

[Trang Chủ](#) / [Bài Viết](#) / RESTful Web Services Là Gì?

## RESTful Web Services Là Gì?

[Hoàng Tùng](#) 19 March 2016

**RESTful Web Services** là một thuật ngữ được sử dụng rất nhiều trong những năm gần đây. Tuy nhiên không ít người cảm thấy khó hiểu về khái niệm này khi mà thậm chí có bạn nói đã Google hàng giờ mà vẫn ...không hiểu gì hết. Chính vì thế RESTful Web Services luôn trở thành nỗi ám ảnh cho các bạn lập trình viên trẻ khi phải đối mặt với các nhà tuyển dụng đặc biệt là khi nó được theo sau bởi một cụm từ cũng khó hiểu chẳng kém là **Web Services** phía sau nó.

Tin đặc biệt tốt là thực ra RESTful Web Services không khó hiểu và đáng sợ như khi bạn bị mấy tay tuyển dụng đặt câu hỏi trong buổi phỏng vấn đề 'dìm hàng', 'dìm luôn lương' bạn. Ngay bây giờ mình sẽ giải thích một cách ngắn gọn đi vào thực tiễn bằng ví dụ luôn chứ không lý thuyết lơ tơ mơ như mấy trang khác để giúp các bạn hiểu RESTful là cái quái gì mà mấy anh trên mạng giải thích lằng nhằng khó hiểu thế.

Thế nên mình xin phép bắt đầu bằng 1 ví dụ và ném cái định nghĩa dài loằng ngoằng ở cuối bài nhe :-)

Giả sử bạn cần tạo một trang blog cá nhân về [học lập trình](#). Trên trang này có những bài viết (tiếng Anh nó gọi là *article* đấy). Các bài viết sẽ được *tạo ra* bởi một tác giả nào đó, sau đó tác giả này có thể *chỉnh sửa* bài viết, tiếp đó sau khi chỉnh sửa chán chê thì tác giả có thể quyết định *cập nhật* bài viết. Và cuối cùng nếu tác giả thấy bài mình viết ra chuối quá éo có ai thèm đọc cả thì tác giả có thể chọn cách *xoá* bài viết cho khỏi phải nhìn thấy đỡ cảm thấy nhục : ))

4 hành động trên: *tạo ra* bài viết (*create*), *chỉnh sửa* bài viết (*edit*), *cập nhật* bài viết (*update*) hay *xoá* bài viết (*delete*) tương ứng với 4 cách thức mà một tài nguyên được quản lý (ở đây tài nguyên chính là bài viết, tuy nhiên *tài nguyên* hay *resource* không chỉ giới hạn ở bài viết mà nó còn có thể là tài khoản người dùng, cũng được quản lý thông qua 4 hành động *tạo ra* tài khoản hay đăng ký, *chỉnh sửa* thông tin tài khoản, *cập nhật* thông tin tài khoản và *xoá* tài khoản).

Cảm thấy 4 hành động trên sẽ là một công việc quá phổ biến đối với hầu hết các trang web (bạn cứ tưởng tượng xem ngay cả 1 trang thuộc loại hàng khủng như Facebook thì người dùng cũng chủ yếu là tạo bài viết, chỉnh sửa bài viết, cập nhật bài viết và xoá bài viết, tương tự cho thông tin tài khoản), nên đã có một vị Giáo sư đại tài (mà mình quên mịa mất tên bác ấy rồi) chế hẳn ra một quy luật thống nhất như sau để cho tất cả các lập trình viên cứ thế mà làm theo mỗi lần phải lập trình một trang có 4 chức năng trên. Bác này phát biểu quy luật như sau:

