|  |  |
| --- | --- |
| **Đại HỌC QUỐC GIA HÀ NỘI** | |
| **tRƯỜNG đẠI HỌC CÔNG NGHỆ** | |
| **----------------------------------------** | |
|  | |
| **BÁO CÁO CÁC VẤN ĐỀ HIỆN ĐẠI CÔNG NGHỆ PHẦN MỀM** | |
| NHÓM 1 | |
|  | |
| **ĐỀ BÀI: TÌM HIỂU PHANTOMJS** | |
|  | |
| Giảng viên: Trương Anh Hoàng | |
|  | |
|  | Hoàng Kim Băng |
|  | Hoàng Đức Tùng |
|  | Nguyễn Việt Hoàng |
|  | Hà Thanh Loan |
|  |  |
|  |  |
| **Hà Nội, tháng 11 năm 2013** | |

**MỤC LỤC**

[I. Tìm hiểu chung 3](#_Toc373523361)

[1. Tổng quan 3](#_Toc373523362)

[2. Lịch sử phát triển 3](#_Toc373523363)

[3. PhantomJS và NodeJS. 4](#_Toc373523364)

[II. Chức năng chính của PhantomJS 5](#_Toc373523365)

[1. Headless Website Testing 5](#_Toc373523366)

[a. Headless Testing 5](#_Toc373523367)

[b. Không là Test FrameWork 5](#_Toc373523368)

[c. Các dự án 5](#_Toc373523369)

[d. Hệ thống tích hợp liên tục 6](#_Toc373523370)

[2. Screen Capture 6](#_Toc373523371)

[3. Page Automation 9](#_Toc373523372)

[a. Thao tác trên DOM 9](#_Toc373523373)

[b. Sử dụng jQuery và các thư viện khác 10](#_Toc373523374)

[4. Network monitoring 11](#_Toc373523375)

[III. Hệ thống xử lý các tiến trình 12](#_Toc373523376)

[1. File I/O 12](#_Toc373523377)

[a. Files 12](#_Toc373523378)

[b. Sockets 14](#_Toc373523379)

[2. HTTP 14](#_Toc373523380)

[a. outgoing HTTP requests (to other processes/services) (Gửi đi yêu cầu HTTP (quy trình/ dịch vụ khác)) 14](#_Toc373523381)

[b. incoming HTTP requests (server). Use the API Reference WebServer webserver module ( Yêu cầu đến HTTP (server). Sử dụng API Reference WebServer modun máy chủ web) 15](#_Toc373523382)

[3. Websockets inside a WebPage context 15](#_Toc373523383)

[4. Hiển thị dòng lệnh 15](#_Toc373523384)

[IV. API Reference 16](#_Toc373523385)

[1. Tùy chọn dòng lệnh 16](#_Toc373523386)

[2. Object 18](#_Toc373523387)

[a. Propeties 18](#_Toc373523388)

[b. Functions 19](#_Toc373523389)

[3. CallBacks 20](#_Toc373523390)

[V. Bắt đầu với PhantomJS. 21](#_Toc373523391)

[1. Cài đặt 21](#_Toc373523392)

[2. Ví dụ 24](#_Toc373523393)

[a. Ví dụ về zing.js 24](#_Toc373523394)

[b. Ví dụ về weather.js 26](#_Toc373523395)

[c. Ví dụ về server.js 28](#_Toc373523396)

[VI. Ứng dụng thực tế của PhantomJS 31](#_Toc373523397)

[1. YSlow 31](#_Toc373523398)

[a. YSlow là gì? 31](#_Toc373523399)

[b. YSlow và PhantomJS. 31](#_Toc373523400)

[c. Ví dụ 32](#_Toc373523401)

[2. Qunit 36](#_Toc373523402)

[3. PhantomJS với Python 36](#_Toc373523403)

[VII. Tổng kết 38](#_Toc373523404)

Tài liệu tham khảo

**TÌM HIỂU PHANTOMJS**

# Tìm hiểu chung

## Tổng quan

Trước khi hiểu PhantomJS là gì thì chúng ta sẽ phải hiểu Webkit là gì? Có thể hiểu Webkit là một khung ứng dụng cung cấp đầy đủ các thành phần để lập trình nên một trình duyệt Web. Có nghĩa là nền tảng để tạo nên các trình duyệt như FireFox, Chrom, Safari … chính là Webkit.

Vậy PhantomJS là gì? Nó là một trình duyệt Web nhưng là một trình duyệt không đầu. Có nghĩa là các kết quả được trả về trên Web sẽ không được hiển thị rõ ràng mà PhantomJS sẽ cung cấp các kịch bản, cú pháp để người dùng thao tác trực tiếp với Webkit và trả về kết quả mong muốn.

Ngôn ngữ tạo lên PhantomJS là JavaScript, một ngôn ngữ thực sự quen thuộc với mỗi nhà lập trình.

PhantomJS là một mã nguồn mở, nó hỗ trợ các chuẩn Web: DOM handling, CSS selector, JSON, Canvas và SVG.

## Lịch sử phát triển

PhantomJS là một dự án mã nguồn mở của Ariya Hidayat, một người sinh ra tại Indonesia. Cùng với Esprima, PhantomJS là một trong hai dự án lớn nhất của ông. Ông có trên 30 bài báo, bằng tiến sĩ tại đại học Paderborn của Đức.

Đến nay PhantomJS vẫn không ngừng phát triển và có đã trải qua khoảng chín phiên bản.

Phiên bản chính thức đầu tiên là PhantomJS 1.1 phát hành vào ngày 27 tháng 4 năm 2011 và được Ariya Hidayat đặt tên là Cherry Blossom. Có một điều khá thú vị của Ariya Hidayat đó là mỗi phiên bản phát hành của ông đều được đặt một tên riêng và chúng đều mang một ý nghĩa riêng. Ví dụ như cái tên Cherry Blossom, đó là tên một loại hoa nở vào mùa xuân của đất nước Nhật Bản, từ đó ông gửi sự trân trọng đến với người Nhật vì sự quan tâm của họ với sản phẩm của ông và muốn PhantomJS phát triển mạnh mẽ như mùa xuân.

Phiên bản mới nhất tính đến nay là bản PhantomJS 1.9.2 được phát hành vào ngày 20 tháng 3 năm 2013, có thể thấy PhantomJS vẫn luôn được phát triển với các phiên bản được tung ra khá đều đặn.

Mỗi phiên bản mới của PhantomJS là sự nâng cấp của phiên bản trước. Nâng cấp ở đây chủ yếu là sửa lỗi phiên bản trước và thêm các tính năng mới do cộng đồng người dùng tìm hiểu, phát triển. Để hiểu rõ sự nâng cấp nay chúng ta sẽ tìm hiểu về sự thay đổi của phiên bản 1.9 so với 1.8. Đã có rất nhiều tính năng và cải tiến mới của bản PhantomJS 1.9 mà có thể kể đến là:

* Thêm khả năng hủy bỏ yêu cầu mạng.
* Thêm chức năng tìm kiếm các dòng.
* Thêm hỗ trợ mã hóa tùy chỉnh.
* Thêm chỉ số tải trang và tiến độ.
* Cố định xây dựng trên FreeBSD.
* Sửa lỗi lưu sai ngày trong các tệp tin cookie.
* ...

