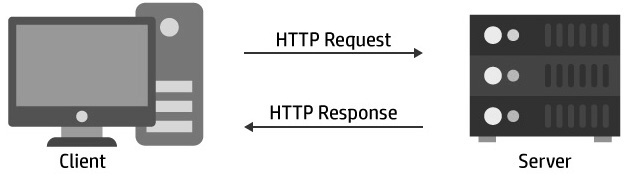
# 1-Khái niệm về giao thức HTTP, Http Request. Http Response

## Giao thức HTTP

* HTTP (HyperText Transfer Protocol) : Giao thức truyền tải siêu văn bản
* Là nền tẳng để trao đổi dữ liệu giữa các ứng dụng web



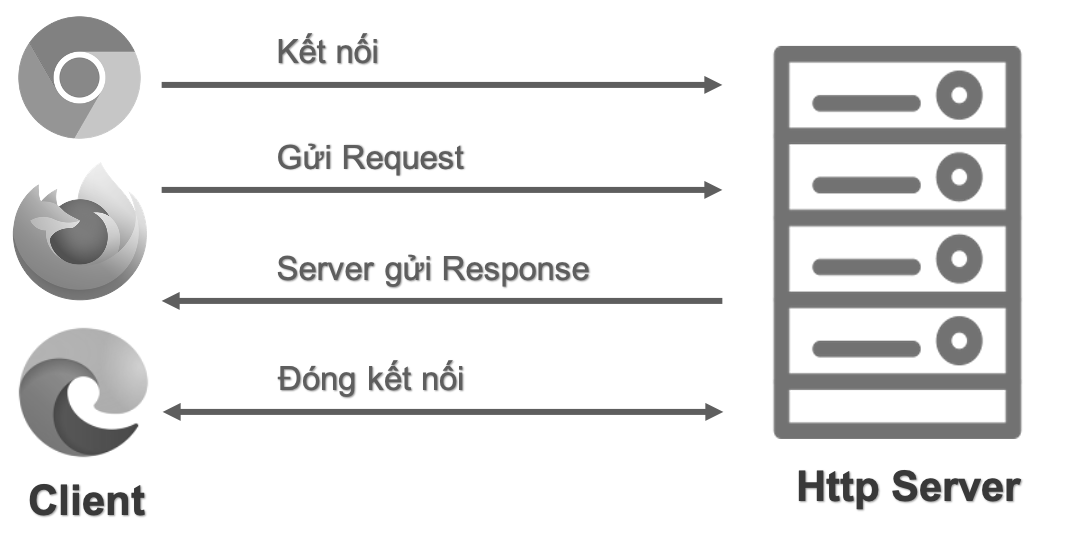
Thông thường thì có 1 máy chủ web lắng nghe, làm việc trên giao thức http

