# Mini HTTP Server

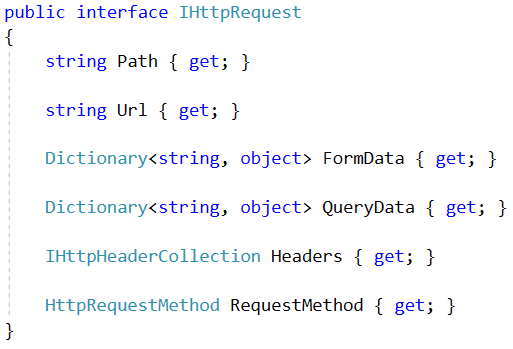
### Requests папка

Сега е време да съберем всичко написано до момента в главните функциониращи класове.

"Requests" папката ще съдържа класове и интерфейси за съхранение и манипулиране данни за HTTP заявките.

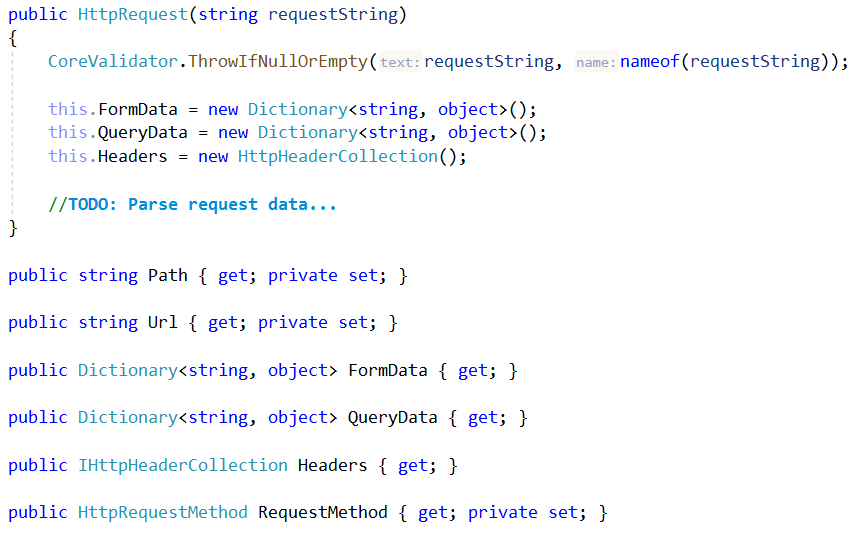
#### IHttpRequest

Създайте интерфейс, който се казва "IHttpRequest", който ще описва поведението на **Request** обекта.



#### HttpRequest

Създайте клас, който се казва "HttpRequest", който имплементира IHttpRequest интерфейса. Класът трябва да имплементира и методите на интерфейса.



.

Както виждате "HttpRequest", съдържа Path, Url, RequestMethod, Headers, Data. Тези данни идвам променлива "**requestString**", която се подава в констуктора. Това е начина, по които "HttpRequest" ще се инициализира.

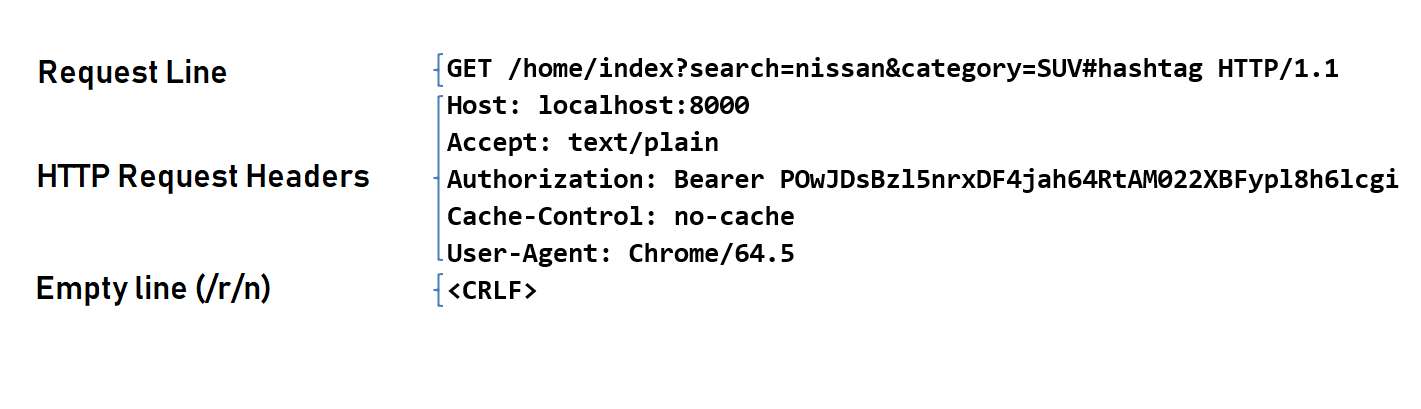
"**requestString**" ще изглежда по този начин:

{method} {url} {protocol}  
{header1key}: {header1value}  
{header2key}: {header2value}  
...  
<CRLF>  
{bodyparameter1key}={bodyparameter1value}&{bodyparameter2key}={bodyparameter2value}...

**ВНИМАНИЕ**: Както вече знаете, че body parameters не за задължителни.

Нека да разбием една нормална заявка и да видим как тя трябва да се мапне към нашите пропъртита.

##### GET заявка



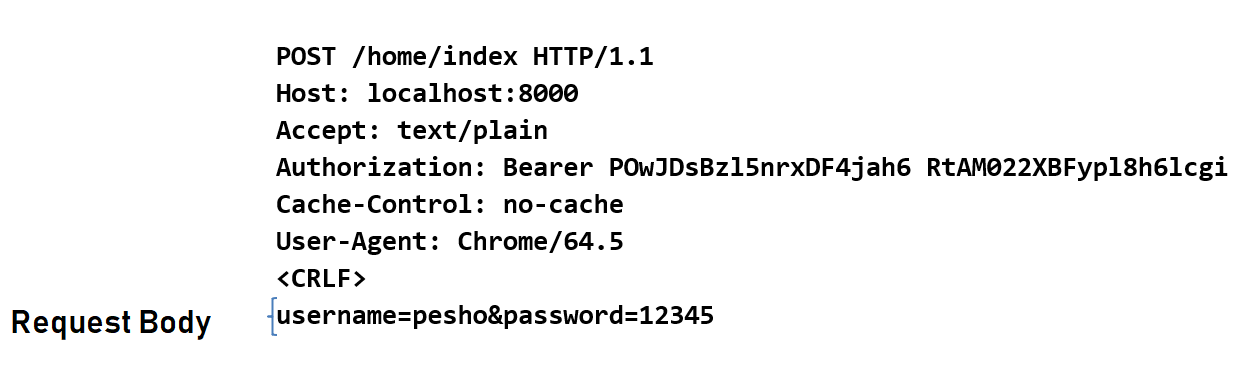
Request Line:

* The Request Method – Името на метода е винаги с **главни** букви, което означа, че някак си трябва да бъде форматиран, когато се парсва към **Enum-a** "HttpRequestMethod". (Не използвайте **switch/case** или **if/else** конструкции за преобразуването към **Enum**).
* The Request URL – Целият **URL**, съдържа "**Path**", "**Query String**" и "**Fragment**".
  + Вземете "Path" частта от URL-a, като го разделите, форматирате и запишете стойността в "Path" пропъртито.
  + Вземете "Query String" частта и добавете стойностите към "Query Data" речника.  
    Параметрите трябва да бъдат преобразувани по следният начин": parameterName = key, parameterValue = value.
  + Fragments are mostly used on the client side, so there is no need to store them in our class, thus there is no property for them.
  + Fragments предимно се използва в "client-side" частта, затова няма пропърти за тях.
* The Request Protocol – **Трябва** да бъде: "HTTP/1.1".