Trên đây là một số tính năng mới của bản PhantomJS 1.9 so với bản 1.8. Bản 1.9 cũng sẽ là bản cuối cùng trong chuỗi 1.x với hứa hẹn các phiên bản 2.x mang đến tính đột phá mới về nền tảng cơ bản hiện đại hơn cùng một module Webkit cập nhật.

## PhantomJS và NodeJS.

NodeJS là một khung mã nguồn mở cung cấp các tiện ích để phát triển các ứng dụng trên nền Web sử dụng JavaScript. NodeJS ra đời trước PhantomJS và có một vài đặc điểm đặc trưng như: ngôn ngữ tiện dụng, phổ biến; tốc độ nhanh (nhờ V8); nhiều thư viện, dễ cài; chia thành các module để dễ quản lý…

Sau khi phát hành PhantomJS thì ngay lập tức có một câu hỏi được đặt ra: Tại sao không xây dựng PhantomJS thành một module của NodeJS, điều này sẽ giúp cải thiện hệ thống (vì nếu sử dụng cả hai thì sẽ phải cài đặt cả nền tảng PhantomJS và NodeJS) cũng như dễ quản lý hơn với người dùng?

Nền tảng của vấn đề rất hợp lý. Thứ nhất, PhantomJS và NodeJS đều là khung ứng dụng cho Website. Thứ hai chúng đều sử dụng JavaScript. Thứ ba PhantomJS ít phổ biến hơn so với NodeJS. Nhưng tại sao lại không xây dựng PhantomJS thành một module của NodeJS?

Có thể lý giải vấn đề này như sau: đầu tiên NodeJS không hỗ trợ các chức năng cần thiết khi xây dựng PhantomJS. Thứ hai, các module trong Node để chạy được thì sẽ phải nạp vào “server Node” mà trong khi đó PhantomJS cần phải có quyền kiểm soát cho mọi thứ như: vòng lặp sự kiện, lưu lượng mạng khả năng biên dịch mã JavaScript,…

Trên đây là sự khác biệt cơ bản giữa NodeJS và PhantomJS.

# Chức năng chính của PhantomJS

## Headless Website Testing

### Headless Testing

Chức năng đầu tiên của PhantomJS là kiểm thử không đầu (Headless Testing). Nó phù hợp để kiểm thử từng dòng lệnh và một phần của hệ thống tích hợp liên tục.

### Không là Test FrameWork

PhantomJS không phải là một test framework, nó chỉ được sử dụng để khởi tạo các cuộc kiểm tra thông qua các test thử nghiệm phù hợp.

### Các dự án

Có một số dự án được xây dựng trên nền PhantomJS để cung cấp các chức năng thuận tiện cho quá trình kiểm thử:

* CasperJS: phù hợp cho việc xây dựng các kịch bản chuyển hưởng và kiểm thử.
* Lotte: thêm phương thức jQuery-like, chuỗi và nhiều khẳng định logic.
* WebSpecter là một BDD-style (behavior-driven development) cho phép kiểm thử framework của ứng dụng Web.

### Hệ thống tích hợp liên tục

* PhantomJS có thể cài đặt với hệ thống CI (Continuous Integration). Một hệ thống tổ chức CI nổi tiếng có tích hợp sẵn PhantomJS là Travis CI.
* PhantomJS cũng có thể tích hợp trên Linux.

## Screen Capture

Webkit là một phương tiện layout và reandering thực, vì vậy PhantomJS có khả năng chụp ảnh trang Web như chụp ảnh màn hình. Các nội dung được chụp có thể là trang Web tĩnh hoặc được truy xuất từ HTML, CSS, SVG và Canvas.

Đoạn mã dưới đây sẽ chụp lại ảnh một trang Web đơn giản. Nó tải trang chủ của Github và lưu lại thành tập tin ảnh: *giphub.png*.

*var page = require(‘webpage’).create();*

*page.open(‘*[*http://github.com/*](http://github.com/)*’, function() {*

*page.render(‘github.png’);*

*phantom.exit();*

*});*

Hình 1. Mã nguồn chụp ảnh Website “github.com”

* *Cách chạy đoạn mã trên sẽ được trình bày chi tiết ở phần Cài đặt.*

Ở ví dụ trên định dạng ảnh được sử dụng là đuôi .PNG, ngoài định dạng này ra PhantomJS cũng hỗ trợ thêm các định dạng khác nữa như: GIF, JPEG và PDF.

Để chứng minh khả năng dựng hình hoàn chỉnh của PhantomJS thì ta sẽ xét thêm một vài ví dụ.

Ví dụ đầu tiên là đoạn mã [rasterize.js](https://github.com/ariya/phantomjs/blob/master/examples/rasterize.js) trong thư mục example, nó sẽ truy vấn SVG và dựng hình ảnh con hổ khi được biên dịch.

Câu lệnh biên dịch: *Phantomjs rasterize.js* [*http://ariya.github.com/svg/tiger.svg*](http://ariya.github.com/svg/tiger.svg) *tiger.png*

Kết quả được hiển thị là ảnh con hổ lưu trong bức ảnh *tiger.png:*

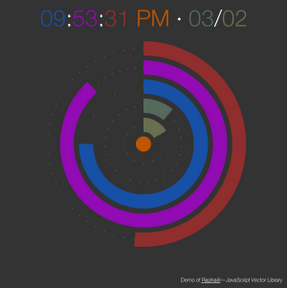


Hình 2. Tiger.png

Ví dụ tiếp theo là truy vấn HTML vẫn qua đoạn mã nguồn [rasterize.js](https://github.com/ariya/phantomjs/blob/master/examples/rasterize.js). Chương trình thực hiện truy vấn HTML và dựng hình một chiếc đồng hồ thời gian thực. Kết quả được xuất ra file ảnh *clock.png* với thời gian tại thời điểm truy vấn (Hình 3).

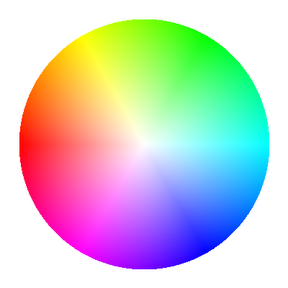
Câu lệnh biên dịch: *Phantomjs rasterize.js* [*http://raphaelis.com/polar-clock.html*](http://raphaelis.com/polar-clock.html) *clock.png*

Và đây là kết quả thu được:



Hình 3. Clock.png

Ví dụ cuối cùng sẽ là về Canvas. Canvas cũng có thể dễ dàng truy xuất và xây dựng thành hình ảnh. Đoạn mã [colorwheel.js](https://github.com/ariya/phantomjs/blob/master/examples/colorwheel.js) sẽ tạo ra bánh xe màu dưới đây:



Hình 4. Colorwheel.png

Qua các ví dụ trên chúng ta có thể thấy khả năng dựng hình từ các trang Web của PhantomJS. Đã có vài dự án dựa vào PhantomJS để hỗ trợ dựng hình như: [Screener](http://screener.brachium-system.net/) và [ChromaNope.](http://chromanope.com/)

## Page Automation

PhantomJS có thể tải và thao tác trên một trang Web như một giao diện duyệt Web bình thường. Nhờ vậy PhantomJS có thể thực hiện các thao tác tự động hóa khác nhau.