- Khi tạo ra một *resource* thì các chú cố gắng sử dụng phương thức **POST** để gửi dữ liệu lên máy chủ nhé (cái này nếu học HTML thì bạn sẽ nhớ ra là phải để thuộc tính **method="POST"** khi tạo một biểu mẫu. Ví dụ như sau: `<form method="POST" action="url-la-cai-me-gi-cung-duoc-het-a.php"></form>`).
- Khi tạo cập nhật một *resource* thì các chú cứ cố gắng sử dụng phương thức **PUT** nhé (chú nào mà không làm thì coi chừng thẳng đồng nghiệp nó chửi cả thầy lẫn trò khi phải viết lại code của chú đấy!)
- Khi xoá một *resource* thì các chú cứ cố gắng sử dụng phương thức **DELETE** nhé.
- Và cuối cùng là khi hiển thị một bài viết cho độc giả họ đọc thì sử dụng phương thức **GET**.

Tóm lại, RESTful service nghe màu mè nhưng rốt cuộc chỉ có thế thôi. Nó là một **kiến trúc** thống nhất giúp thiết kế các website để có thể dễ dàng quản lý các tài nguyên. Nó không phải là một quy luật buộc bạn phải tuân theo mà đơn giản là một *kiến trúc* được đề xuất ra và kiến trúc này hiện đang được sử dụng rất phổ biến vì tính đơn giản, dễ hiểu và rất ưu việt của nó. Nhân tiên nhớ ra là REST được đề xuất bởi nhà khoa học máy tính Roy Thomas Fielding vào năm 2012.

Ơ chứ không phải chú mày mới viết ở trên RESTful là 1 *quy luật thống nhất* à???

Hic, khúc này thành thật xin lỗi các bạn vì đã ...cố tình viết sai sự thật một tí. RESTful nói đúng hơn là một kiến trúc chứ không phải quy luật. Nhưng mà... nếu mình không viết vậy lại thay bằng chữ kiến trúc chắc hẳn se khiến các bạn khó hiểu vì cái cụm từ hàn lâm này nghe rất trừu tượng (thế nên thôi để chữ quy luật cho các bạn dễ hình dung). Lạc đề tí nhưng mà mình có quen 3 cô tên Trúc cô nào cũng xinh đẹp hết á : ))).

Nói tóm lại RESTful là cái mẹ gì? Vâng thì ngay bây giờ mình xin được đưa ra định nghĩa về REST như đã hứa ở phần đầu bài viết.

RESTful Là Gì

REST là viết tắt của cụm từ Representational State Transfer (đôi khi còn được viết là ReST) là một kiểu kiến trúc được sử dụng trong việc giao tiếp giữa các máy tính (máy tính cá nhân và máy chủ của trang web) trong việc quản lý các tài nguyên trên internet. REST được sử dụng rất nhiều trong việc phát triển các ứng dụng Web Services sử dụng giao thức HTTP trong giao tiếp thông qua mạng internet. Các ứng dụng sử dụng kiến trúc REST này thì sẽ được gọi là ứng dụng phát triển theo kiểu RESTful.

*Bonus:* Tặng luôn phần chú thích về ngữ pháp tiếng Anh

Trong từ *RESTful*, thì từ *ful* (đọc là phù, phồ, phò hay phùn đều được) chính là *suffix* trong tiếng Anh, giống như từ *help* có nghĩa là giúp đỡ thì từ *helpful* là rất hữu ích.

Kiến Trúc    Architechture    REST    RESTful    API

[Chỉnh sửa](#)

4 Phản Hồi

[Cảnh Dương](#) 28 November 2017

Cảm ơn anh nhiều <3

[Thêm bình luận](#)

15

Vote

[Dai minh](#) 09 January 2018

cảm ơn anh nhé, anh viết dễ hiểu với cả vui tính lắm :p

[Thêm bình luận](#)

32

Vote

[Đỗ Nhật](#) 28 February 2018

Bác viết rất hay, Cảm ơn bác. Mình không thấy Like Button ở đâu cả.

