The function below ilustrates us all the data from any Http Response. As a parameter it takes the content i.e. `Task<HttpResponseMessage>`. This method is static and asynchronous one. I have extract all the possible data which may be given to HttpClient as a response to his/her queries.

```
        async static void ShowContent(Task<HttpResponseMessage> content)
        {
            Console.WriteLine("Http Version: " + content.Result.Version + "\n");

            Console.WriteLine("Http StatusCode: " + content.Result.StatusCode +
"\n");

            Console.WriteLine("Http RequestMessqge: " +
content.Result.RequestMessage + "\n");

            Console.WriteLine("Http ReasonPhrase: " + content.Result.ReasonPhrase
+ "\n");

            Console.WriteLine("Http AcceptRanges: " +
content.Result.Headers.AcceptRanges + "\n");

            Console.WriteLine("Http Age: " + content.Result.Headers.Age + "\n");

            Console.WriteLine("Http CacheControl: " +
content.Result.Headers.CacheControl + "\n");

            Console.WriteLine("Http Connection: " +
content.Result.Headers.Connection + "\n");

            Console.WriteLine("Http ConnectionClose: " +
content.Result.Headers.ConnectionClose + "\n");

            Console.WriteLine("Http Date: " + content.Result.Headers.Date.Value +
"\n");

            Console.WriteLine("Http ETag: " + content.Result.Headers.ETag + "\n");

            Console.WriteLine("Http Location: " + content.Result.Headers.Location
+ "\n");

            Console.WriteLine("Http Pragma: " + content.Result.Headers.Pragma +
"\n");

            Console.WriteLine("Http ProxyAuthenticate: " +
content.Result.Headers.ProxyAuthenticate + "\n");

            Console.WriteLine("Http RetryAfter: " +
content.Result.Headers.RetryAfter + "\n");

            Console.WriteLine("Http Server: " + content.Result.Headers.Server +
"\n");
```

```
            Console.WriteLine("Http Trailer: " + content.Result.Headers.Trailer +
"\n");

            Console.WriteLine("Http TransferEncoding: " +
content.Result.Headers.TransferEncoding + "\n");

            Console.WriteLine("Http TransferEncodingChunked: " +
content.Result.Headers.TransferEncodingChunked + "\n");

            Console.WriteLine("Http Upgrade: " + content.Result.Headers.Upgrade +
"\n");

            Console.WriteLine("Http Vary: " + content.Result.Headers.Vary + "\n");

            Console.WriteLine("Http Via: " + content.Result.Headers.Via + "\n");

            Console.WriteLine("Http Warning: " + content.Result.Headers.Warning +
"\n");

            Console.WriteLine("Http WwwAuthenticate: " +
content.Result.Headers.WwwAuthenticate + "\n");

            Console.WriteLine("Http Content: " + await
content.Result.Content.ReadAsStringAsync() + "\n");

            Console.WriteLine("Http Headers: " + content.Result.Headers + "\n");
        }
```

In order to process the GET request response:

```
 static async Task<HttpResponseMessage> HttpGetRequest_response(string url)
{
    using (HttpClient client = new HttpClient())
    {
        HttpResponseMessage response = await client.GetAsync(url);
        {
            return response;
        }
    }
}
```

All this method needs is an URL to web resource. Inside the method I'm crearing the `HttpClient` object in `using` block and I'm doing the next manipulations on the data. The fact that the object was created in the using block says us that inside it was implemented the `Dispose` method and it will be automatically applied when we'll live the respective block of code.

As a result the method returns the response on client's request.

```
string url = "https://httpbin.org/get";
 // full data on GET request
using (var content = HttpGetRequest_response(url))
{
    ShowContent(content);
}
```

All we need in the following step is to use the response of the previous method in the `using` block and to unpack the result of the returning value. In order to read all the possible information we should enter the respective content in the `Showcontent` method.

If the type of the return object is asyncronous one, we should write the *await* keyword in order to do any manipulation on the respective data

For another requests the logic is the same, but the main difference is that some of them require additional parameters:

POST

```
static async Task<HttpResponseMessage> HttpPostRequest_response(string url,
List<KeyValuePair<string, string>> iterable)
{
    using (HttpClient client = new HttpClient())
    {
        using (HttpContent queries = new FormUrlEncodedContent(iterable))
        {
            HttpResponseMessage response = await client.PostAsync(url, queries);
            return response;
        }
    }
}
```

DELETE

```
static async Task<HttpResponseMessage> HttpDeleteRequest_response(string url)
{
    using (HttpClient client = new HttpClient())
    {
        HttpResponseMessage response = await client.DeleteAsync(url);
        return response;
    }
}
```

PUT

```
static async Task<HttpResponseMessage> HttpPutRequest_response(string url,
List<KeyValuePair<string, string>> iterable)
{
    using (HttpClient client = new HttpClient())
    {
        using (HttpContent queries = new FormUrlEncodedContent(iterable))
        {
            HttpResponseMessage response = await client.PutAsync(url, queries);
            return response;
        }
    }
}
```

