exception {

    WebClient client = new WebClient(BrowserVersion.FIREFOX_3);

    HtmlPage startPage = client.getPage("http://www.google.com/");
    assertEquals("Google", startPage.getTitleText());

    HtmlElement queryField = startPage.getElementByName("q");
    queryField.click();
    queryField.type("HtmlUnit");

    HtmlElement button = startPage.getFirstByXPath("//input[@name='btnI']");
    HtmlPage page2 = button.click();
    assertEquals("HtmlUnit - Welcome to HtmlUnit", page2.getTitleText());
}
```

## HtmlUnit Simulates "Real" Browsers

- Focuses on 2 browser families:
  - Firefox 2 & 3
  - Internet Explorer 6, 7 & 8
- Mimics browser behavior:
  - HTTP requests
  - HTML parsing
  - CSS parsing
  - JavaScript execution



## Architecture



## Configuration

- Enable / Disable
  - JavaScript
  - CSS
  - Popup Blocker
- Throw / No Throw
  - On script error
  - On HTTP failure status codes
- Use Insecure SSL
- …

## Extension Points

> Alert / Confirm / Prompt / Status Handlers
> JavaScript Pre-processors
> JavaScript Debugger Callbacks
> Custom Web Connections
> Incorrectness Listeners: HTML, CSS, etc.
> …

# Extension Point Example

```java
WebClient client = new WebClient();

client.setWebConnection(new FalsifyingWebConnection(client) {
    @Override
    public WebResponse getResponse(WebRequestSettings wrs) throws IOException {
        if ("some.other.server".equals(wrs.getUrl().getHost())) {
            int status = 500;
            String msg = "Internal server error.";
            byte[] body = "An error occurred.".getBytes();
            List<NameValuePair> headers = Collections.emptyList();
            WebResponseData wrd = new WebResponseData(body, status, msg, headers);
            return new WebResponseImpl(wrd, wrs.getUrl(), wrs.getHttpMethod(), 1);
        }
        else {
            return super.getResponse(wrs);
        }
    }
});
```

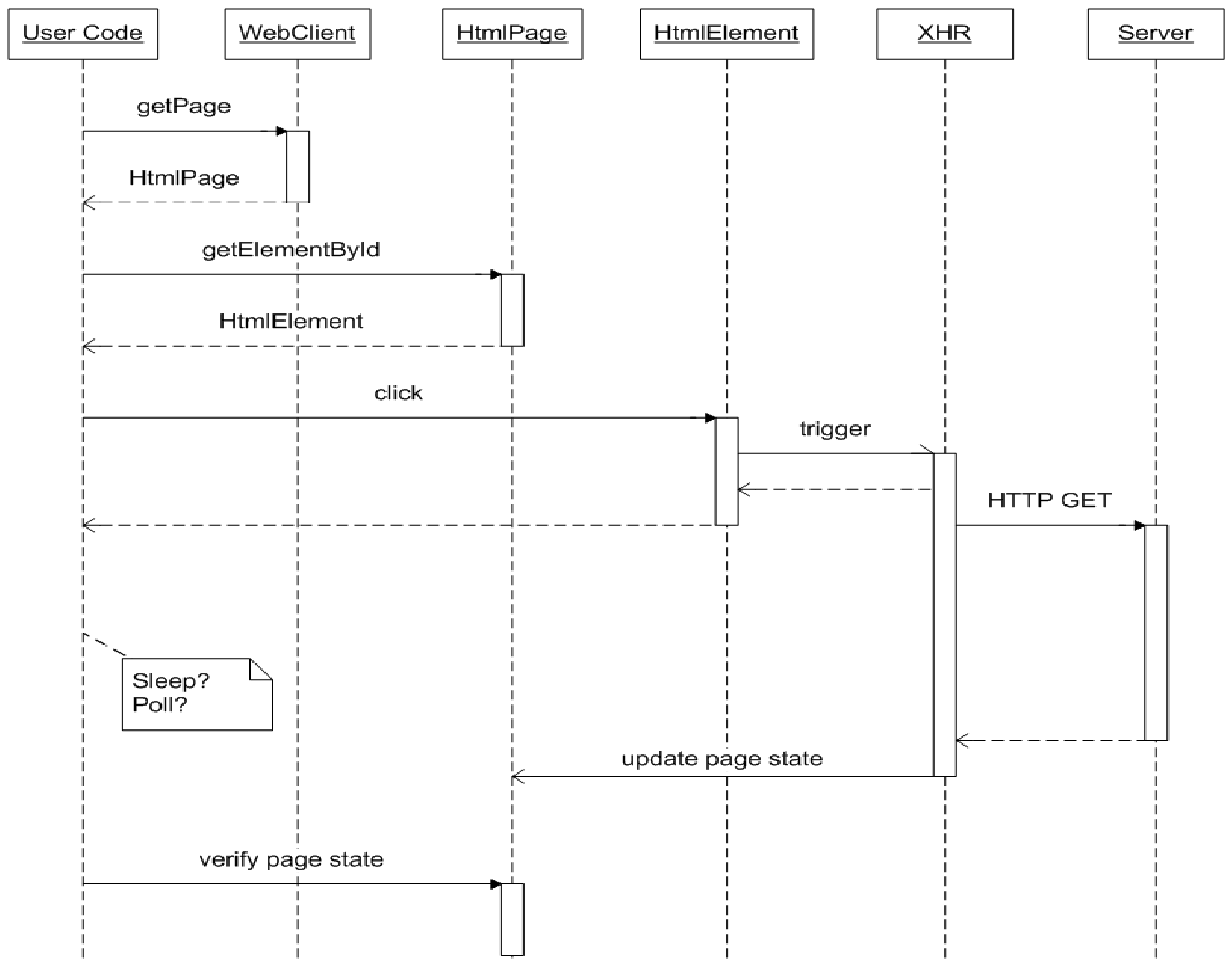
## Extension Point Example

