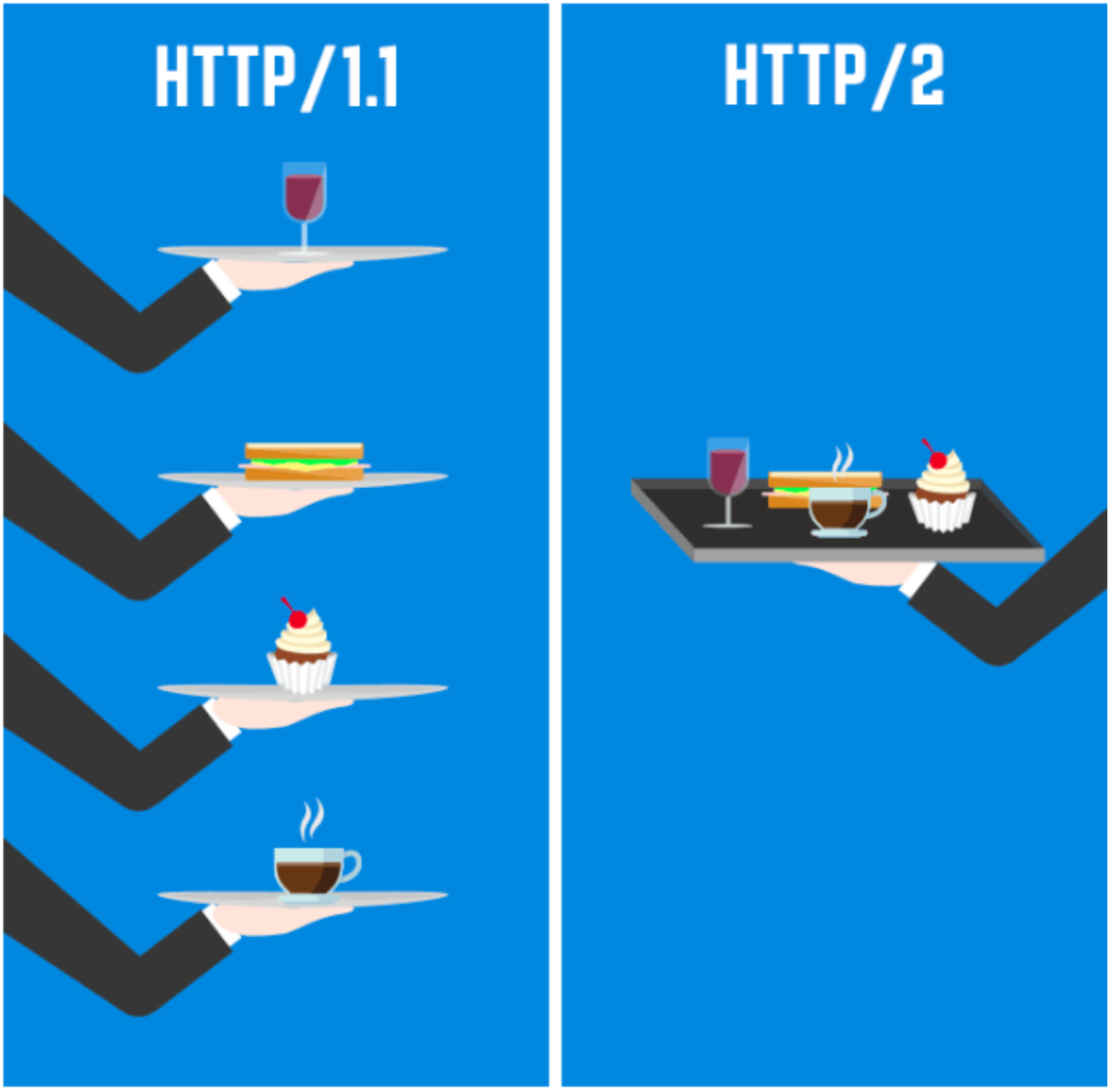
LẬP TRÌNH MẠNG/HTTP client

I. Lý thuyết

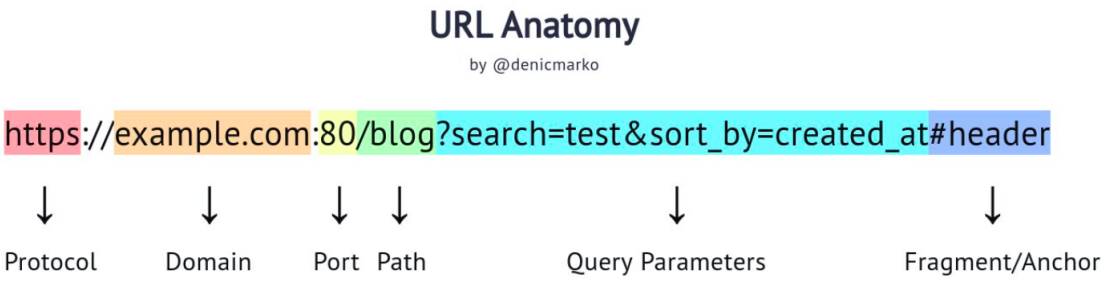
1. Request line của mội HTTP request
    - Request line là dòng đầu tiên trong HTTP request. Nó bao gồm 3 phần:
        - Phương thức HTTP được sử dụng
        - URI(Uniform Resource Identifier) giúp xác định các tài nguyên mà client yêu cầu.
        - Phiên bản của giao thức HTTP
    - Một request line sẽ có định dạng như sau:
        - `POST /admin/login HTTP/2`



- Cải tiến của HTTP 2 Binary protocol
- Tiêu thụ ít băng thông hơn, được phân tích cú pháp hiệu quả hơn và ít bị lỗi hơn các text protocol được sử dụng bởi HTTP/1.1. Ngoài ra, chúng có thể xử lý tốt hơn các yếu tố như khoảng trắng, viết hoa và kết thúc dòng.
- Multiplexing - HTTP/2 được ghép kênh, tức là nó có thể khởi tạo nhiều yêu cầu song song trên một kết nối TCP duy nhất. Do đó, các trang web chứa một số phần tử được phân phối qua một kết nối TCP.
- HTTP/2 sử dụng nén tiêu đề để giảm chi phí do cơ chế khởi động chậm của TCP gây ra .