[Thêm bình luận](#)

25

Vote

[Nguyễn Dương](#) 06 March 2018

cảm ơn bạn nhiều :))

[Thêm bình luận](#)

1

Vote

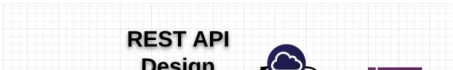
Thêm Phản Hồi

Đăng nhập để thêm nội dung...

Gửi

Bài Viết Liên Quan

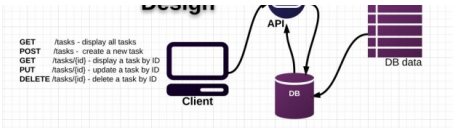
**[RESTful API Cho Người Bắt Đầu](#)**



13 March 2017

Restful API trong 5 phút

14 November 2018



```
Restful API


POST    /articles
PUT     /articles/123
GET     /articles/123
DELETE  /articles/123


CodeHub.vn
```

OpenID và OAuth Khác Nhau Như Thế Nào

08 March 2017

Please sign in.

 Sign in with Google

 Sign in with Facebook

or

Email

Next

Điểm Danh Những HTTP Status Code Thông Dụng Developer Cần Phải Thuộc Là Màng

16 June 2017

| Code | Message                         | Meaning  | Example  |
|------|---------------------------------|--|--|
| 200  | OK                              | The request has succeeded.   | GET / HTTP/1.1 200 OK                              |
| 201  | Created                         | The request has been fulfilled and resulted in a new resource being created.                       | POST / HTTP/1.1 201 Created                        |
| 202  | Accepted                        | The request has been accepted for processing, but the processing has not been completed.           | POST / HTTP/1.1 202 Accepted                       |
| 204  | No Content                      | The request has been fulfilled and there is no content to send back.                               | DELETE / HTTP/1.1 204 No Content                   |
| 301  | Moved Permanently               | The resource has been moved permanently to the location returned by the Location: response header. | GET / HTTP/1.1 301 Moved Permanently               |
| 302  | Found                           | The resource has been moved temporarily to the location returned by the Location: response header. | GET / HTTP/1.1 302 Found                           |
| 303  | See other                       | The resource has been moved to the location returned by the Location: response header.             | GET / HTTP/1.1 303 See other                       |
| 304  | Not Modified                    | The resource has not been modified since the last time it was requested.                           | GET / HTTP/1.1 304 Not Modified                    |
| 400  | Bad Request                     | The request could not be understood by the server.   | GET / HTTP/1.1 400 Bad Request                     |
| 401  | Unauthorized                    | The request requires user authentication.  | GET / HTTP/1.1 401 Unauthorized                    |
| 403  | Forbidden                       | The server refuses to fulfill the request because it is not authorized to do so.                   | GET / HTTP/1.1 403 Forbidden                       |
| 404  | Not Found                       | The resource requested by the client does not exist.   | GET / HTTP/1.1 404 Not Found                       |
| 405  | Method Not Allowed              | The method is not supported by the resource.   | DELETE / HTTP/1.1 405 Method Not Allowed           |
| 406  | Not Acceptable                  | The resource is not available in the format requested by the client.                               | GET / HTTP/1.1 406 Not Acceptable                  |
| 407  | Proxy Authentication Required   | The client must first authenticate itself with the proxy.  | GET / HTTP/1.1 407 Proxy Authentication Required   |
| 408  | Request Timeout                 | The server timed out waiting for the request.  | GET / HTTP/1.1 408 Request Timeout                 |
| 409  | Conflict                        | The request conflicts with the current state of the resource.                                      | PUT / HTTP/1.1 409 Conflict                        |
| 410  | Gone                            | The resource no longer exists and will not be available again.                                     | GET / HTTP/1.1 410 Gone                            |
| 411  | Length Required                 | The request does not have a valid Content-Length header.   | POST / HTTP/1.1 411 Length Required                |
| 412  | Precondition Failed             | The precondition of the request is not satisfied.  | PUT / HTTP/1.1 412 Precondition Failed             |
| 413  | Request Entity Too Large        | The request entity is too large to be processed.   | POST / HTTP/1.1 413 Request Entity Too Large       |
| 414  | Request-URI Too Long            | The request-URI is too long.   | GET / HTTP/1.1 414 Request-URI Too Long            |
| 415  | Unsupported Media Type          | The request entity is of a type not supported by the resource.                                     | POST / HTTP/1.1 415 Unsupported Media Type         |
| 416  | Requested Range Not Satisfiable | The range requested by the client is not satisfiable.  | GET / HTTP/1.1 416 Requested Range Not Satisfiable |
| 417  | Expectation Failed              | The expectation of the request is not satisfied.   | POST / HTTP/1.1 417 Expectation Failed             |
| 500  | Internal Server Error           | The server encountered an internal error.  | GET / HTTP/1.1 500 Internal Server Error           |
| 501  | Not Implemented                 | The server does not support the requested functionality.   | POST / HTTP/1.1 501 Not Implemented                |
| 502  | Bad Gateway                     | The server received an invalid response from the upstream server.                                  | GET / HTTP/1.1 502 Bad Gateway                     |
| 503  | Service Unavailable             | The server is currently unavailable.   | GET / HTTP/1.1 503 Service Unavailable             |
| 504  | Gateway Timeout                 | The server timed out waiting for the upstream server.  | GET / HTTP/1.1 504 Gateway Timeout                 |
| 505  | HTTP Version Not Supported      | The server does not support the requested HTTP version.  | GET / HTTP/1.1 505 HTTP Version Not Supported      |