Calling methods in the programm:

```
        static void Main(string[] args)
        {
            string url = "https://httpbin.org/get";

            // full data on GET request
            using (var content = HttpGetRequest_response(url))
            {
                ShowContent(content);
            }


Console.Write("==========================================================================
===\n\n");

            url = "https://httpbin.org/post";

            // full data on Post request
            List<KeyValuePair<string, string>> queries = new
List<KeyValuePair<string, string>>()
            {
                new KeyValuePair<string, string>("accept", "application/json")
            };

            using (var content = HttpPostRequest_response(url, queries))
            {
                ShowContent(content);
            }


Console.Write("==========================================================================
===\n\n");

            url = "https://httpbin.org/put";
            using (var content = HttpPutRequest_response(url, queries))
            {
                ShowContent(content);
```

```
            }


Console.Write("============================================================================
===\n\n");

            url = "https://httpbin.org/delete";
            using (var content = HttpDeleteRequest_response(url))
            {
                ShowContent(content);
            }


Console.Write("============================================================================
===\n\n");

            Console.ReadKey();
        }
```

Results:

```
Http Version: 1.1

Http StatusCode: OK

Http RequestMessqge: Method: GET, RequestUri: 'https://httpbin.org/get', Version:
1.1, Content: <null>, Headers:
{
}

Http ReasonPhrase: OK

Http AcceptRanges:

Http Age:

Http CacheControl:

Http Connection: keep-alive

Http ConnectionClose:

Http Date: 10.03.2019 9:53:13 +00:00

Http ETag:

Http Location:

Http Pragma:

Http ProxyAuthenticate:
```

```
Http RetryAfter:

Http Server: nginx

Http Trailer:

Http TransferEncoding:

Http TransferEncodingChunked:

Http Upgrade:

Http Vary:

Http Via:

Http Warning:

Http WwwAuthenticate:

Http Content: {
  "args": {},
  "headers": {
    "Host": "httpbin.org"
  },
  "origin": "92.115.245.74, 92.115.245.74",
  "url": "https://httpbin.org/get"
}


Http Headers: Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Connection: keep-alive
Date: Sun, 10 Mar 2019 09:53:13 GMT
Server: nginx


========================================================================

Http Version: 1.1

Http StatusCode: OK

Http RequestMessqge: Method: POST, RequestUri: 'https://httpbin.org/post',
Version: 1.1, Content: System.Net.Http.FormUrlEncodedContent, Headers:
{
  Content-Type: application/x-www-form-urlencoded
  Content-Length: 25
}

Http ReasonPhrase: OK

Http AcceptRanges:
```

```
Http Age:

Http CacheControl:

Http Connection: keep-alive

Http ConnectionClose:

Http Date: 10.03.2019 9:53:14 +00:00

Http ETag:

Http Location:

Http Pragma:

Http ProxyAuthenticate:

Http RetryAfter:

Http Server: nginx

Http Trailer:

Http TransferEncoding:

Http TransferEncodingChunked:

Http Upgrade:

Http Vary:

Http Via:

Http Warning:

Http WwwAuthenticate:

Http Content: {
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "accept": "application/json"
  },
  "headers": {
    "Content-Length": "25",
    "Content-Type": "application/x-www-form-urlencoded",
    "Host": "httpbin.org"
  },
  "json": null,
  "origin": "92.115.245.74, 92.115.245.74",
  "url": "https://httpbin.org/post"
}
```

```
Http Headers: Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Connection: keep-alive
Date: Sun, 10 Mar 2019 09:53:14 GMT
Server: nginx


========================================================================

Http Version: 1.1

Http StatusCode: OK

Http RequestMessqge: Method: PUT, RequestUri: 'https://httpbin.org/put', Version:
1.1, Content: System.Net.Http.FormUrlEncodedContent, Headers:
{
  Content-Type: application/x-www-form-urlencoded
  Content-Length: 25
}

Http ReasonPhrase: OK

Http AcceptRanges:

Http Age:

Http CacheControl:

Http Connection: keep-alive

Http ConnectionClose:

Http Date: 10.03.2019 9:53:15 +00:00

Http ETag:

Http Location:

Http Pragma:

Http ProxyAuthenticate:

Http RetryAfter:

Http Server: nginx

Http Trailer:

Http TransferEncoding:

Http TransferEncodingChunked:
```

```
Http Upgrade:

Http Vary:

Http Via:

Http Warning:

Http WwwAuthenticate:

Http Content: {
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "accept": "application/json"
  },
  "headers": {
    "Content-Length": "25",
    "Content-Type": "application/x-www-form-urlencoded",
    "Host": "httpbin.org"
  },
  "json": null,
  "origin": "92.115.245.74, 92.115.245.74",
  "url": "https://httpbin.org/put"
}


Http Headers: Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Connection: keep-alive
Date: Sun, 10 Mar 2019 09:53:15 GMT
Server: nginx


================================================================================

Http Version: 1.1

Http StatusCode: OK

Http RequestMessqge: Method: DELETE, RequestUri: 'https://httpbin.org/delete',
Version: 1.1, Content: <null>, Headers:
{
}

Http ReasonPhrase: OK

Http AcceptRanges:

Http Age:

Http CacheControl:
```

```
Http Connection: keep-alive

Http ConnectionClose:

Http Date: 10.03.2019 9:53:15 +00:00

Http ETag:

Http Location:

Http Pragma:

Http ProxyAuthenticate:

Http RetryAfter:

Http Server: nginx

Http Trailer:

Http TransferEncoding:

Http TransferEncodingChunked:

Http Upgrade:

Http Vary:

Http Via:

Http Warning:

Http WwwAuthenticate:

Http Content: {
  "args": {},
  "data": "",
  "files": {},
  "form": {},
  "headers": {
    "Host": "httpbin.org"
  },
  "json": null,
  "origin": "92.115.245.74, 92.115.245.74",
  "url": "https://httpbin.org/delete"
}


Http Headers: Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: *
Connection: keep-alive
Date: Sun, 10 Mar 2019 09:53:15 GMT
Server: nginx
```

```
==========================================================================
```