### Thao tác trên DOM

Khi các dữ liệu được hiển thị trên các trình duyệt Web cũng có nghĩa là dữ liệu từ DOM hay CSS đã hoạt động tốt, không xảy ra mất mát dữ liệu.

Ví dụ sau đây sẽ in ra thuộc tính của phần tử có id là myagent khi được biên dịch. Đoạn mã *useragent.js*

*var page=require(‘webpage’).create();*

*console.log(‘The default user agent is’ + page.settings.userAgent);*

*page.settings.userAgent=’SpecialAgent’;*

*page.open(‘http://www.httpuseragent.org’, function(status){*

*if(status!=’success’){*

*console.log('Unable to access network');*

*}* ***else*** *{*

***var*** *ua* ***=*** *page.evaluate(****function*** *() {*

***return*** *document.getElementById('myagent').textContent;*

*});*

*console.log(ua);*

*}*

*Phantom.exit();*

*});*

Hình 5. Mã nguồn useragent.js

Ví dụ trên cho thấy khả năng tùy chỉnh các nhân viên (user agent) từ xa.

### Sử dụng jQuery và các thư viện khác

Sử dụng page.includeJs để thao tác với jQuery

*var page=require(‘webpage’).create();*

*page.open(‘http://www.sample.com’, function(){*

*page.includeJs(*[*http://ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.min.js*](http://ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.min.js)*, function(){*

*page.evaluate(function(){*

*$(“button”).click();*

*}):*

*Phantomjs.exit();*

*});*

*});*

Hình 6. Mã nguồn thể hiển chức năng thao tác với JQuery

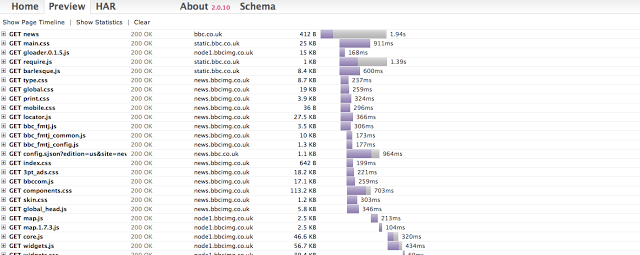
Đoạn mã trên sẽ tải trang Web [*www.sample.com*](http://www.sample.com)gồm có các jQuery và bấm vào tất cả các nút có sử dụng jQuery. Sau đó nó sẽ thoát khỏi trang Web.

## Network monitoring

PhantomJs cho phép kiểm tra lưu lượng mạng, từ đó xây dựng và phân tích các hành vi và hiệu suất mạng.

Sử dụng *onResourceRequested* và *onResourceReceived* để tải các tài nguyên và thực hiện phân tích lưu lượng mạng.

Sau đây là một sơ đồ thác nước thu được từ trang Web BBC:



Hình 7. Biểu đồ lưu lượng mạng của trang Web BBC

Có một vài dự án liên quan đến PhantomJS giúp phân tích hiệu suất mạng hiệu quả hơn, điển hình là hai dự án Confess.js và YSlow. Trong đó Yslow rất hữu ích cho việc giám sát hiệu năng Web tự động.

# Hệ thống xử lý các tiến trình

Có nhiều cách để xử lý các quá trình liên lạc giữa PhantomJS và các tiến trình khác. Ở đây chúng ta tìm hiểu bốn cách chính là sử dụng file I/O; HTTP; Websocket (Websockets inside a WebPage context) và hiện thị dòng lệnh (stdin, stdout, stderr).

## File I/O

### Files

***Module:Filesystem:***

Một tập hợp các chức năng API có sẵn để truy cập các tệp tin và thư mục, theo mô hình đề nghị hệ thống tệp tin CommonJS .

Để bắt đầu sử dụng , bạn phải **require** một tham chiếu đến mô đun **fs**

var fs = require('fs');

***Propertie:***

* separator {String}: Phân cách trong đường dẫn (/ hoặc \ tùy thuộc vào hệ điều hành).
* workingDirectory {String}: Thư mục làm việc hiện tại.

***Functions:***

* Query Functions
* list(path): Trả về danh sách các file trong đường dẫn được chỉ định.
* absolute(path): Trả về đường dẫn tuyệt đối bắt đầu từ thư mục gốc, chuyển từ thư mục làm việc hiện tại
* exists(path): Trả về true nếu thư mục hoặc tập tin đã tồn tại
* isDirectory(path): Trả về true nếu đường dẫn được chỉ định là một thư mục
* isFile(path): Trả về true nếu đường dẫn được chỉ định là một file.
* isAbsolute(path): Trả về true nếu đường dẫn được chỉ định là một đường dẫn tuyệt đối.
* isExecutable(path): Trả về true tập tin được chỉ định có thế được thực hiện
* isReadable(path: Trả về true nếu tập tin hoặc thư mục có thể đọc được.
* isWritable(path): Trả về true nếu tập tin hoặc thư mục có thể viết được.
* isLink(path): Trả về true nếu đường dẫn chỉ định là một liên kết tượng trưng.
* readLink(path): Trả về đích của liên kết tượng trưng.
* Directory Functions
* changeWorkingDirectory(path): Thay đổi thư mục làm việc hiện tại đến đường dẫn được chỉ định
* makeDirectory(path): Tạo một thư mục mới.
* makeTree(path): Tạo ra một thư mục bao gồm các thư mục cha bị mất.
* removeDirectory(path): Xóa một thư mục nếu nó rỗng
* removeTree(path): Xóa đường dẫn chỉ định, kể cả file hay thư mục.
* copyTree(source, destination): Sao chép các file từ nguồn tới đích.
* File Functions
* open(path, mode): Trả về luồng các đối tượng thể hiện cho luồng giao diện đến file chỉ định(có thể dùng ‘r’ để đọc, ‘w’ để ghi, ‘a’ để thêm vào sau, ‘rb’ để đọc dưới dạng nhị phân, ‘wb’ để viết dưới dạng nhị phân.
* read(path): Trả về toàn bộ nội dung của tệp tin.
* write(path, content, mode): Viết nội dung vào file (có thể dùng ‘w’ để viết, 'a' để thêm vào sau hoặc 'wb' ghi dưới dạng nhị phân).
* size(path): Trả về kích thước của file được chỉ định bởi đường dẫn.
* remove(path): Xóa file được chỉ định bởi đường dẫn.
* copy(source, destination): Sao chép tới một file khác.
* move(source, destination): Di chuyển file tới thư mục khác và đổi tên nó.
* touch(path): Tạo file.
* Stream object

Một luồng các đối tượng được trả về từ hàm fs.open() có các phương thức sau :

* read(): Trả về nội dung của luồng.
* readLine(): Dọc một dòng từ luồng và trả về nó.
* atEnd(): Trả về true nếu đã hết tập tin.
* write(data): Viết các chuỗi vào luồng.
* writeLine(data): Ghi dữ liệu như một dòng vào luồng.
* seek(pos): Di chuyển đến mọt vị trí trong luồng.
* flush(): Chuyển vào/ra.
* close(): Hoàn thành các thao tác luồng.