- Server push - HTTP/2 đẩy các tài nguyên có khả năng được sử dụng vào bộ nhớ cache của trình duyệt, ngay cả trước khi chúng được yêu cầu. Điều này cho phép các trình duyệt hiển thị nội dung mà không cần các chu kỳ yêu cầu bổ sung. Tăng cường bảo mật
- - Các trình duyệt web chỉ hỗ trợ HTTP/2 thông qua các kết nối được mã hóa, tăng tính bảo mật của người dùng và ứng dụng.

- Các phương thức HTTP request bao gồm:
  - GET: Là phương thức được Client gửi dữ liệu lên Server thông qua đường dẫn URL nằm trên thanh địa chỉ của Browser. Server sẽ nhận đường dẫn đó và phân tích trả về kết quả cho Client.
    - Một số đặc điểm chính của phương thức GET là:
      - Giới hạn độ dài của các giá trị là 255 kí tự
      - Chỉ hỗ trợ kiểu dữ liệu string
      - Có thể lưu vào bộ nhớ cache
  - POST: Là phương thức gửi dữ liệu đến server giúp bạn có thể thêm mới dữ liệu hoặc cập nhật dữ liệu đã có vào database
    - Một số đặc điểm chính của POST:
      - Dữ liệu cần thêm hoặc cập nhật không được hiển thị trong URL của trình duyệt
      - Dữ liệu không được lưu trong lịch sử trình duyệt
      - Không có hạn chế về độ dài của dữ liệu