```
final MutableInt invocationCount = new MutableInt(0);

Debugger debugger = new DebuggerAdapter() {
    @Override
    public DebugFrame getFrame(Context cx, DebuggableScript fnOrScript) {
        return new DebugFrameAdapter() {
            @Override
            public void onEnter(Context cx, Scriptable act, Scriptable thiz, Object[] args) {
                invocationCount.increment();
            }
        };
    }
};

WebClient client = new WebClient();
client.getJavaScriptEngine().getContextFactory().setDebugger(debugger);

client.getPage("http://www.google.com/");
System.out.println(invocationCount);
```

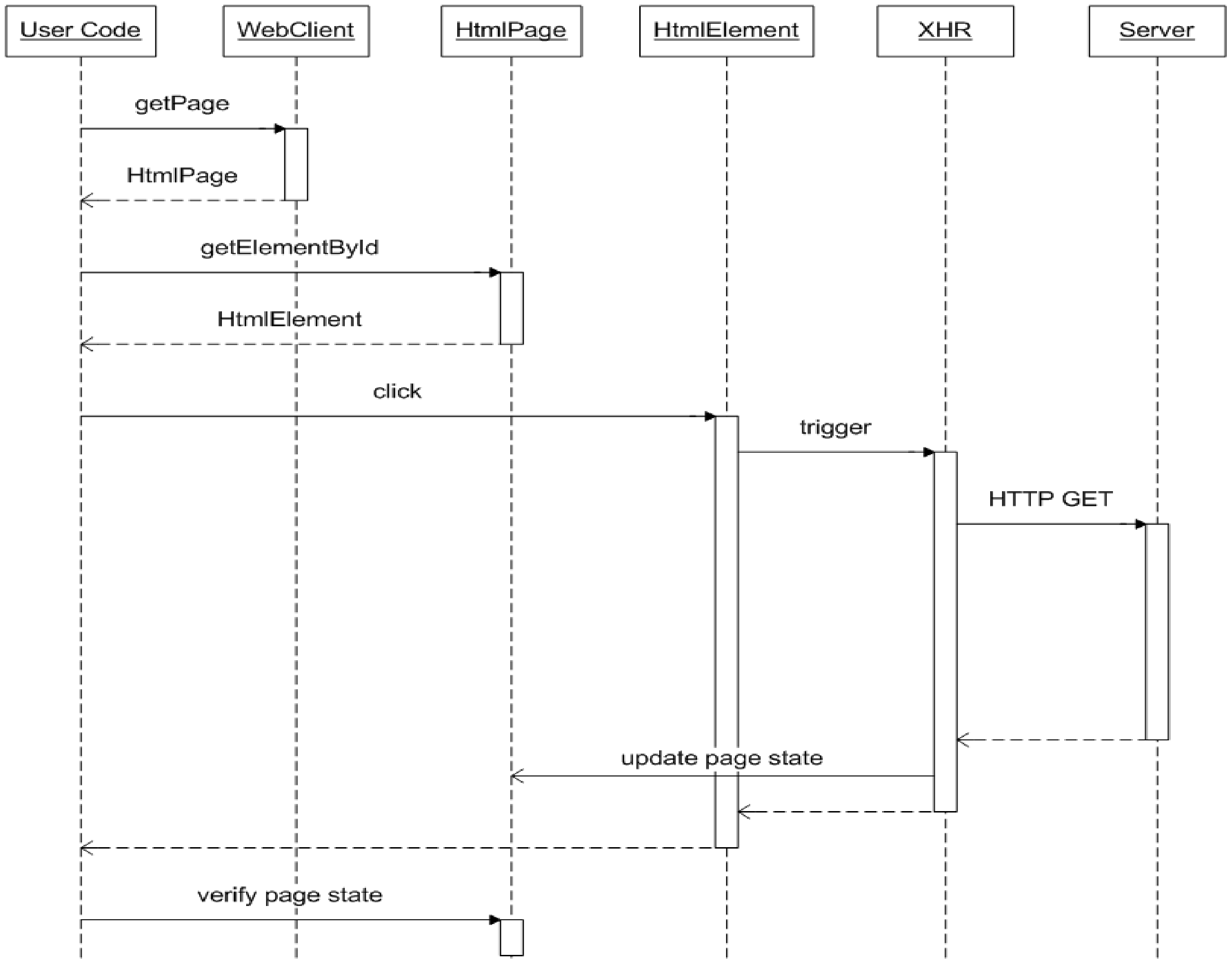
## AJAX Timing

- Need to re-synchronize asynchronous logic
- Basic solutions:
  - Thread.sleep(long)