### Sockets

PhantomJS cũng sử dụng sockets như một thiết bị tạo kết nối các tiến trình.

## HTTP

### outgoing HTTP requests (to other processes/services) (Gửi đi yêu cầu HTTP (quy trình/ dịch vụ khác))

* GET/POST dữ liệu đến thiết bị đầu cuối.
* GET/POST dữ liệu đến thiết bị đầu cuối, phân tích kết quả JSON/XML/HTML/etc.
* Tạo một đối tượng XMLHttpRequest và sử dụng nó một trang web bình thường.
* Yêu cầu Cross-domain được giới hạn mặc định bởi (PhantomJS code is considered to be in file:// scope). Dùng [Access-Control-Allow-Origin](http://www.w3.org/TR/cors/#access-control-allow-origin-response-header) để cho phép bạn truy cập.

### incoming HTTP requests (server). Use the [API Reference WebServer](https://github.com/ariya/phantomjs/wiki/API-Reference-WebServer) webserver module ( Yêu cầu đến HTTP (server). Sử dụng [API Reference WebServer](https://github.com/ariya/phantomjs/wiki/API-Reference-WebServer) modun máy chủ web)

## Websockets inside a WebPage context

WebSockets của PhantomJS sử dụng giao thức Hixie-76. Đây là một giao thức của socket với tên gọi đầy đủ là Giao thức WebSocket draft-hixie-thewebsocketprotocol-76

Giao thức WebSocket cho phép giao tiếp hai chiều giữa một người sử dụng  
tác nhân chạy đoạn mã không tin cậy chạy trong một môi trường có kiểm soát một máy chủ từ xa tham gia vào giao tiếp với mã . Các mô hình bảo mật được sử dụng ở đây là các mô hình bảo mật thường được sử dụng bởi các trình duyệt web . Giao thức bao gồm: ban đầu là bắt tay, tiếp theo là các khung thông điệp cơ bản, tầng trên TCP . Các mục tiêu của công nghệ này là để cung cấp một cơ chế dựa trên trình duyệt các ứng dụng cần giao tiếp hai chiều với các máy chủ mà không dựa trên mở nhiều kết nối HTTP (ví dụ như sử dụng XMLHttpRequest hoặc <iframe> s và bỏ phiếu dài ) .

## Hiển thị dòng lệnh

Đây là một cơ chế khá đơn giản và phổ biến. Nó chỉ đơn thuần là tiến trình giúp người phát triển ứng dụng hiển thị các thông tin trong đoạn mã nguồn nào đó lên màn hình command line. Qua đó giúp họ kiểm soát được công việc mình đang làm.

Để hiểu chi tiết hơn chúng ta cùng hiểu vài ví dụ cơ bản sau:

* Stdout: hiển thị thông tin tức thời lên màn hình. Ví dụ:

*system.stdout.write('Hello, PhantomJS!');*

Đoạn mã nguồn trên đơn giản là hiển thị đoạn văn bản “hello, PhantomJS!” lên màn hình.

* Stderr: hiển thị chú ý hoặc một lỗi nào đó lên màn hình. Nó cũng tương tự như Stdout chỉ khác là dòng hiển thị sẽ nổi bật hơn bằng màu sắc hoặc cách gạch chân.
* Stdin: đọc thông tin từ một nguồn nào đó rồi hiển thị lên màn hình. Nguồn ở đây có thể hiểu là từ JSON, DOM hay Jquery, ....

# API Reference

PhantomJS giả định được xây dựng và thực thi của nó là nơi ở đâu đó trong đường dẫn , nó có thể được gọi như sau: phantomjs [ options ] somescript.js [ arg1 [ arg2 [ ... ] ] ]

Các mã script sẽ được thực hiện như thể nó đang chạy trong một trình duyệt web với một trang rỗng . Từ PhantomJS là không đầu , sẽ không có bất cứ điều gì có thể nhìn thấy hiện lên trên màn hình.

Nếu PhantomJS được gọi mà không có bất kỳ đối số, nó sẽ vào chế độ tương tác ( REPL )

## Tùy chọn dòng lệnh

Tuỳ chọn dòng lệnh hỗ trợ:

* --help or -h : Liệt kê tất cả các tùy chọn dòng lệnh có thể. *Ngừng ngay lập tức, sẽ không chạy một kịch bản thông qua như là đối số*
* --version or -v : In ra phiên bản của PhantomJS. *Ngừng ngay lập tức, sẽ không chạy một kịch bản thông qua như là đối số*
* --cookies-file=/path/to/cookies.txt : Xác định tên tập tin để lưu các [Cookies](https://github.com/ariya/phantomjs/wiki/API-Reference#wiki-cookie).
* --disk-cache=[true|false] : cho phép bộ nhớ cache (at desktop services cache storage location, mặc định là false). Chấp nhận: [yes|no].
* --ignore-ssl-errors=[true|false] : Bỏ qua lỗi SSL, chẳng hạn như lỗi chứng nhận hết hạn hoặc ký tự (mặc định là false). Chấp nhận: [yes|no].
* --load-images=[true|false] : Tải tất cả các hình ảnh (mặc định là true). Chấp nhận: [yes|no].
* --local-storage-path=/some/path : Đường dẫn để lưu nội dung LocalStorage và nội dung WebSQL.
* --local-storage-quota=number :Kích thước tối đa cho phép với dữ liệu.
* --local-to-remote-url-access=[true|false] : cho phép nội dung địa phương để truy cập URL từ xa (mặc định là false). Chấp nhận: [yes|no].
* --max-disk-cache-size=size : Giới hạn kích thước cho bộ nhớ cache (in KB).
* --output-encoding=encoding :Thiết lập bảng mã cho vào ra (mặc định là utf8).
* --remote-debugger-port : Bắt đầu gỡ lỗi và lắng nghe trên cổng được chỉ định
* --remote-debugger-autorun : Chạy để sửa lỗi ngay lập tức: ‘có’ hoặc ‘không’ (mặc định)
* --proxy=address:port : Xác đinh các máy chủ proxy để sử dụng (ví dụ. --proxy=192.168.1.42:8080).
* --proxy-type=[http|socks5|none] : Quy định các loại máy chủ proxy (mặc định là http).
* --proxy-auth : Quy định cụ thể các thông tin xác thực cho proxy, ví dụ. --proxy-auth=username:password).
* --script-encoding=encoding : Thiết lập bảng mã để bắt đầu (mặc định là utf8).
* --ssl-protocol=[sslv3|sslv2|tlsv1|any'] : Thiết lập giao thức SSL cho kết nối an toàn (mặc định là SSLv3).
* --ssl-certificates-path=<val> : Thiết lập vị trí cho CA (Nếu như không thiết lập, hệ thống sẽ sử dụng mặc định).
* --web-security=[true|false] : Cho pháo bảo mật web và cấm cross-domain XHR (default is true). Chập nhận chọn: [yes|no].
* --webdriver starts in 'Remote WebDriver mode' (embedded GhostDriver): '[[:]]' (default '127.0.0.1:8910')
* --webdriver-selenium-grid-hub URL to the Selenium Grid HUB: 'URL\_TO\_HUB' (default 'none') (NOTE: works only together with '--webdriver'