        - Hỗ trợ nhiều kiểu dữ liệu như: String, binary, integers,...
    - PUT: Hoạt động tương tự như POST nhưng nó sẽ ghi đè tất cả thông tin của đối tượng được gửi lên. Khi gửi một request POST với cùng một nội dung hai lần thì sẽ có thông báo lỗi, nhưng PUT thì không, nó luôn trả về kết quả như nhau
    - PATCH: Ghi đè các thông tin được thay đổi của đối tượng và không thay đổi toàn bộ thông tin của đối tượng
    - DELETE: Xóa tài nguyên trên server.
    - CONNECT: Thiết lập một kết nối tới server theo URL.
    - OPTIONS: Response trả về các phương thức HTTP request mà server cho phép
    - TRACE: Thực hiện một bài test loop - back theo đường dẫn đến resource. TRACE thường được sử dụng cho mục đích debug
- Định dang của URL:
    - Protocol: URL (Uniform Resource Locator) tạm dịch là "định vị tài nguyên thống nhất" Cụ thể, URL là địa chỉ của một tài nguyên duy nhất trên Web. Mỗi URL hợp lệ sẽ trỏ đến một tài nguyên duy nhất, tài nguyên đó có thể là trang HTML, tài liệu CSS, hình ảnh, video, file PDF,...
    - Url có định dạng



    - Protocol: Giao thức được sử dụng để truy cập tài nguyên. Có các giao thức như: HTTP, HTTPS, FTP,...
    - Domain: Bao gồm Subdomain, Domain và Top-level Domain.
    - Port:
    - Path: Truy vấn tới các tài nguyên của trang web

    - Query Parameters: Các cặp key, value, bắt đầu sau dấu ?, cung cấp tài nguyên cho path, các tài nguyên được phân cách bởi dấu &. Ví dụ: "?key1=value1&key2=value2
    - Fragment/Anchor: Dùng để truy cập nhanh tới tài nguyên của trang web
- Các HTTP header
    - HOST: Thông tin về server và cổng port mà client gửi request tới. Nêu không có cổng port thì sẽ mặc định sẽ là 80 đối với HTTP và 443 đối với HTTPS
    - Location: Chỉ ra cho URL chuyển hướng tới trang nào.
    - Referer: Cho phép trang web nhận diện được người dùng đến từ trang nào
    - Content-Length: Kích thước của body gửi đến server tính theo byte.
    - Cache-control: Header có hướng dẫn (chỉ thị) ở cả request và response về cách xử lý caching ở trình duyệt và chia sẻ cache

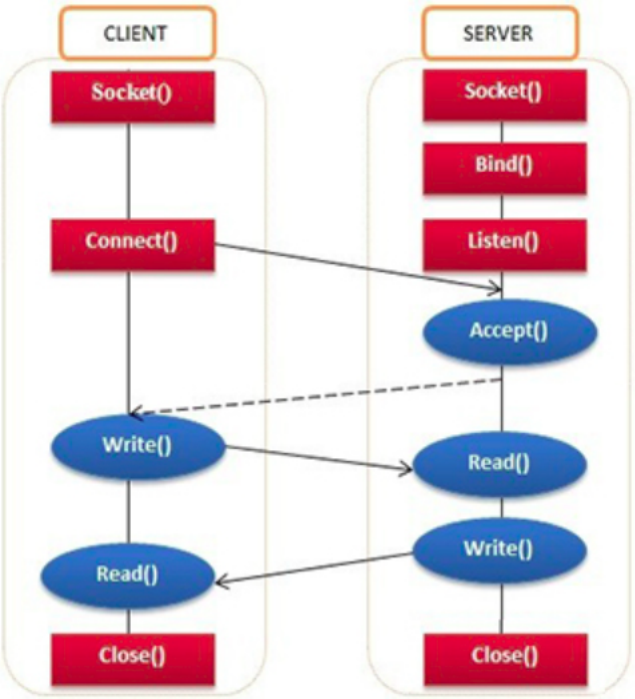
Request	Response
`max-age`	`max-age`
`max-stale`	-
`min-fresh`	-
-	`s-maxage`
`no-cache`	`no-cache`
`no-store`	`no-store`
`no-transform`	`no-transform`
`only-if-cached`	-
-	`must-revalidate`
-	`proxy-revalidate`
-	`must-understand`
-	`private`
-	`public`
-	`immutable`
-	`stale-while-revalidate`
`stale-if-error`	`stale-if-error`