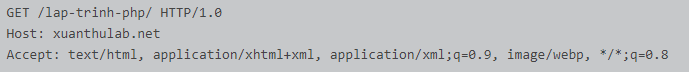
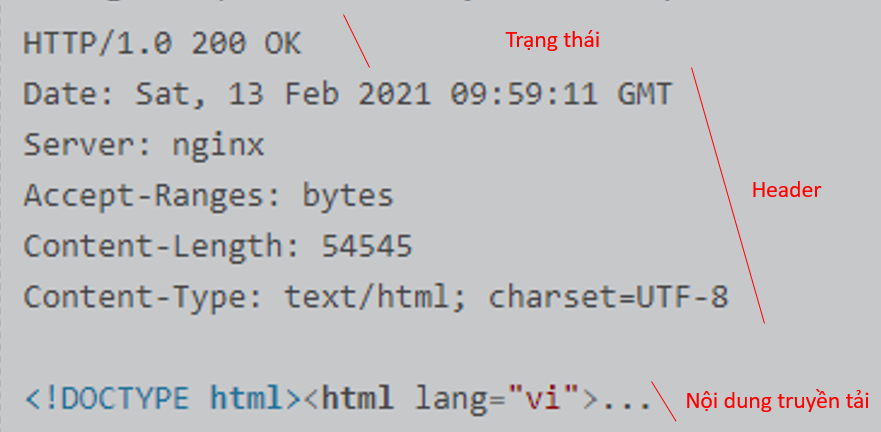
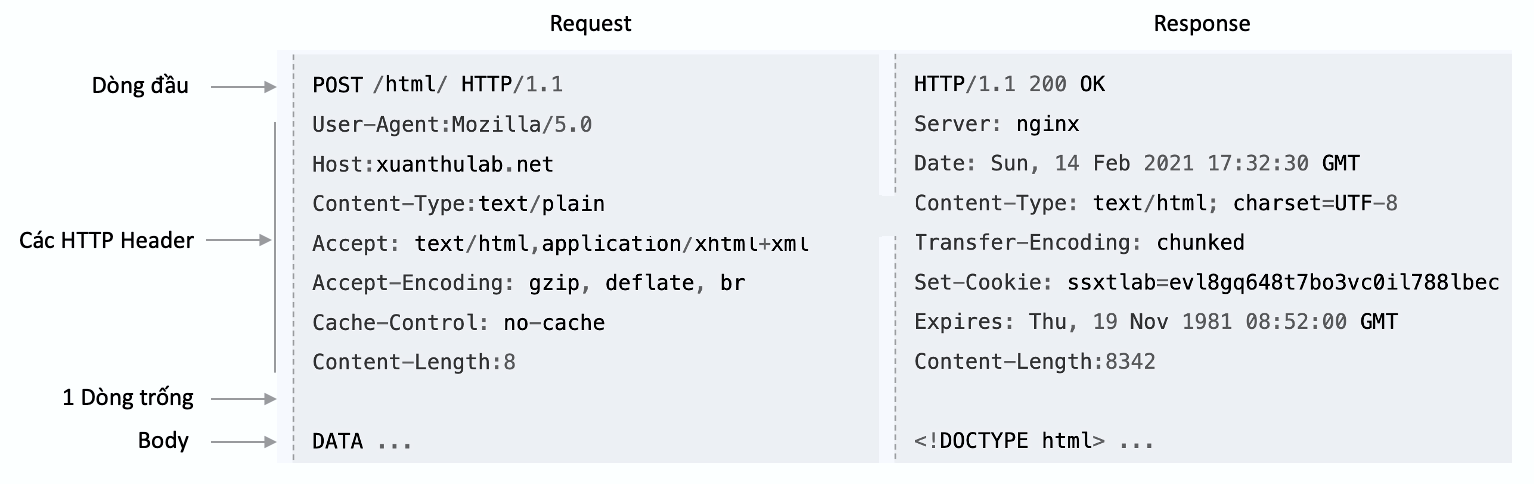
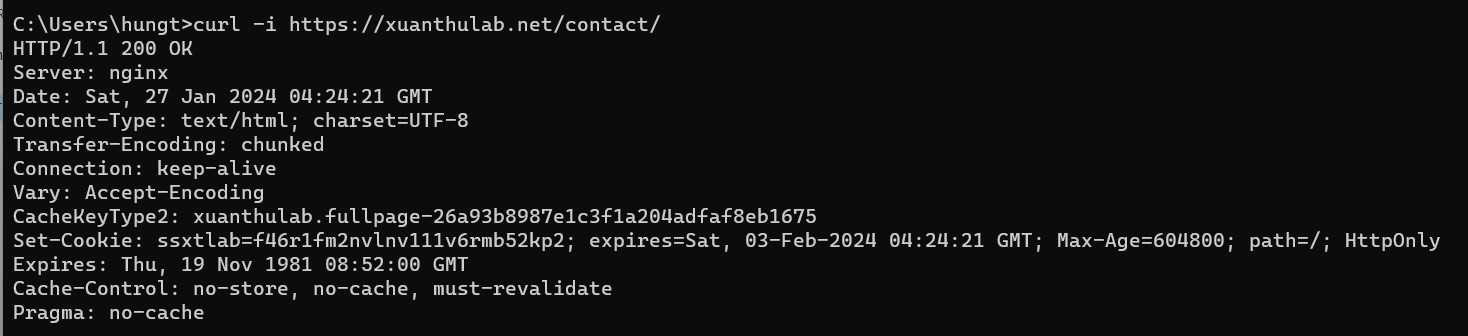
Các client (browser) gửi những yêu cầu, truy vấn đến trang web đó, những truy vấn này gọi là HTTP Request

Server nhận được những yêu cầu gửi đến, những yêu cầu này là những thông điệp dạng văn bản