Ngoài ra, kể từ PhantomJS 1.3, bạn cũng có thể sử dụng một Object Notation (JSON ) tập tin cấu hình JavaScript thay vì đi qua trong nhiều tùy chọn dòng lệnh:

* config=/path/to/config.json

Nội dung của config.json phải là một đối tượng JavaScript độc lập. Phím được de-dashed, camel-cased tương đương của các tùy chọn dòng lệnh hỗ trợ khác (excluding --version/-v and --help/-h). Giá trị tương đương JavaScript: 'true'/'false' (or 'yes'/'no') giá trị dịch trong Boolean là đúng/sai, số các số, chuỗi các chuỗi. Ví dụ:

*{*

*/\* Same as: --ignore-ssl-errors=true \*/*

*"ignoreSslErrors": true,*

*/\* Same as: --max-disk-cache-size=1000 \*/*

*"maxDiskCacheSize": 1000,*

*/\* Same as: --output-encoding=utf8 \*/*

*"outputEncoding": "utf8"*

*/\* etc. \*/*

*}*

## Object

Giao diện PhantomJS với chức năng khác nhau được thực hiện bằng cách sử dụng một đối tượng máy chủ mới có tên phantom , thêm vào như là con của cửa sổ đối tượng. Các tính chất và chức năng của các đối tượng phantom được mô tả trong các phần sau:

### Propeties

#### args {String[]}:

Sự ổn định: phản đối - Sử dụng system.args từ các mô-đun hệ thống  
Chỉ đọc. Một mảng của các đối số được truyền vào kịch bản.

#### cookes {[Cookie](https://github.com/ariya/phantomjs/wiki/API-Reference-phantom#wiki-cookie)[]}:

Giới thiệu: PhantomJS 1.7

Nhận hoặc thiết lập cookies cho bất kỳ tên miền (tuy nhiên, để thiết lập, sử dụng phantom.addCookie được ưa thích) . Những cookie được lưu trữ trong CookieJar và sẽ được cung cấp khi mở các trang web thích hợp . Mảng này được đưa đến trước bất kỳ dữ liệu Cookie hiện được lưu giữ trong các tập tin cookie được chỉ định trong các tùy chọn cấu hình / dòng lệnh khởi động PhantomJS , nếu có.

#### cookiesEnabled {Boolean}:

**Giới thiệu: PhantomJS 1.7**  
Điều khiển CookieJar được kích hoạt hay không. Mặc định là true.

#### LibraryPath {String}:

LibraryPath lưu trữ các đường dẫn được sử dụng bởi chức năng injectJs để giải quyết tên kịch bản . Ban đầu nó được thiết lập vị trí của kịch bản bởi PhantomJS

#### Version {Object}

Chỉ đọc. Các phiên bản ví dụ của PhantomJS thực hiện. Ví dụ: { 'major': 1, 'minor': 7, 'patch': 0 }.

### Functions

* **addCookie([Cookie](#cookie)) {Boolean}**

**Giới thiệu: PhantomJS 1.7**

Thêm một Cookie với CookieJar . Trả về true nếu thêm thành công, nếu không sai . Xem phantom.cookies để biết thêm thông tin về CookieJar.

Ví dụ:

*phantom.addCookie({*

*'name': 'Added-Cookie-Name',*

*'value': 'Added-Cookie-Value',*

*'domain': '.google.com'*

*});*

* **clearCookies() {void}**

**Giới thiệu: PhantomJS 1.7**

Xóa tất cả các cookie trong CookieJar . Xem phantom.cookies để biết thêm thông tin về CookieJar.

* **deleteCookie(cookieName) {Boolean}**

**Giới thiệu: PhantomJS 1.7**

Xóa bất kỳ cookie trong CookieJar với một " tên " là cookieName phù hợp. Trả về true nếu xóa thành công , nếu không sai . Xem phantom.cookies để biết thêm thông tin về CookieJar

Ví dụ:

*phantom.deleteCookie('Added-Cookie-Name');*

* **exit(returnValue) {void}**

Thoát khỏi chương trình với giá trị trả về chỉ định. Nếu không có giá trị trả lại được chỉ định, nó được thiết lập là 0.

* **injectJs(filename) {boolean}**

Thêm mã kịch bản từ các tập tin được chỉ định vào không gian bên ngoài Phantom . Nếu các tập tin không thể được tìm thấy trong thư mục hiện hành , libraryPath được sử dụng để tìm kiếm thêm . Chức năng này trả về true nếu thêm là thành công , ngược lại trả về sai.

## **CallBacks**

Phản hồi này được gọi khi có một lỗi thực thi JavaScript không bị bắt bởi một bộ xử lý khác onerror WebPage . Sự đóng lại này nó được cho là có một lỗi xử lý toàn cầu trong PhantomJS , và vì vậy nó là một thực hành tốt nhất để thiết lập xử lý onError này lên để bắt bất kỳ vấn đề bất ngờ. Các đối số truyền cho phản hồi là những thông báo lỗi và ngăn xếp vết [ như một mảng ].

# Bắt đầu với PhantomJS.

## Cài đặt

Như bao nhiêu ứng dụng khác, để sử dụng được PhantomJS thì chúng ta cũng cần phải cài đặt biến môi trường của nó trong máy tính của mình.

Sau đây là các bước cài đặt cho PhantomJS:

**Bước 1**: Lên trang chủ của PhantomJS (phantomjs.org) tải bản mới nhất về. Chú ý tải bản tương ứng với hệ điều hành của máy tính.

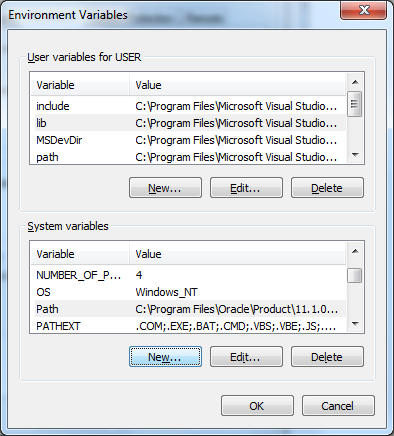
Phiên bản dùng ở ví dụ này là bản PhantomJS 1.9.2 và dùng cho Window.