  - Polling
- HtmlUnit solutions:
  - NicelyResynchronizingAjaxController
  - waitForBackgroundJavaScript(long)
  - waitForBackgroundJavaScriptStartingBefore(long)

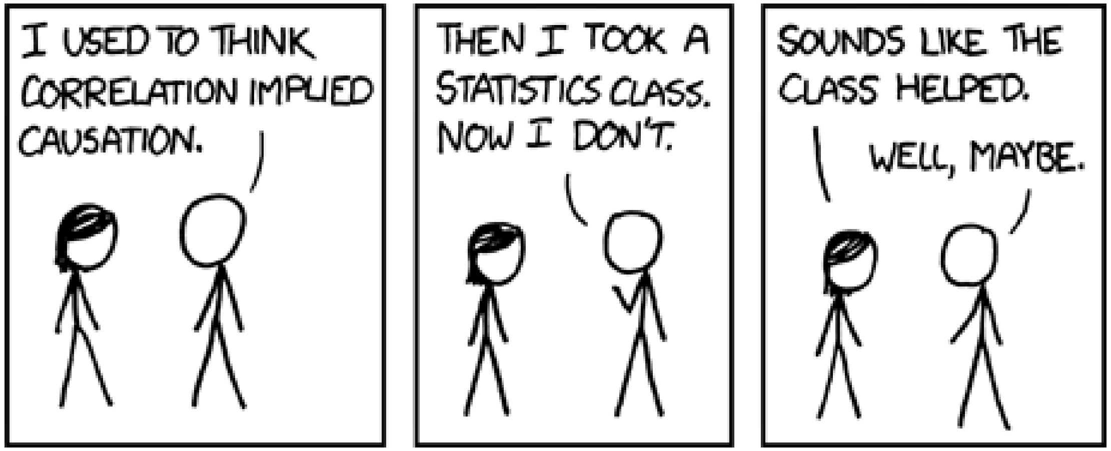
# XMLHttpRequest



# NicelyResynchronizingAjaxController



# Performance


I USED TO THINK CORRELATION IMPLIED CAUSATION.
THEN I TOOK A STATISTICS CLASS. NOW I DON'T.
SOUNDS LIKE THE CLASS HELPED.
WELL, MAYBE.

## Performance

> Reduce network traffic

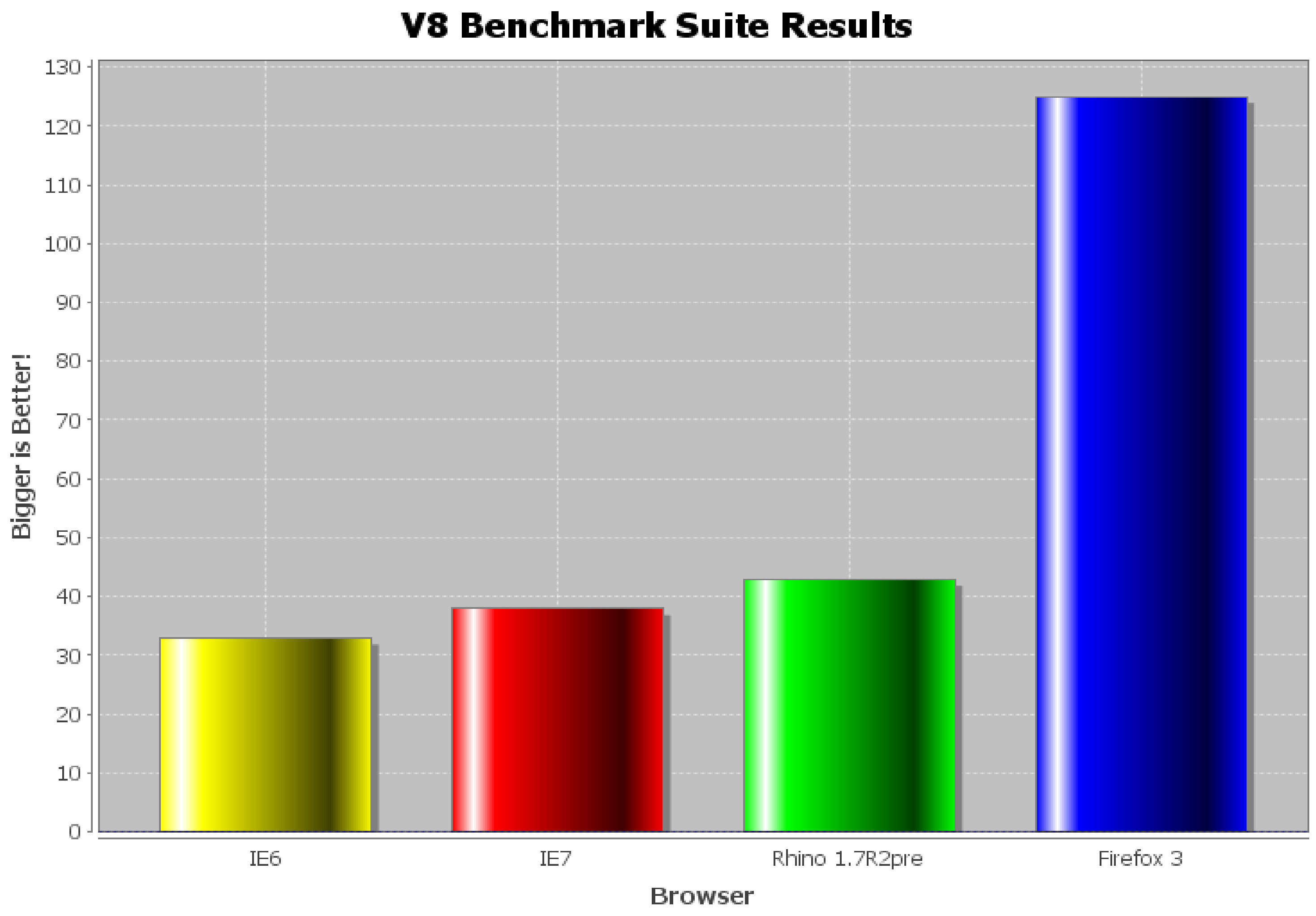
> No rendering

> No browser startup pause

> Data point: Celerity vs. Watir

- Simple local file: test time reduced by 99%
- Google image search: test time reduced by 69%
- Digg front page scraping: test time reduced by 74%
- Local file (DOM access): test time reduced by 97%

## Performance: JavaScript Engines



## Other Advantages

> Platform Independence
- "I'm a PC" developer…
- vs. "I'm a Mac" developer…
- vs. build server…
- vs. continuous integration server

> Scalability
- Standard JVM setup…
- vs. grid component…
- vs. cloud infrastructure

## Limitations

- Simulation is not 100% correct
  - Some malformed HTML
  - Some JavaScript execution
  - On the HTTP layer
- RIAs
  - No support for Flash, Silverlight or JavaFX
  - Applet support is very basic
- No recorder

## Ensuring Accuracy

- We use…
  - Targeted unit tests (HTML+CSS+JS)
  - WebDriver-based unit tests
  - AJAX library integration tests
  - JavaScript execution flow comparisons
  - Over 14,000 tests in all

# Targeted Unit Tests: Example