Khi server nhận được yêu cầu này nó sẽ trả về cho client những thông điệp (HTTP Response)

## Sơ đồ làm việc HTTP truyền tải dữ liệu Clinet Server



* Tại các client (Các browser) tạo ra 1 kết nối tới Http Server
* Client gửi Request là những yếu cầu, yêu cầu này được gọi là Http message
* Http message là nội dung văn bản text, nó được định dạng theo một cấu trúc
* VD: 
* Khi Server nhận được Http message do client gửi đến nó cũng tạo ra một http message để trả về cho client
* Request và Response có cùng cấu trúc
* 
* Các HTTP Message do client gửi đi haowjc do server trả về chia sẻ cùng một cấu trúc
* 
* Tát cả nội dung này đều được truyền tải dưới dạng text
* Để kiểm tra nội dung HTTP Request và HTTP Response ta có thể kiểm tra qua dev tool của chrome trong phần network hoặc sử dụng terminal với lệnh: **curl -i <url>**
* 

# 2-URI, URN và URL

## URI

* **URI** (Uniform Resource Identifier) là một chuỗi ký tự để định vị tài nguyên (tài nguyên đó ở đâu, là gì trên internet)
* Các tài nguyên có để là tài liệu (document - text, html ...), hình ảnh (image) ..., **tài nguyên** chính là mục tiêu của HTTP Request.

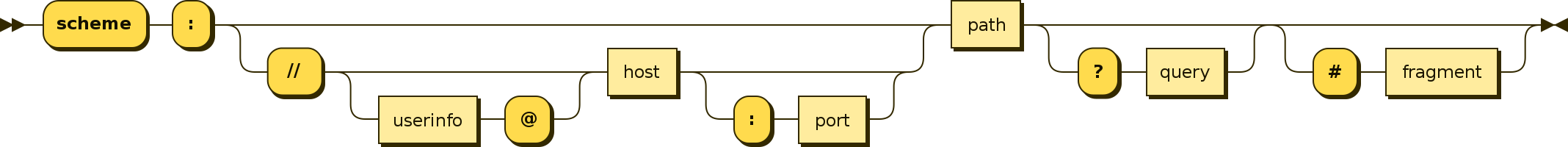
## URN

* **URN** (Uniform Resource Name) - *Định danh tài nguyên thống nhất*
* là một loại URI, xác định tài nguyên thông qua tên hợp lệ, duy nhất (các tài nguyên ở trường hợp này có thể là bất kỳ thứ gì)
* Cấu trúc của URN thường có dạng: **URN:NID:NSS** ví dụ như **URN:ISBN:8934974170617** trong đó:
  + **URN** chuỗi cho biết đây là URN
  + **NID** không gian tên (namespace) do IANA cấp (Internet Assigned Numbers Authority)
  + **NSS** chuỗi xác định đối tượng cụ thể

## URL

* **URL** - Uniform Resource Locator
* Là địa chỉ xác định tài nguyên trên Internet
* Là 1 loại URI được dùng trong các siêu văn bản (HTML)
* Được sử dụng bởi các browser (client) để lấy về hay cập nhật tài nguyên trên web

### Thành phần URL



VD:



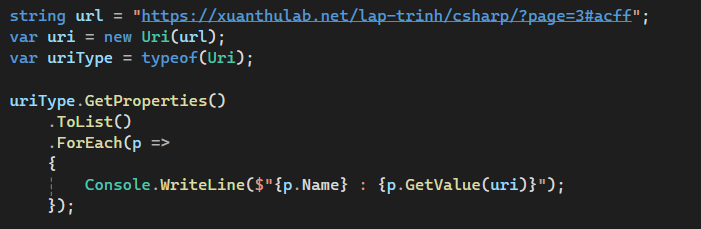
Nó có các thành phần:

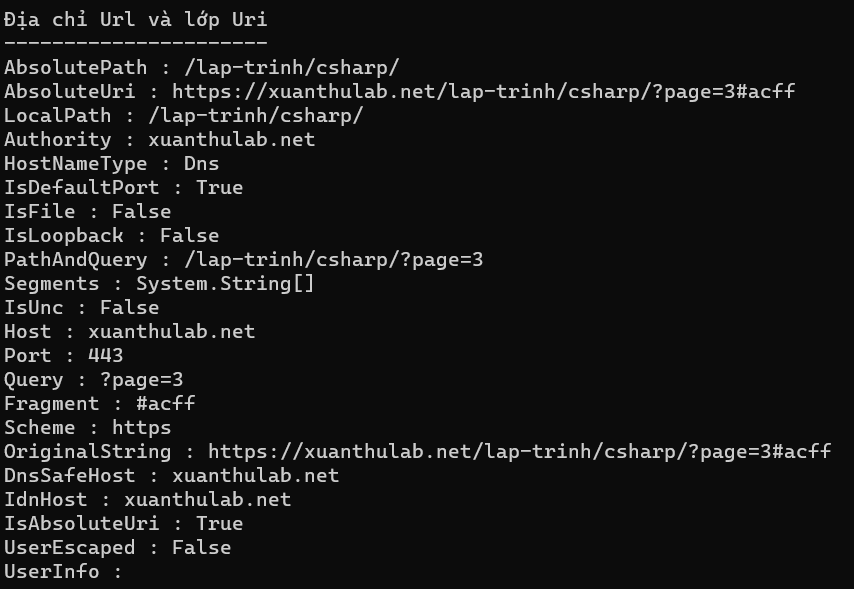
* **scheme** ví dụ https://, http://, ftp:// ... cho biết giao thức sử dụng để yêu cầu tài nguyên
* **host** hoặc domain (ví dụ xuanthulab.net) có thể có port ví dụxuanthulab.net:80 ... không cần chỉ ra nếu sử dụng cổng tiêu chuẩn (cổng 80 với http và 443 với https)
* **path** (ví dụ /path/to/page/) đường dẫn trên server dẫn tới tài nguyên, hiện nay không hẳn là một đường dẫn thực mà có thể là một logic ánh xạ bởi web server
* **query** là chuỗi truy vấn, nó chứa các tham số ví dụ ?a=1&b=price, bắt đầu chuỗi query là dấu ? mỗi tham số thường gồm key=value, các tham số cách nhau bởi &
* **fragment** (ví dụ #section), trỏ đến một phần củ thể trong tài nguyên, ví dụ một vị trí nào đó trong văn bản HTML.

# 3-Lớp URI

**System.Uri** là lớp biểu diễn địa chỉ URI (URL)

* Cho phép lấy các thông tin các thành phần của URL





# 4-Lớp tĩnh DNS và lớp IPHostEntry

## DNS

* Là hệ thống phân giải tên miền
* Giúp cho các trình client (như các trình duyệt) truy vấn để chuyển đổi một tên miền (như xuanthulab.net) sang địa chỉ IP vật lý tương ứng của tên miên đó
* Sau đó địa chỉ IP này được dùng để kết nối client/server.
* Dữ liệu DNS được lưu trữ và phục vụ truy vấn từ các Server DNS được vận hành bởi các nhà cung cấp dịch vụ và các tổ chức

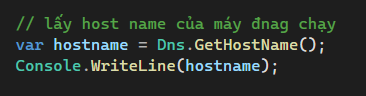
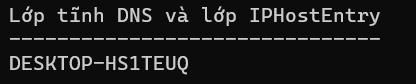
## Lớp tĩnh DNS

* Lớp **Dns** *(System.Net.Dns)* cung cấp các phương thức tính để lấy thông tin về host (địa chỉ website, server cung cấp các dịch vụ mạng) từ hệ thống phân giải tên miền (Dns)
* Các thông tin truy vấn được nó trả về một đối tượng giao diện **IPHostEntry**

### Các phương thức của lớp Dns

|  |  |
| --- | --- |
| GetHostName() | Lấy hostname của máy local |
| GetHostEntry(String) GetHostEntry(IPAddress) | Phân giải host hoặc IP thành đối tượng IPHostEntry.  Đối tượng kiểu IPHostEntry nó chứa thông tin địa chỉ về host. |

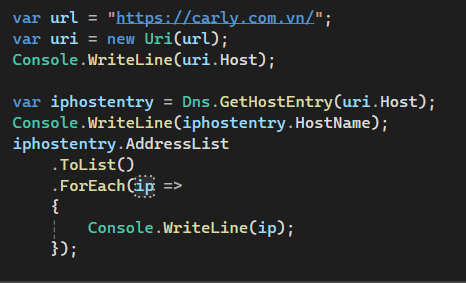
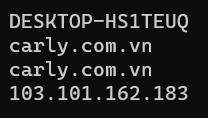
VD:



### Thuộc tính của IPHostEntry

|  |  |
| --- | --- |
| HostName | Chuỗi chứa hostname của Server |
| AddressList | Mảng các phần tử kiểu IPAddress chứa các địa chỉ IP |

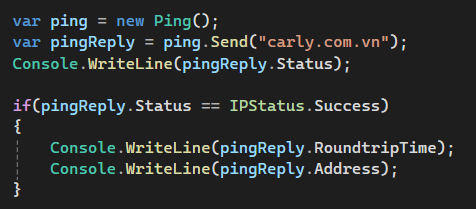
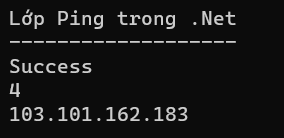
VD:

# 5-Lớp Ping

* Lớp **Ping** *(System.Net.NetworkInformation.Ping)*
* Lớp này cho phép ứng dụng xác định một máy từ xa (như server, máy trong mạng ...) có phản hồi không.

VD:

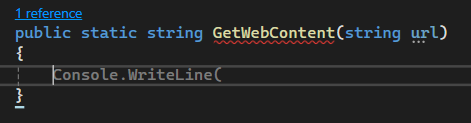
# 6-Sử dụng HttpClient gửi truy vấn GET tới Http Server

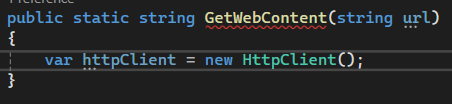
## HttpClient

* Lớp **HttpClient** được sử dụng để gửi truy vấn HTTP (Http Request Message - Request) và nhận phản hồi Response (Http Response Message) từ các truy vấn đó.
* Lớp này thuộc namespace **System.Net.Http**
* Namespace này chứa các lớp giúp tạo ra sự liên lạc giữa client và server.
* Để làm việc với HttpClient dùng những namespace sau:
  + **using** System;
  + **using** System.Linq;
  + **using** System.Net;
  + **using** System.Net.Http;
  + **using** System.Net.Http.Headers;
  + **using** System.Threading;
  + **using** System.Threading.Tasks;
  + **using** System.IO;
  + **using** System.Text;

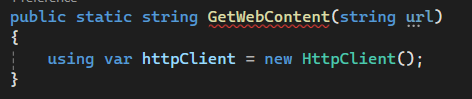
## Đọc nội dung trang web

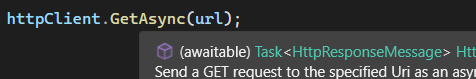
Ví dụ: xây dựng phương thức trả về nội dung trang web

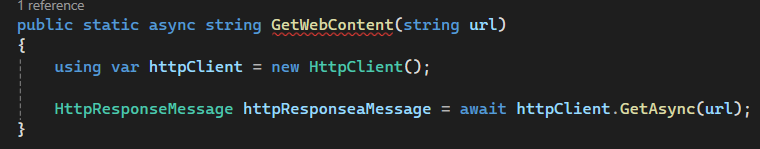
Ta khai báo 1 phương thức GetWebContent nhận vào 1 url và trả về nội dung trang web tức là 1 chuỗi string

Để thực hiện 1 truy vấn Http chúng ta cần tạo ra 1 đối tượng HttpClient

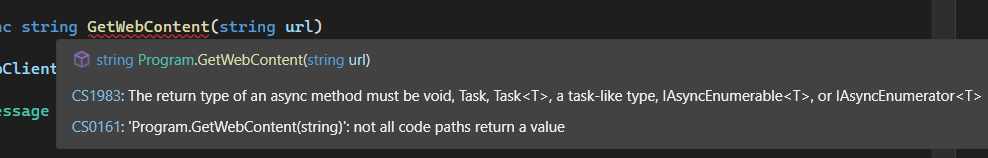
Để sử dụng được lớp HttpClient chúng ta cần nạp vào namespace **System.Net.Http**