**Bước 2**: Cài đặt.

Ngoài màn hình máy tính chúng ta bấm chuộc phải vào biểu tượng **Computer/ chọn Properties.**

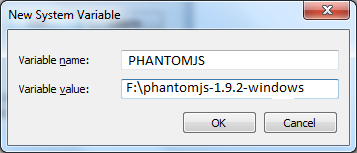
Một cửa sổ mới hiện lên, chúng ta lần lượt đi đến các đường dẫn sau: **Advanced System Settings** -> chọn tab Advanced -> chọn **Environment Variables.**

Trong cửa sổ **Environment variables** chọn mục **System variables rồi** chọn button **New**.



Hình 8. Hướng dẫn cài đặt PhantomJS

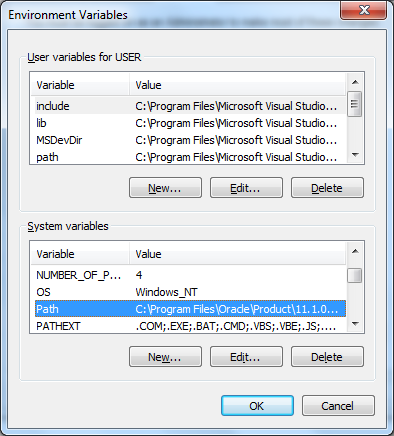
Một bảng mới hiện nên ta lần lượt điền các thông tin như hình vẽ sau:



Hình 9. Hướng dẫn cài đặt PhantomJS

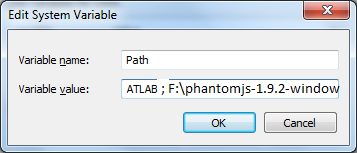
Sau khi điền xong thông tin, click chọn OK.

Trong mục System Variables tìm tới biến Path rồi chọn mục Edit.



Hình 10. Hướng dẫn cài đặt PhantomJS

Trong mục **Variable value**, các bạn di chuyển tới cuối, copy thư mục phantomjs vào:



Hình 11. Hướng dẫn cài đặt PhantomJS

Click chọn OK là đã hoàn thành quá trình cài đặt PhantomJS.

**Bước 3**: Biên dịch chương trình

Tạo một file mã nguồn JavaScript rồi lưu đoạn mã nguồn sau vào:

*Console.log(‘Hello!World’);*

*Phantomjs.exit();*

Hình 12. Mã nguồn chương trình cơ bản với PhantomJS

Giả sử chương trình được lưu với tên file là *hello.js* và có đường dẫn là *F:\JS\CodeJS\hello.js* .

Mở command line trong máy tính rồi gõ lệnh: *phantomjs F:\JS\CodeJS\hello.js*

Chương trình sẽ in ra *Hello!World.*

Đến đây đã hoàn thành quá trình biên dịch một file JavaScript bằng PhantomJS.

## Ví dụ

Sau đây là một vài ví dụ quan trọng được biên dịch bằng PhantomJS để thể hiện rõ các chức năng cũng như đặc tính riêng của ứng dụng PhantomJS.

### Ví dụ về zing.js

Mã nguồn chương trình:

*console.log("Start");*

*var page = require('webpage').create();*

*page.onConsoleMessage = function(msg) {*

*console.log(msg);*

*};*

*page.open("http://login.me.zing.vn/", function(status) {*

*page.render('login\_zing\_s1.png');*

*if ( status === "success" ) {*

*page.evaluate(function() {*

*document.querySelector("input[name='u']").value = "kimbang987";*

*document.querySelector("input[name='p']").value = "bang123";*

*document.querySelector("#frmLogin").submit();*

*console.log("Login submitted!");*

*});*

*page.render('login\_zing\_s2.png');*

*window.setTimeout(function () {*

*page.render('login\_zing\_s3.png');*

*phantom.exit();*

*}, 5000);*

*}*

*});*

Hình 13. Mã nguồn chương trình zing.js

Biên dịch :$ phantomjs $zing.js

$: đường dẫn

Kết quả:

Chương trình tải trang web zing.vn rồi thực hiện đăng nhập vào Website với tài khoản như trên mã nguồn.

Kết quả được hiển thị ra ba bức ảnh: 'login\_zing\_s1.png', 'login\_zing\_s2.png' và 'login\_zing\_s3.png'.

Đây là một cách duyệt Web, bởi nó cũng tải Website trên server về và thao tác như các trình duyệt Web thông thường.

### Ví dụ về weather.js

Mã nguồn có thể tải về trên trang chủ của PhantomJS như sau:

*var page = require('webpage').create(),*

*system = require('system'),*

*city,*

*url;*

*city = 'Mountain View, California'; // mặc định*

*if (system.args.length > 1) {*

*city = Array.prototype.slice.call(system.args, 1).join(' ');*

*}*

*url = encodeURI('http://api.openweathermap.org/data/2.1/find/name?q=' + city);*

*console.log('Checking weather condition for', city, '...');*

*page.open(url, function(status) {*

*var result, data;*

*if (status !== 'success') {*

*console.log('Error: Unable to access network!');*

*} else {*

*result = page.evaluate(function () {*

*return document.body.innerText;*

*});*

*try {*

*data = JSON.parse(result);*

*data = data.list[0];*

*console.log('');*

*console.log('City:', data.name);*

*console.log('Condition:', data.weather.map(function(entry) {*

*return entry.main;*

*}).join(', '));*

*console.log('Temperature:', Math.round(data.main.temp - 273.15), 'C');*

*console.log('Humidity:', Math.round(data.main.humidity), '%');*

*} catch (e) {*

*console.log('Error:', e.toString());*

*}*

*}*

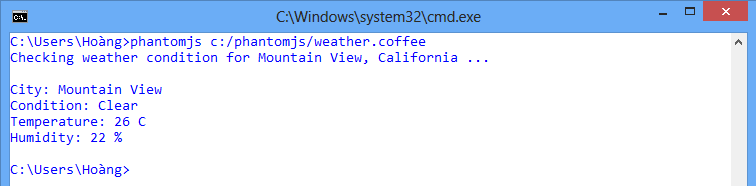
*phantom.exit();*

*});*

Hình 14. Mã nguồn chương trình weather.js

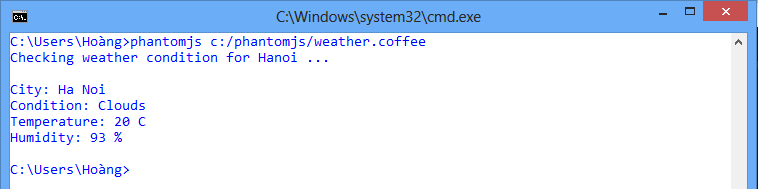
Khi biên dịch chương trình, kết quả sẽ là thông tin thời tiết cập nhật tức thời của một địa điểm nào đó do người dùng thiết lập.

Ví dụ thông tin thời tiết khi thành phố là *“Mountain View, California”:*



Hình 15. Thông tin thời tiết của Mountain View, California

Khi thành phố là “Hà Nội”



Hình 16. Thông tin thời tiết của Hà Nội

Chương trình weather.js cho thấy tính năng truy xuất tức thời vào dữ liệu DOM, CSS của một Website.