- Bảng liệt kê các chỉ thị Cache-Control tiêu chuẩn:
- Connection: Header này được sử dụng để kiểm tra xem kết nối mạng có mở sau khi gói tin được hay không. Nếu giá trị được gửi là *Keep-alive,* kết nối vẫn ổn định và không bị đóng cho phép gửi các request tiếp theo đến cùng máy chủ.
- Range: Header này cho biết phần tài liệu của máy chủ trả về. Một số phần có thể yêu cầu Range một lúc và server sẽ trả về những yêu cầu này trong một tài liệu có nhiều phần.
- Strict-Transport-Security: Header này dùng để báo rằng trang web chỉ có thể truy cập bằng HTTPS, nếu có truy cập sử dụng HTTP thì nó sẽ được tự động đổi thành HTTPS
- Access-Control-Allow-*: Đều là các response header
    - Access-Control-Allow-Credentials Mặc định, cookie sẽ không được sử dụng trong các truy vấn CORS. Header này sẽ biểu thị giá trị logic rằng có thể sử dụng cookie hay không. Giá trị duy nhất của header này là true. Nếu không muốn sử dụng cookie thì thông thường người ta sẽ bỏ header này trong response chứ không phải đặt giá trị nó là false. -
    - Access-Control-Allow-Headers Trả về 1 hoặc danh sách các header được server chấp thuận, nếu request gửi header server không chấp thuận thì sẽ bị bỏ quả
    - Access-Control-Allow-Methods: Trả về một hoặc danh sách các method được server chấp thuận và cho phép client sử dụng
    - Access-Control-Allow-Origin: Các request gửi từ giá trị của header này sẽ được server chấp thuận
    - Access-Control-Expose-Headers: Bổ sung thêm cho trình duyệt header có thể sử dụng
    - Access-Control-Max-Age: Thời gian phản hồi của preflight request được lưu trong bộ nhớ cache
    - Access-Control-Request-Headers: Đây là danh sách (ngăn cách bằng dấu phẩy) các header được thêm vào truy vấn.
    - Access-Control-Request-Method: Method mà truy vấn sử dụng
- Các status code của HTTP response:

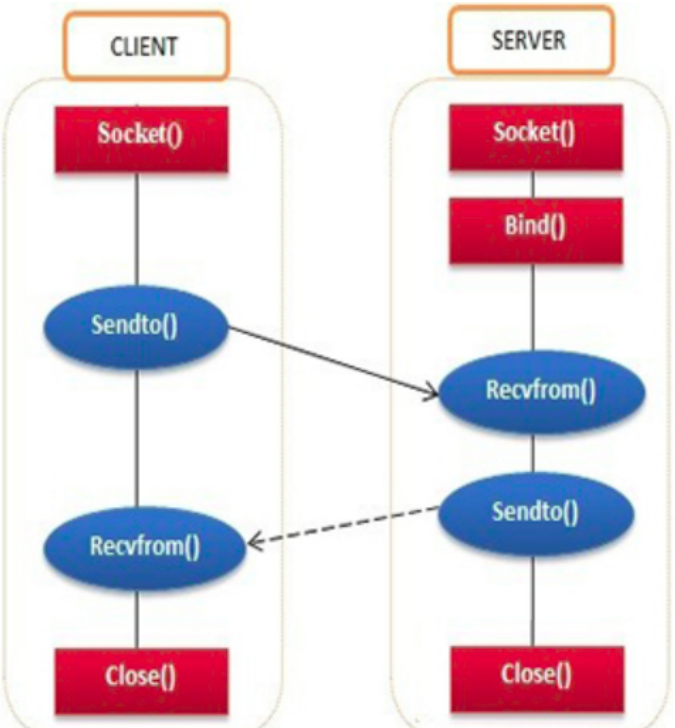
- Tất cả các HTTP status code phản hồi được chia ra thành 5 hạng mục riêng biệt và là các số nguyên có 3 chữ số. Chữ số đầu được dùng để xác định loại phản hồi, trong khi 2 chữ số cuối thì không có bất kỳ vai trò phân loại nào. HTTP status code sẽ cho ta biết liệu 1 yêu cầu HTTP cụ thể đã được hoàn thành thành công hay chưa.
- 1xx: request đã được server tiếp nhận và quá trình xử lý request đang được tiếp tục.
- 2xx: request đã được server tiếp nhận, hiểu và xử lý thành công
- 3xx: Redirection messages
- 4xx: Lỗi ở phía client
- 5xx: Lỗi ở phía server

