Báo cáo Prog4: Lập trình mạng/HTTP client

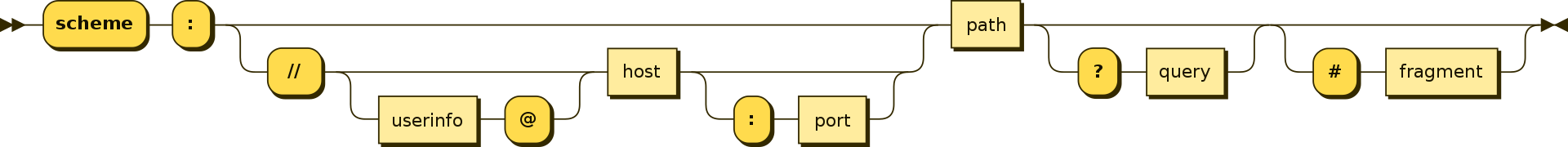
# Tìm hiểu giao thức HTTP

## Các methods phổ biến

Các methods HTTP phổ biến: GET, POST, HEAD, PUT, DELETE, CONNECT, OPTIONS, TRACE, PATCH.

* GET
  + Chỉ được dùng để truy vấn dữ liệu từ web server thông qua URL, có thể kèm thêm URL parameters
  + GET method có thể được cached, bookmarked và được lưu vào lịch sử trình duyệt
  + GET có giới hạn length của header
  + Không nên dùng GET để truy vấn dữ liệu nhạy cảm
* POST
  + Dùng để gửi data đến server để tạo/cập nhật một tài nguyên nào đó
  + Trái với GET, POST không giới hạn length của header, không được cached, bookmarked, và không được lưu vào lịch sử trình duyệt
* HEAD
  + Giống với GET, nhưng chỉ trả về header từ server
  + Hữu ích khi muốn kiểm tra kết nối với server, kiểm tra những thông tin có thể nhận được từ server trước khi tạo GET request
  + HEAD có thể được cached giống GET
* PUT
  + PUT gửi data đến server để tạo một tài nguyên nào đó
  + Ngoài ra, PUT sẽ coi như tài nguyên đó trên server không tồn tại hoặc có thể được ghi đè, nên khi được gọi nhiều lần sẽ trả về cùng 1 kết quả (trong khi POST có thể có những kết quả khác nếu gọi nhiều lần)
* DELETE
  + Dùng để xóa tài nguyên nào đó
* CONNECT
  + Tạo kết nối tunnel đến tài nguyên nào đó thay vì tương tác với tài nguyên như những methods trên
* OPTIONS
  + Xác định những methods có thể sử dụng với tài nguyên nào đó trên server
  + OPTIONS request với request-target là \* sẽ áp dụng với toàn bộ server nói chung
* TRACE
  + Thực hiện loopback test, truy vết path đến tài nguyên được truy cập
  + Hữu ích trong việc debug
* PATCH
  + PATCH gửi data đến server để cập nhật, chỉnh sửa một tài nguyên có sẵn trên server

## Cấu trúc URL



* Scheme:
  + Không được để trống
  + Kết thúc bằng dấu “:”
  + Những scheme phổ biến: http, https, ftp, mailto, file, data và irc
* Authority:
  + Không bắt buộc phải có
  + Bắt đầu bằng dấu “//”
  + Thành phần phụ userinfo có format username:password, kết thúc bởi dấu “@” (vcth4nh:vcth4nh@)
  + Host: có thể là tên miền (www.google.com), IPv4 (172.217.194.147) hoặc IPv6 ([::ffff:acd9:c293]) (IPv6 cần có dấu hoặc vuông bao ngoài)
  + Thành phần phụ port, bắt đầu bằng dấu “:” (:80)
* Path:
  + Bao gồm những phân đoạn được tách với nhau bằng dấu “/” (/wiki/URL)
* Query:
  + Không bắt buộc phải có
  + Bắt đầu bằng dấu “?”
  + Bao gồm những URL parameter, phân cách nhau bằng dấu “&” hoặc “;” (?key1=value1&key2=value2)
* Fragment:
  + Không bắt buộc phải có
  + Bắt đầu bằng dấu “#”
  + Khiến cho trình duyệt đến chỗ có id attribute tương ứng (#heading1)