### Ví dụ về server.js

Mã nguồn của *server.js* như sau

*var page = require('webpage').create();*

*var server = require('webserver').create();*

*var system = require('system');*

*var host, port; // host và port không được định nghĩa sẵn*

*if (system.args.length !== 2) {*

*console.log('Usage: server.js <some port>');*

*phantom.exit(1);*

*} else {*

*port = system.args[1];*

*var listening = server.listen(port, function (request, response) {*

*console.log("GOT HTTP REQUEST");*

*console.log(JSON.stringify(request, null, 4));*

*// cài đặt headers*

*response.statusCode = 200;*

*response.headers = {"Cache": "no-cache", "Content-Type": "text/html"};*

*// có thể:*

*response.setHeader("foo", "bar");*

*// body*

*// các tiêu đề trên sẽ được gửi đi ngầm định*

*response.write("<html><head><title>YES!</title></head>");*

*// writeBody có thể được gọi đồng thời*

*response.write("<body><p>pretty cool :)</body></html>");*

*response.close();*

*});*

*if (!listening) {*

*console.log("could not create web server listening on port " + port);*

*phantom.exit();*

*}*

*var url = "http://localhost:" + port + "/foo/bar.php?asdf=true";*

*console.log("SENDING REQUEST TO:");*

*console.log(url);*

*page.open(url, function (status) {*

*if (status !== 'success') {*

*console.log('FAIL to load the address');*

*} else {*

*console.log("GOT REPLY FROM SERVER:");*

*console.log(page.content);*

*}*

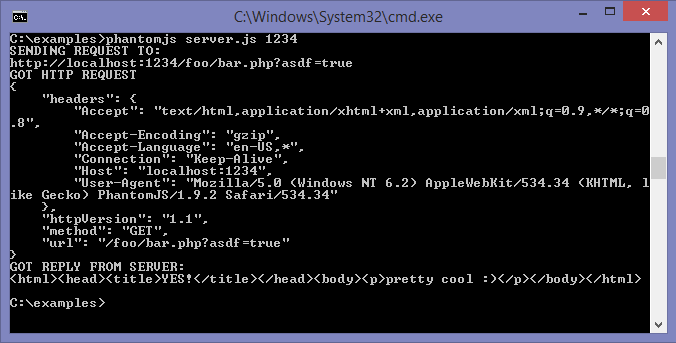
*phantom.exit();*

*});*

*}*

Hình 17. Mã nguồn server.js

Khi biên dịch chương trình, phantomjs sẽ trả về các thành phần cấu thành phần như: “Accept-Encoding”, “Accept-Language”, “Connection”, “Host”, “User-Agent”,… của trang web kể cả html như Hình 18 dưới đây.



Hình 18. Kết quả sau khi chạy server.js với cổng vào là 1234

Kết quả trả về các thông tin cần thiết mà người dùng cần biết về server.

# Ứng dụng thực tế của PhantomJS

## YSlow

### YSlow là gì?

YSlow là một tiện ích được gắn trên trình duyệt Web dùng để phân tích trang Web dựa trên chuẩn các quy tắc của Yahoo. Qua đó cải thiện hiệu suất của Website.

### YSlow và PhantomJS.

YSlow cho PhantomJS là một kịch bản dòng lệnh cho phép phân tích hiệu suất trang Web trực tiếp từ URL.

YSlow cung cấp cho PhantomJS các chuẩn đầu ra mới, sử dụng kiểm tra tự động: TAP (Test Anything Protocol) và Junit. Các chuẩn đầu ra mới vẫn đang được phát triển.

### Ví dụ

Sau đây là ví dụ của YSlow cho PhantomJS.

***Cài đặt:***

* Cài đặt PhantomJS
* Vào trang chủ của YSLow (yslow.org) tải về file *yslow.js*

***Hỗ trợ:***

Gõ lệnh: ($: đường dẫn đến thư mục yslow.js) *$ phantomjs yslow.js - help*

Kết quả sẽ hiển thị ra các cú pháp hỗ trợ người dùng:

*Usage: phantomjs [phantomjs options] yslow.js [yslow options] [url ...]*

*PhantomJS Options:*

*http://y.ahoo.it/phantomjs/options*

*YSlow Options:*

*-h, --help output usage information*

*-V, --version output the version number*

*-i, --info <info> specify the information to display/log (basic|grade|stats|comps|all) [all]*

*-f, --format <format> specify the output results format (json|xml|plain|tap|junit) [json]*

*-r, --ruleset <ruleset> specify the YSlow performance ruleset to be used (ydefault|yslow1|yblog) [ydefault]*

*-b, --beacon <url> specify an URL to log the results*

*-d, --dict include dictionary of results fields*

*v, --verbose output beacon response information*

*-t, --threshold <score> for test formats, the threshold to test scores ([0- 100]|[A-F]|{JSON}) [80]*

*e.g.: -t B or -t 75 or -t '{"overall": "B", "ycdn": "F", "yexpires": 85}'*

*-u, --ua "<user agent>" specify the user agent string sent to server when the page requests resources*

*-vp, --viewport <WxH> specify page viewport size WxY, where W = width and H = height [400x300]*

*-ch, --headers <JSON> specify custom request headers, e.g.: -ch '{"Cookie": "foo=bar"}'*

*-c, --console <level> output page console messages (0: none, 1: message, 2: message + line + source) [0]*

*--cdns "<list>" specify comma separated list of additional CDNs*

*Examples:*

*phantomjs yslow.js http://yslow.org*

*phantomjs yslow.js -i grade -f xml www.yahoo.com www.cnn.com www.nytimes.com*

Hình 19. Các cú pháp hỗ trợ người dùng sử dụng YSlow

***Ví dụ các tiện ích cơ bản:***

* Thông tin cơ bản của JSON của Website *www.yahoo.com*

Câu lệnh:

*$ phantomjs yslow.js --info basic http://www.yahoo.com*

Kết quả:

*{"w":491065,"o":95,"u":"http%3A%2F%2Fwww.yahoo.com","r":60,"i":"ydefault","lt":966}*

* Thông tin cơ bản của trang Web *www.cnn.com*

*$ phantomjs yslow.js --info basic --format plain http://www.cnn.com*

Kết quả:

*size: 561.4K (561458 bytes)*

overall score: D (64)

*url: http://www.cnn.com/*

*# of requests: 137*

*ruleset: ydefault*

*page load time: 2576*

* Xác định CDN

*$ phantomjs yslow.js -i grade --cdns "foo.com,bar.com,baz.com" example.com*

* Kiểm tra TAP cơ bản

*$ phantomjs yslow.js --info basic --format tap --threshold B http://yslow.org*

Kết quả:

*TAP version 13*

*1..1*

*ok 1 B (86) overall score*

* Kiểm tra TAP chi tiết.