Те лесно могат да се преобразуват в следният формат: "{key}: {value}". Трябва да ги разделите и да създадете нова инстанция на "**HttpHeader**", и след това да се добави към "**Headers**" на "**Request**".

Empty Line –краят на "**Request** **Headers**"

##### POST заявка

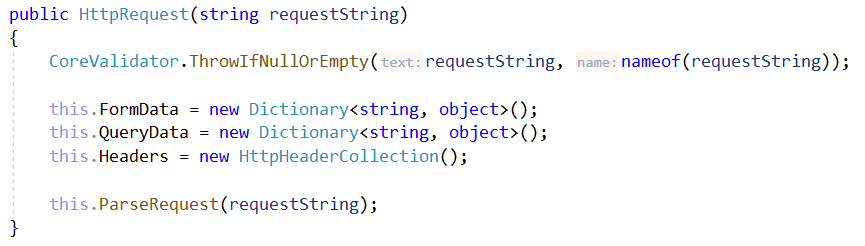


"POST Request" е почти същият, освен неговото "**body**". "**Request Body**" съдържа параметри, които трябва да бъдат прехвърлени към "**Form Data**" речника, по същият начин, както "**Query Parameters**" бяха прехвърлени към "**Query Data**" речника.

Сега е време да имплементираме повече логика, което означава много методи, ако искаме да спазваме принципите за "**High-Quality Code**". Имплементирайте следните методи.



ParseRequest() е метода откъдето започва всичко:



Нека да видим как изглежда той:



Както виждате "**requestString**" е разделен на нови редове в масив. Взимаме първият ред (The **Request Line**) и го разделяме. След това следват серия от проверки и присвояване не стойности към пропъртита.

Ще се наложи вие да имплементирате тези методи. Разбира се, ще ви бъдат дадени насоки, как да се справите с тях.

##### IsValidRequestLine() метод

Този метод проверя дали, разделеният "requestLine" съдържа точно 3 елемента и също така дали последният елемент е равен на "HTTP/1.1". Метода връща булев резултат.

##### IsValidRequestQueryString() метод

Този метод се използва в "ParseQueryParameters()" метода. Проверява дали "Query" низа е **NOT** **NULL** или **празен** и също така дали има поне **един** или много queryParameters.

##### ParseRequestMethod() метод

**RequestMethod** присвоя стойността, като преобразуваме първият елемент от разделеният "requestLine".

##### ParseRequestUrl() метод

**Url** присвоява стойността от вторият елемент на разделеният "requestLine".

##### ParseRequestPath() метод

Path присвоява стойността, като разделим **Url** и вземем само пътя от него.

##### ParseHeaders() метод

Пропускаме първият ред от "requestLine" и обхождаме всички останали редове, докато не стигнем празен ред. Всеки ред представлява "header", който трябва да бъде разделен и преобразуван към правилният тип. След това информацията от низа е прехвърлена към "HttpHeader" обекта и е добавен към "Headers" пропертито на "Request".

Хвърлете "BadRequestException", ако "Host" липсва след преобразуването.

##### ParseQueryParameters() метод

Извадете "Query" низа, като разделите "Request's Url" и вземете само "query" от него. След това разделете "Query" низа в различни параметри и го прехвърлете към "Query Data Dictionary".

Валидирайте "Query" низа, като извикате "IsValidrequestQueryString()"метода.

Ако в " Request's Url" липсва " Query" низа, не предприемайте действия.

Хвърлете "BadRequestException", ако "Query" не е валиден.

##### ParseFormDataParameters() метод

Разделете "Request's Body " в различни параметри и го добавате към "Form Data Dictionary".

Не предприемайте действия, ако "Request" не съдържа тяло.

##### ParseRequestParameters() метод

Този метод извиква "ParseQueryParameters()" и " ParseFormDataParameters()" методите. Това е просто "**wrapping**" метод.

Ако сте имплементирали всички правилно, би трябвало да преобразувате дори и много сложни заявки без проблем.