## Status codes phổ biến

* 1xxs – Informational responses

Những status code 1xxs không được dùng phổ biến

* 2xxs – Success: request đã được xử lí thành công và được gửi về trình duyệt kết quả như mong đợi
  + 200 OK: request thành công. Định nghĩa thành công khác nhau với từng method:
    - GET: Tài nguyên được lấy và gửi thông qua message body
    - HEAD: phản hồi từ server có representation headers nhưng không có message body
    - PUT/POST: kết quả của hành động PUT/POST được phản hồi lại trong message body
    - TRACE: message body chứa request mà server nhận được
  + 201 Created:
    - Tài nguyên mới được tạo theo request từ client
    - Status code này thường được phản hồi sau PUT/POST
* 3xxs – Redirection: được điều hướng sang chỗ khác
  + 301 Moved Permanently
    - Cần chuyển hướng đến nơi khác để truy cập URL được yêu cầu. URL mới sẽ có trong phải hồi từ server
  + 302 Found
    - URI của tài nguyên được request được thay đổi tạm thời
  + 307 Temporary Redirect
    - Client sẽ được chuyển hướng đến nơi khác và sử dụng cùng request method để truy cập URI được yêu cầu
    - Việc chuyển hướng này là tạm thời
  + 308 Permanent Redirect
    - Client sẽ được chuyển hướng đến nơi khác và sử dụng cùng request method để truy cập URI được yêu cầu
    - Việc chuyển hướng này là dài hạn
* 4xxs – Client errors: tài nguyên không tìm thấy/không thể truy cập
  + 400 Bad Request
    - Server không hiểu request
    - Ví dụ như do lỗi cú pháp trong request
  + 403 Forbidden
    - Client không có quyền truy cập tài nguyên nào đó
  + 404 Not Found
    - Server phản hồi lại status code này nếu tài nguyên được request không tồn tại
    - Ngoài ra, server cũng có thể phản hồi status code này thay vì 403 Forbidden để giấu sự tồn tại của tài nguyên với client không có quyền truy cập.
  + 405 Method Not Allowed
    - Server không chấp nhận request method đến tài nguyên nào đó
    - Ví dụ: server chỉ hỗ trợ GET và POST nhưng client lại gửi Delete
  + 410 Gone
    - Tài nguyên được truy cập đã bị xóa khỏi server và không có fowarding address
    - Báo rằng client nên xóa cache và links đến tài nguyên đó
* 5xxs – Server errors: request đúng được gửi đi nhưng server không thể xử lý request đó.
  + 500 Internal Error
    - Server gặp sự cố khi phản hồi request
  + 501 Not Implemented
    - Server không có khả năng xử lý request method nhận được
  + 504 Gateway Timeout
    - Server được kết nối đang đóng vai trò một gateway hoặc proxy và không nhận được phải hồi từ upstream server
  + 505 HTTP Version Not Supported
    - Phiên bản HTTP trong request không được hỗ trợ bởi Server

## Request headers

* Host: chỉ định tên của domain, có thể thêm TCP port
* User-Agent: giúp server xác định loại ứng dụng truy cập đến server, hệ điều hành, nhà phân phối phần mềm và phiên bản phần mềm của client.
* Content-Type: Media type được gửi trong message body
* Content-Length: kích thước của message body, đơn vị bytes
* Cookie: chứa những HTTP cookies cần được gửi đến server
* Location: chứa URL mà client sẽ được redirect đến, chỉ có hiệu lực khi gửi cùng status code 3xxs hoặc 201

## Các kiểu encode data trong POST request

Có 3 kiểu encode trong POST request:

* application/x-www-form-urlencoded
  + Những cặp key và value sẽ được encode theo dạng ‘key’=’value’, các cặp key-value được ngăn cách nhau bằng dấu “&”
  + Những ký tự không phải chữ hoặc số được chuyển về dạng Percent encoded
  + Ví dụ: home=Cosby&favoriteflavor=fire%20flies
* multipart/form-data
  + Mỗi value được gửi như một data block, mỗi data block phân cách với nhau bằng boundary (được định nghĩa trong request header bởi user-agent)
  + Mỗi data block có header Content-Disposition chứa key của value tương ứng
  + Dạng encode này thường được dùng để gửi file
  + Ví dụ:

------WebKitFormBoundarykuakICCVkiKpxkBt

Content-Disposition: form-data; name="action"

get-attachment

------WebKitFormBoundarykuakICCVkiKpxkBt

Content-Disposition: form-data; name="id"

182

------WebKitFormBoundarykuakICCVkiKpxkBt—

* text/plain
  + Gửi plain text đến server, không có encode

# Tìm hiểu công cụ Burpsuit để intercept HTTP request

* Graphical user interface, application

  Description automatically generatedMở Burp Suit, chọn Next -> Start Burp để bắt đầu phiên làm việc
* Graphical user interface, application

  Description automatically generatedVào tab Proxy, chọn Open browser

Graphical user interface, application

Description automatically generated

* Graphical user interface, text, application

  Description automatically generatedLúc này trình duyệt Chromium sẽ hiện lên. Nếu toggle “Intercept is on” thì mỗi lần request được gửi trên trình duyệt đó, Burp sẽ chặn lại để mình có thể chỉnh sửa message gửi server. Nếu toogle “Intercept is off”, Burp sẽ không chặn request và người dùng dùng browser như bình thường.
* Có thể ấn Ctrl + R để gửi request vào tab Repeat. Trong tab Repeat chúng ta có thể gửi request nhiều lần, hữu ích khi muốn thử nhiều payload