*$ phantomjs yslow.js --info grade --format tap --threshold '{"overall": "B", "ycdn"*

*: 65}' http://yslow.org*

Kết quả:

*TAP version 13*

*1..24*

*ok 1 B (88) overall score*

*not ok 2 C (72) ynumreq: Make fewer HTTP requests*

*---*

*message: This page has 7 external Javascript scripts. Try combining them into one.*

*This page has 5 external stylesheets. Try combining them into one.*

*...*

*ok 3 C (70) ycdn: Use a Content Delivery Network (CDN)*

*---*

*message: There are 3 static components that are not on CDN. <p>You can specify CDN hostnames in your preferences. See <a href="https://yslow.org/faq#faq\_cdn">YSlow FAQ</a> for details.</p>*

*offenders:*

*- "yslow.org: 1 component, 8.0K (8.0K GZip)"*

*- "fonts.googleapis.com: 1 component, 1.0K (1.0K GZip)"*

*- "widgets.twimg.com: 1 component, 0.8K"*

*...*

*ok 4 A (100) yemptysrc: Avoid empty src or href*

*not ok 5 F (12) yexpires: Add Expires headers*

*---*

*message: There are 8 static components without a far-future expiration date.*

*offenders:*

*- "http://yslow.org/stylesheets/styles-min.css"*

*- "https://fonts.googleapis.com/css?family=Lato:300italic,700italic,300,700"*

*- "http://widgets.twimg.com/j/2/widget.css"*

*- "http://www.google-analytics.com/ga.js"*

*- "http://widgets.twimg.com/j/2/widget.js"*

*- "http://d.yimg.com/jc/ydn/yslow-ss.png"*

*- "http://widgets.twimg.com/i/widget-logo.png"*

*- "http://yslow.org/favicon.ico"*

*...*

*ok 6 A (100) ycompress: Compress components with gzip*

*ok 7 A (100) ycsstop: Put CSS at top*

## Qunit

Vào trang chủ của PhantomJS tải hai file: run-qunit.js và test.html

Biên dịch file trên như sau:

*$phantomjs run-qunit file://`pwd`/test.html*

Kết quả chương trình:

*$ phantomjs run-qunit.js file://`pwd`/test.html*

*'waitFor()' finished in 200ms.*

*Tests completed in 21 milliseconds.*

*5 tests of 5 passed, 0 failed.*

Giải thích kết quả:

* *'waitFor()' finished in 200ms:* thời gian chạy chương trình là 200ms
* Tiếp theo ý nghĩa là số kiểm thử được hoàn thành

Kết luận: Qunit cho PhantomJS giúp người dùng thực hiện kiểm thử thời gian chạy và tính chính xác của một sự kiện trên Web.

## PhantomJS với Python

Có thể nói PhantomJS là một Webkit không đầu. Với đặc tính này nó có một vài ứng dụng trong việc kiểm thử ứng dụng Web. Ở phần này chúng ta sẽ kết hợp nó với Selenium Webdriver để chạy thử một test case bằng dòng lệnh command line. Qua đó sẽ thấy được tốc độ của PhantomJS khi ứng dụng nó vào việc kiểm thử không đầu.

Selenium Webdriver là một trong những công cụ kiểm thử phần mềm mã nguồn mở mạnh mẽ nhất hiện nay cho việc kiểm thử ứng dụng Web. Selenium script có thể chạy được trên hầu hết các trình duyệt như IE, FireFox, Chrome, Safari, Opera...; và hầu hết các hệ điều hành như Windows , Mac, Linux.

Hình 20 là mã nguồn của một Unittest sử dụng Selenium.

*import unittest*

*from selenium import webdriver*

*import time*

***class TestThree(unittest.TestCase):***

***def setUp(self):***

*self.startTime = time.time()*

***def test\_url\_fire(self):***

*time.sleep(2)*

*self.driver = webdriver.Firefox()*

*self.driver.get("http://www.realpython.com/")*

*self.driver.find\_element\_by\_class\_name("simple-goods-btn").click()*

*self.driver.switch\_to\_frame(1)*

*self.assertIn("https://www.simplegoods.co/embed/XDGLSALS",self.driver.current\_url)*

***def test\_url\_phantom(self):***

*time.sleep(1)*

*self.driver = webdriver.PhantomJS()*

*self.driver.get("http://www.realpython.com/")*

*self.driver.find\_element\_by\_class\_name("simple-goods-btn").click()*

*self.driver.switch\_to\_frame(1)*

*self.assertIn("https://www.simplegoods.co/embed/XDGLSALS", self.driver.current\_url)*

*def tearDown(self):*

*t = time.time() - self.startTime*

*print "%s: %.3f" % (self.id(), t)*

*self.driver.quit*

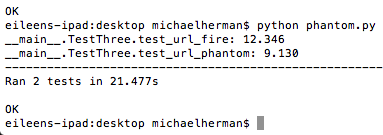
*if \_\_name\_\_ == '\_\_main\_\_':*

*suite = unittest.TestLoader().loadTestsFromTestCase(TestThree)*

*unittest.TextTestRunner(verbosity=0).run(suite)*

Hình 20. Mã nguồn Unittest

Mã nguồn hình 17 là một unit test cơ bản. Unit test này kiểm tra xem nút bấm có tên “*simple-goods-btn”* của trang Web “[*http://www.realpython.com/*](http://www.realpython.com/)*”* khi click vào có đúng là chuyển đến trang “*https://www.simplegoods.co/embed/XDGLSALS”.* Chương trình được viết cho FireFox và PhantomJS để so sánh tốc độ khi sử dụng hai trình duyệt này trong việc viết test. Kết quả chương trình:



Nhìn vào thời gian chạy trên chúng ta có thể thấy việc viết test case cho ứng dụng Web bằng PhantomJS chạy nhanh hơn so với trình duyệt khác. Ngoài ra có nhiều mã nguồn của PhantomJS hỗ trợ cho việc này, cũng như có nhiều dự án đi trước để chúng ta tìm hiểu.

# Tổng kết

Trên đây là bản báo cáo tìm hiểu về PhantomJS.

Nói chung PhantomJS là một công cụ hỗ trợ lập trình viên trong nhiều lĩnh vực của mảng lập trình Web. Nổi bật lên đó là khả năng kiểm thử không đầu, nó giúp người làm Web viết ra các test case cho ứng dụng của mình, nó khá là hữu ích.

Có rất nhiều các Website đã sử dụng PhantomJS làm giải pháp cho vấn đề của mình. Ví dụ như Website Yslow đã được trình bày ở trên...

#### Tài liệu tham khảo

Tài liệu là các trang Web hỗ trợ thông tin về PhantomJS. Các bạn có thể vào các link để tham khảo thêm.

* Trang Web chính thức của PhantomJS:

<http://phantomjs.org>

* Thông tin về nhà phát triển Ariya Hidayat:

<http://ariya.ofilabs.com/about>

* Lịch sử phát triển của PhantomJS (các phiên bản được tung ra):

<http://phantomjs.org/release-names.html>

* Các trang Web thể hiện ứng dụng thực tế:

<http://yslow.org/>

<https://gist.github.com/gmarik/1305062>

-HẾT-