```
@Test
@Alerts(IE  = {"undefined", "undefined"},
        FF2 = {"[Node]", "[Element]"},
        FF3 = {"[object Node]", "[object Element]"})
public void windowProperties() throws Exception {

    String html =
          "<html>"
        + "<head><script>"
        + "  function test() {"
        + "    alert(window.Node);"
        + "    alert(window.Element);"
        + "  }"
        + "</script></head>"
        + "<body onload='test()'></body>"
        + "</html>";

    loadPageWithAlerts(html);
}
```

## Library Integration Tests

*JavaScript is a language of many contrasts. It contains many errors and sharp edges, so you might wonder, "Why should I use JavaScript?" There are two answers. The first is that you don't have a choice.*

*- Douglas Crockford*

## Library Integration Tests

- Dojo
- Sarissa
- Prototype
- CurvyCorners
- jQuery
- MochiKit
- YUI
- ExtJS
- GWT
- …

## Wrappers

WebDriver

Canoo WebTest

Wepawet

JSFUnit

GWT

Hue Doj

AppPerfect

Grails Functional Testing Plugin

JWebUnit

TestPlan

PushToTest TestMaker

Geb

Ruhu

Steam

Schnell

Perl HtmlUnit

Celerity

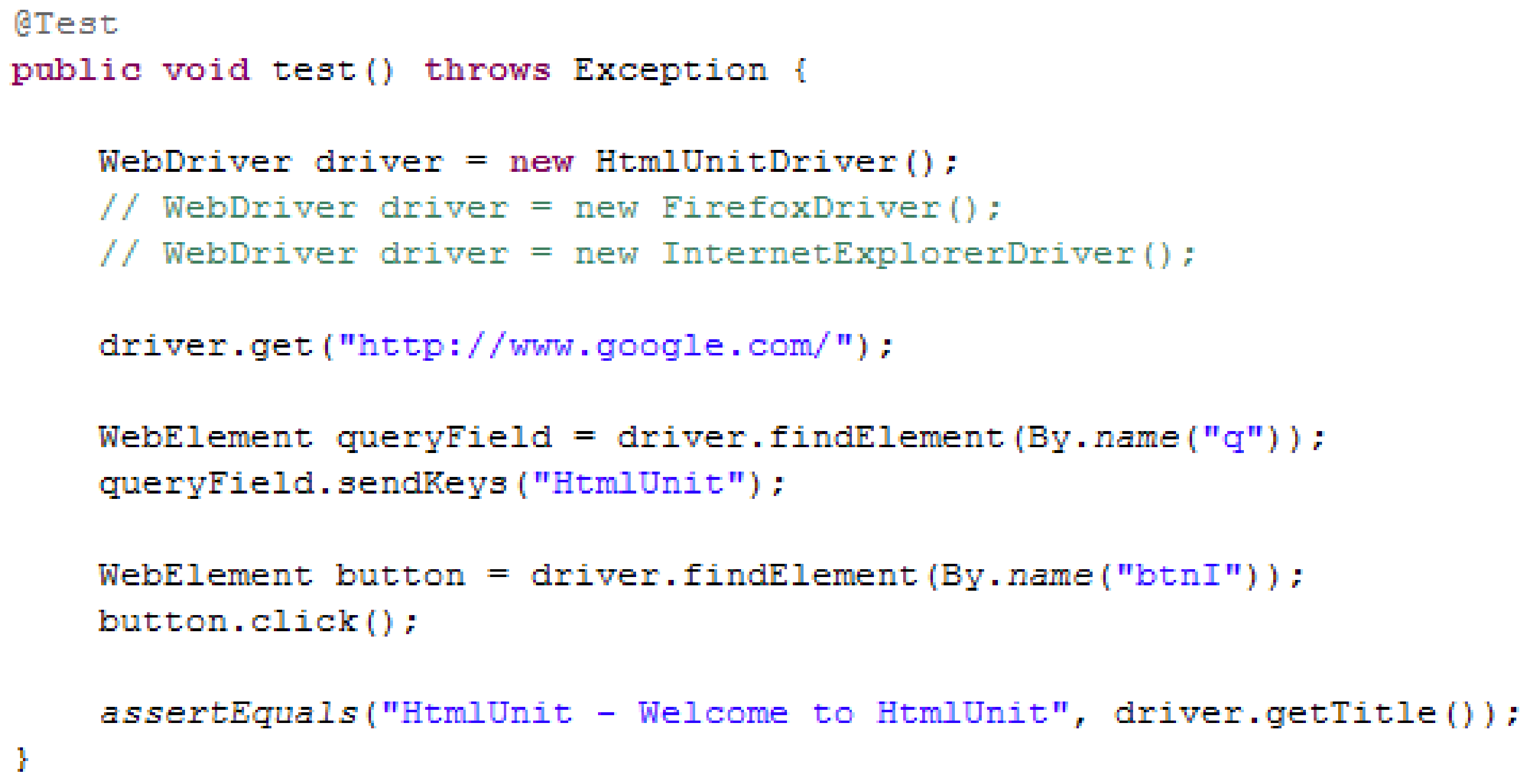
Capybara

Culerity

# Example Wrapper: WebTest