Hướng Dẫn Cách Nhúng Google Calendar vào Trang Web

17 January 2018

Cài đặt

Thêm lịch

Nhập và xuất

Cài đặt lịch của lịch của tôi

**Viết bài trên CodeHub...**

Cài đặt lịch

Quản lý trang nhập

Chia sẻ với người cụ thể

Thông báo sự kiện

Thông báo sự kiện cá nhân

Thông báo chung

Tích hợp lịch

Xóa lịch

Sinh nhật

Cài đặt lịch

Tên:

Mô tả:

Màu:  [Mặc định](#) [Mặc định](#)

Chia sẻ với:  [Chỉ mình tôi](#) [Chỉ mình tôi](#)

Họ tên:  [Họ tên của bạn](#) [Họ tên của bạn](#)

Quản lý trang nhập

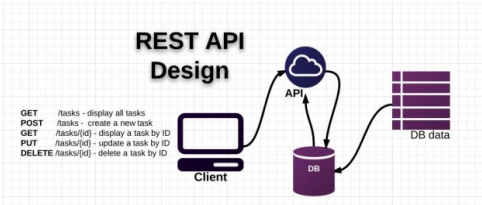
☐ Hiện thị cho mọi người [Hiện thị cho mọi người](#)

Họ tên:  [Họ tên của bạn](#) [Họ tên của bạn](#)

Bài Viết Nổi Bật

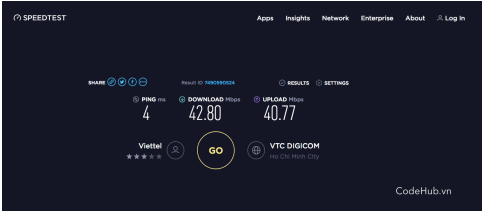
RESTful API Cho Người Bắt Đầu

13 March 2017



Kiểm tra tốc độ đường truyền internet với Speedtest

22 July 2018



JSON Là Gì và Sử Dụng JSON Như Thế Nào

12 March 2017

Cơ Sở Dữ Liệu và Hệ Quản Trị Cơ Sở Dữ Liệu

19 September 2015



hoclaptrinh.org

[Giới Thiệu](#) [Blog](#) [Việc Làm](#) [Chủ Đề](#) [Người Dùng](#) [Liên Hệ](#)

Thiết kế và phát triển bởi [CodeHub](#) © 2019