II. Thực hành

1. Socket
    - Socket là điểm cuối của một liên kết hai chiều giữa hai chương trình chạy trên internet. Socket cho phép 2 chương trình có thể giao tiếp với nhau
    - **Cách thức hoạt động:**
        - Thông qua TCP/IP và UDP, socket sẽ tiến hành truyền và nhận dữ liệu từ internet. Từ đó tạo nên kênh kết nối giữa client và server. Điều kiện để hoạt động này diễn ra là có đủ thông tin về thông số IP và dữ liệu về cổng của 2 chương trình muốn kết nối với nhau.
    - **Các loại socket**
        - **Stream socket** (Socket TCP): Hoạt động dựa trên giao thức TCP. Stream socket chỉ hoạt động khi server và client đã được kết nối với nhau
        - Ưu điểm:
            - Dữ liệu truyền đi được đảm bảo truyền đến đúng nơi nhận, đúng thứ tự với thời gian nhanh chóng
            - Mỗi thông điệp gửi đi đều có xác nhận trả về để thông báo cho người dùng thông tin về quá trình truyền tải.
        - Nhược điểm:
            - Khi kết nối một máy phải chờ máy còn lại chấp nhận kết nối



- **Datagram socket:** Hoạt động dựa trên giao thức UDP. Datagram có thể hoạt động kể cả khi không có sự thiết lập kết nối giữa 2 máy với nhau.
    - Ưu điểm:
        - Quá trình kết nối và truyền tải thông tin đơn giản
        - Thời gian truyền tải dữ liệu cực nhanh
    - Nhược điểm;
        - Quá trình truyền thông tin không đảm bảo được độ tin cậy



- **Web Socket:** công cụ hỗ trợ việc kết nối qua lại trên internet giữa client và server. Giúp diễn ra nhanh chóng và hiệu quả hơn thông qua việc sử dụng TCP socket. Không chỉ sử dụng riêng cho ứng dụng web, Websocket có thể áp dụng cho bất kì ứng dụng nào khác cần có sự trao đổi thông tin trên Internet
    - **Ưu điểm:**
        - Tăng tốc độ truyền tải thông tin giữa 2 chiều
        - Dễ phát hiện và xử lý trong trường hợp có lỗi xảy ra
        - Dễ dàng sử dụng, không cần cài đặt thêm các phần mềm bổ sung khác
        - Không cần sử dụng nhiều phương pháp kết nối khác nhau
    - **Nhược điểm:**
        - Chưa hỗ trợ trên tất cả các trình duyệt
- **Unix socket:** Được dùng để giao tiếp giữa các ứng dụng khác nhau ngay trên cùng một máy tính. Các hoạt động của Unix socket được diễn ra ở nhân của hệ điều hành nên tốc độ kết nối và truyền tải giữa các ứng dụng nhanh, nhẹ và hiệu quả hơn.

- **HTTP :**
    - Giao thức truyền tải siêu văn bản. Được sử dụng với mục đích tạo nền tảng kết nối giữa Client và server. HTTP là một giao thức cho phép trao đổi và sử dụng các nguồn tài nguyên khác nhau, chẳng hạn như HTML doc. Một doc hoàn chỉnh sẽ được tạo nên từ nhiều doc con bao gồm văn bản, layout, media, video, script… HTTP sử dụng TCP làm giao thức nền tảng.