```
import com.canoo.webtest.WebtestCase

class SimpleTest extends WebtestCase {
  void testWebtestOnGoogle() {
    webtest("check that WebTest is Google's top 'WebTest' result") {
      invoke "http://www.google.com/ncr",
         description: "Go to Google (in English)"
      verifyTitle "Google"
      setInputField name: "q", value: "WebTest"
      clickButton "I'm Feeling Lucky"
      verifyTitle "Canoo WebTest"
    }
  }
}
```

## Example Wrapper: WebDriver



```java
@Test
public void test() throws Exception {

    WebDriver driver = new HtmlUnitDriver();
    // WebDriver driver = new FirefoxDriver();
    // WebDriver driver = new InternetExplorerDriver();

    driver.get("http://www.google.com/");

    WebElement queryField = driver.findElement(By.name("q"));
    queryField.sendKeys("HtmlUnit");

    WebElement button = driver.findElement(By.name("btnI"));
    button.click();

    assertEquals("HtmlUnit - Welcome to HtmlUnit", driver.getTitle());
}
```

## Future Plans

- Expand AJAX library integration testing
- Improved control of asynchronous JavaScript
- Support for more browsers?
- JavaScript-thread-per-client model
- GAE compatibility
- JavaScript execution tracing proxy
- HttpClient 4 migration
- Continue releasing frequently!

# Links / Credits

- HtmlUnit: http://htmlunit.sourceforge.net/
- Selenium 2 / WebDriver: http://code.google.com/p/selenium/
- Canoo WebTest: http://webtest.canoo.com/
- JWebUnit: http://jwebunit.sourceforge.net/
- JSFUnit: http://www.jboss.org/jsfunit/
- Push to Test: http://www.pushtotest.com/
- Celerity: http://celerity.rubyforge.org/
- Culerity: http://github.com/langalex/culerity
- Steam: http://github.com/svenfuchs/steam
- Capybara: http://github.com/jnicklas/capybara
- Hans Rosling, TED: http://ted.com/talks/hans_rosling_shows_the_best_stats_you_ve_ever_seen.html
- “Tag Soup”:  http://www.accessibility.nl/internet/artikelen/valideenwebstandaarden
- “IE6?”: http://jimcloudman.tumblr.com/post/427545558/i-just-published-this-as-a-tshirt-design-on
- “Correlation”: http://xkcd.com/552/
- Celerity vs Watir benchmark: http://celerity.rubyforge.org/benchmarks.html