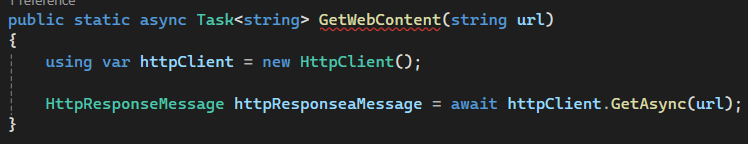
****Để đối tượng này tự động hủy khi thoát khỏi phương thức sử dụng tư khóa **using**

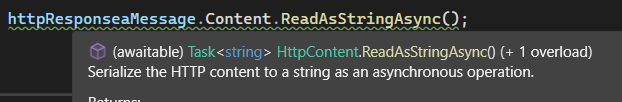
****Khi có đối tượng httpClient ta sẽ thực hiên truy vấn GET, ta sẽ gọi phương thức **GetAsync**, phương thức này là phương thức bất đồng bộ, nó trả về 1 Task vì vậy ta có thể lưu đối tượng **HttpRessponseMessage** này lại

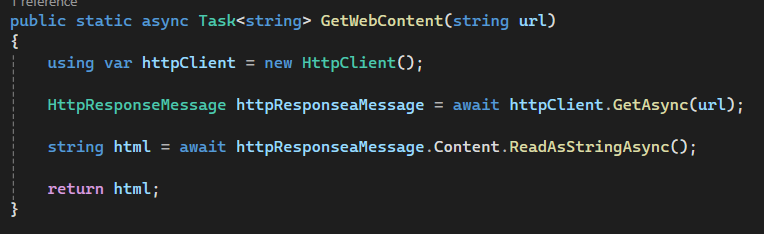
****Do phương thức trên là bất đồng bộ nên cần sử dụng từ khóa **async** và **await**

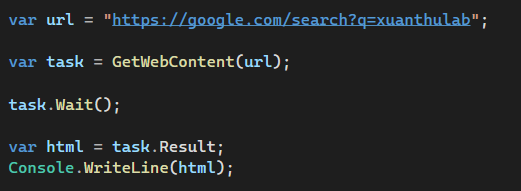
Lúc này ta nhận được lỗi:



Lý do vì phương thức bất đồng bộ chỉ có thể trả về void hoặc Task vì vậy ta sửa lại như sau

Khi có được HttpResponseMessage ta có thể đọc được nội dung của nó trong thuộc tính **Content** trong Content có 1 phương thức bất đồng bộ để độc nội dung trả về

Đây cũng là 1 phương thức bất đồng bộ trả về 1 Task<string> vì vậy ta phải sử dụng **await**

Sau khi xây dựng xong phương thức ta sẽ sử dụng như sau:

Khi ta gọi phương thức **GetWebContent** mặc dù phương thức này có return html nhưng do nó là một phương thức bất đồng bộ nên nó sẽ trả về 1 Task ở thời điểm await cuối cùng trong phương thức kết thúc vì vậy ta sẽ lưu task này vào 1 biến.

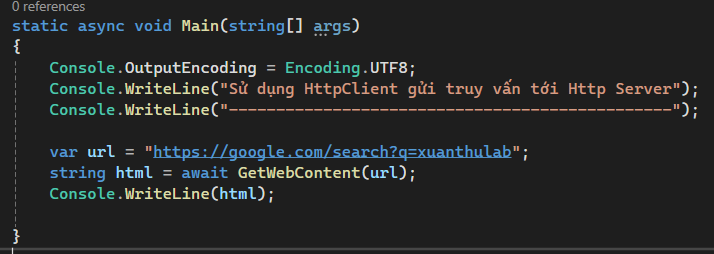
Sau đó ta sẽ cần đợi cho tác vụ này hoàn thành bằng cách sử dụng phương thức **Wait()**

Khi tác vụ này hoàn thành ta mới có thể đọc được html trả về

Kết quả:

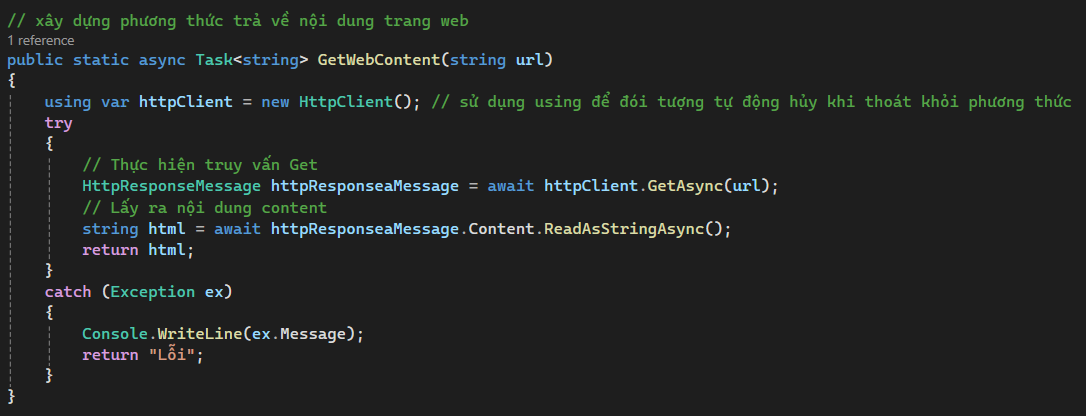


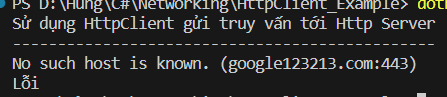
Chúng ta cũng có thể chuyển hàm main thành phương thức bất đồng bộ



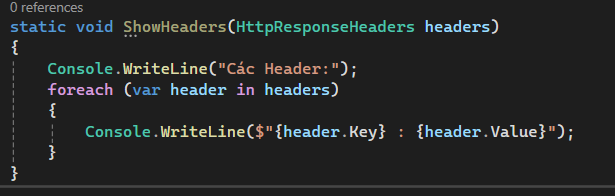
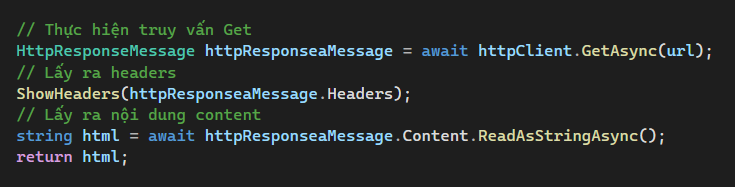
Kết quả cũng tương tự

Khi chúng ta thực hiện truy vấn GetAsync có thể xảy ra những exception vì vậy chúng ta cần phải bắt lỗi

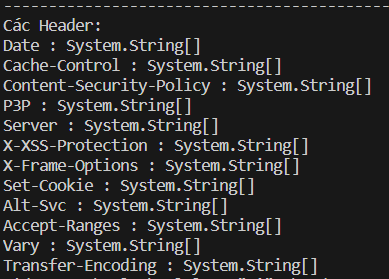




## Đọc nội dung headers

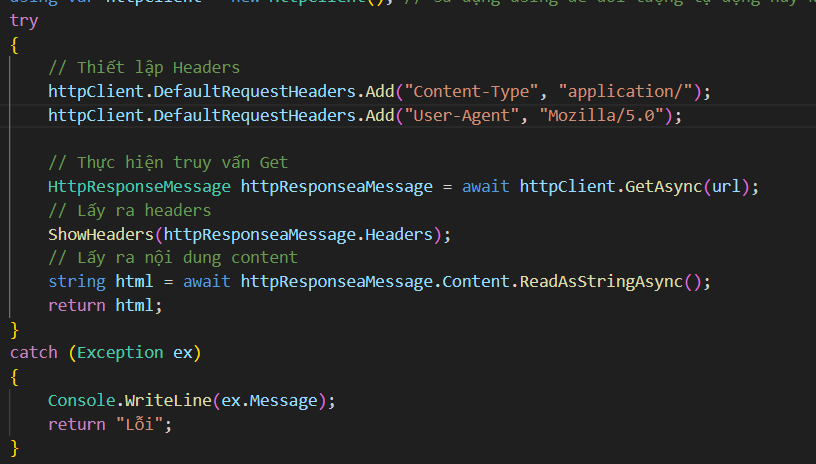
 

Kết quả:



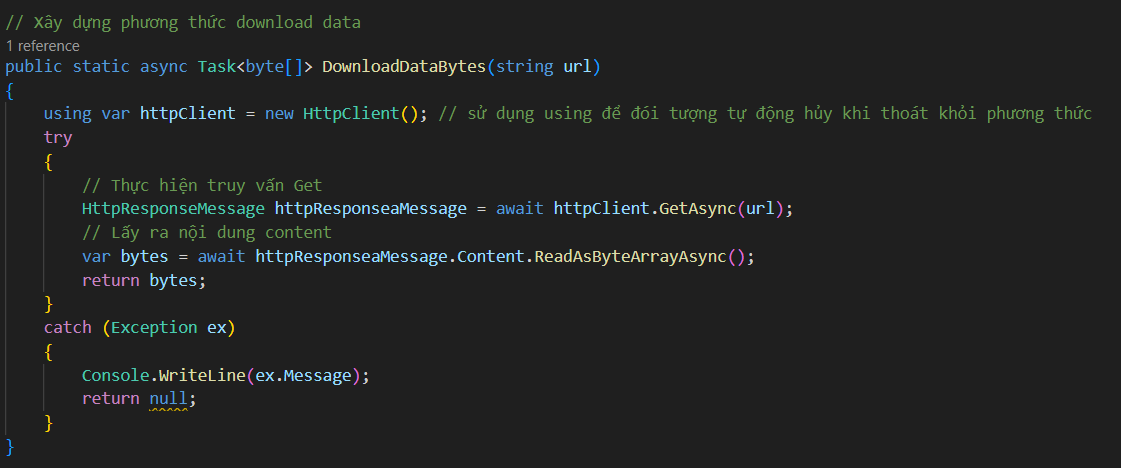
## Request Headers

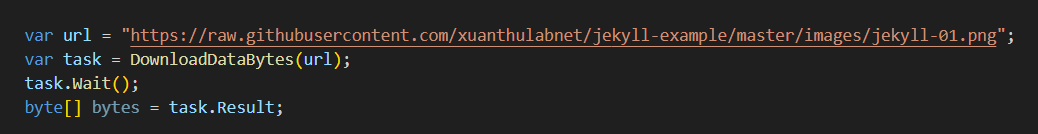
Khi chúng ta gọi các truy vấn, chúng ta cũng có thể thiết lập các header mà nó gửi đi



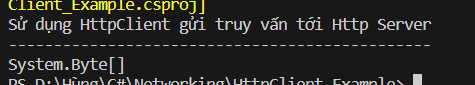
## Phương thức ReadAsByteArrayAsync

Bên trên ngoài cách đọc nội dụng trả về bằng phương thức **ReadAsStringAsync** chúng ta còn có thể đọc dữ liệu này như một mảng byte trả về (áp dụng trong trường hợp download file trên trang web)

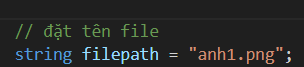




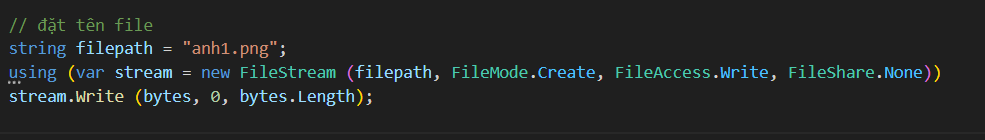
Sau khi thực thi ta nhận được một mảng byte



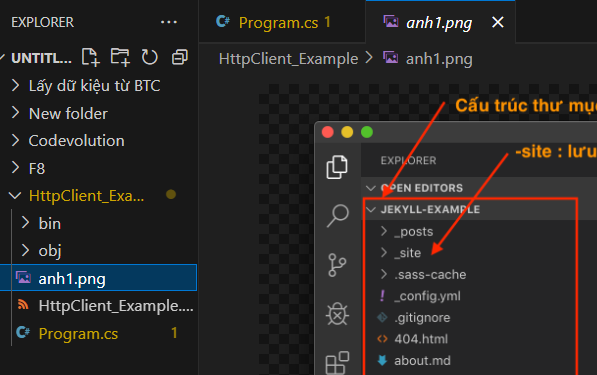
Chúng ta có thể lưu nó thành file



Sau đó ta sử dụng FileSream để mở file này ra và lưu mnagr byte đó vào file



Sau khi chạy ứng dụng ta được kết quả:



## Phương thức ReadAsStreamAsync

Phương thức này cũng dùng để đọc file theo byte tức là đọc theo từng đoạn, phương thức này tối ưu nhưng sử dụng khá phức tạp