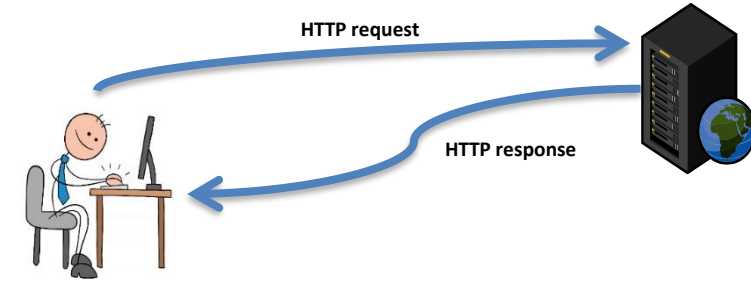


HTTP Protocol Headers

Web Engineering

Recovering HTTP communication....



HTTP request message
(Port 80 of TCP)

HTTP Client
(for ex.,
the browser)

HTTP Server
(Apache, IIS,
etc.)

HTTP response message



Recovering HTTP communication....

HTTP Messages

Request

Request line CRLF

Header lines CRLF

CRLF

Body

Response

Status line CRLF

Header lines CRLF

CRLF

Body

HTTP Headers ???

HTTP headers let the client and the server pass additional information with an HTTP request or response. They contain metadata in key-value pairs that are sent along with HTTP requests and responses.

Headers can be grouped according to their contexts:

- **General Header:** This type of headers applied on Request and Response headers both but without affecting the database body.
- **Request Header:** contains more information about the resource to be fetched, or about the client requesting the resource.
- **Response Header:** hold additional information about the response, like its location or about the server providing it.
- **Entity Header:** contains the information about the body of the resources like MIME type, Content-length.

Network Working Group
Request for Comments: 2616
Obsoletes: 2068
Category: Standards Track

Everything about HTTP

R. Fielding
UC Irvine
J. Gettys
Compaq/W3C
J. Mogul
Compaq
H. Frystyk
W3C/MIT
L. Masinter
Xerox
P. Leach
Microsoft
T. Berners-Lee
W3C/MIT
June 1999

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

14	Header Field Definitions	100
14.1	Accept	100
14.2	Accept-Charset	102
14.3	Accept-Encoding	102
14.4	Accept-Language	104
14.5	Accept-Ranges	105
14.6	Age	106
14.7	Allow	106
14.8	Authorization	107
14.9	Cache-Control	108
14.9.1	What is Cacheable	109
14.9.2	What May be Stored by Caches	110
14.9.3	Modifications of the Basic Expiration Mechanism	111
14.9.4	Cache Revalidation and Reload Controls	113
14.9.5	No-Transform Directive	115
14.9.6	Cache Control Extensions	116
14.10	Connection	117
14.11	Content-Encoding	118
14.12	Content-Language	118
14.13	Content-Length	119
14.14	Content-Location	120
14.15	Content-MD5	121
14.16	Content-Range	122
14.17	Content-Type	124
14.18	Date	124
14.18.1	Clockless Origin Server Operation	125
14.19	ETag	126
14.20	Expect	126
14.21	Expires	127
14.22	From	128

Some headers...

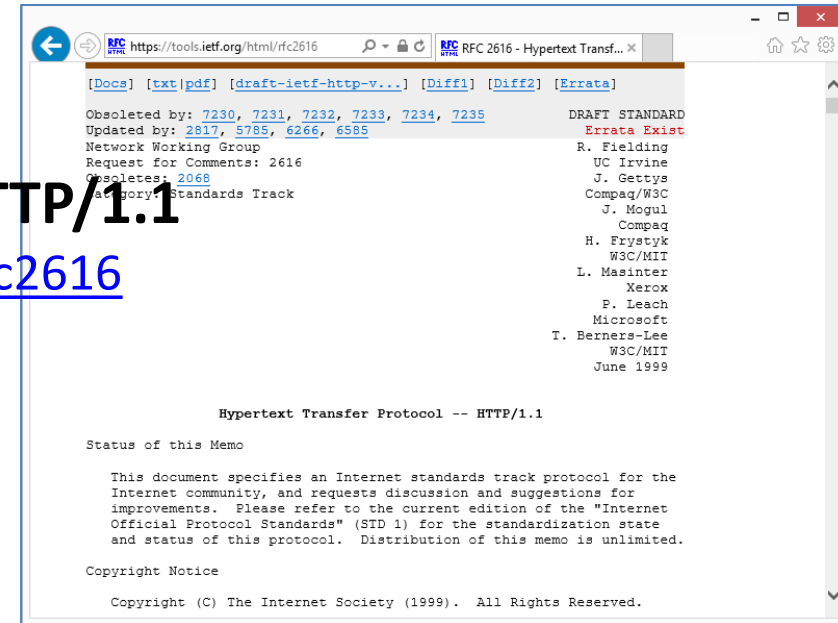
14.23	Host	128
14.24	If-Match	129
14.25	If-Modified-Since	130
14.26	If-None-Match	132
14.27	If-Range	133
14.28	If-Unmodified-Since	134
14.29	Last-Modified	134
14.30	Location	
14.31	Max-Forwards	
14.32	Pragma	
14.33	Proxy-Authenticate	137
14.34	Proxy-Authorization	137
14.35	Range	138
14.35.1	Byte Ranges	138
14.35.2	Range Retrieval Requests	139
14.36	Referer	140
14.37	Retry-After	141
14.38	Server	141
14.39	TE	142
14.40	Trailer	143
14.41	Transfer-Encoding.....	143
14.42	Upgrade	144
14.43	User-Agent	145
14.44	Vary	145
14.45	Via	146
14.46	Warning	148
14.47	WWW-Authenticate	150

...more headers...

Complete header's list:

Hypertext Transfer Protocol -- HTTP/1.1

<https://datatracker.ietf.org/doc/html/rfc2616>



Let's see some examples

GET /tc/home.html HTTP/1.1

Host: www.tribunalconstitucional.pt

Date: Tue, 30 Sep 2008 13:45:29 GMT

User-Agent: Mozilla/5.0 (Windows; U; Windows
NT 6.0; en-US; rv:1.9.0.3) Gecko/2008092417
Firefox/3.0.3

Referer: http://home.utad.pt/~lfb/teste.htm

If-Modified-Since: Tue, 30 Sep 2008 13:40:29 GMT

HTTP/1.1 200 OK

Server: Microsoft-IIS/4.0

Date: Tue, 30 Sep 2008 13:45:38 GMT

Content-Type: text/html

Accept-Ranges: bytes

Last-Modified: Wed, 06 Jan 1999 18:56:06 GMT

Content-Length: 9934

Who (and when) sets (inserts) the request headers?

- Web clients (browsers) following a request for a resource and ... web application's code

Who (and when) sets (inserts) the headers of static content responses?

- Web servers following the receipt of a content request

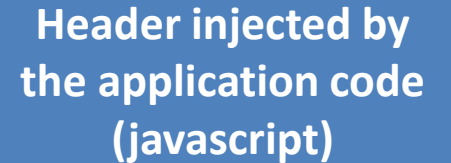
Who (and when) defines (inserts) the headers of the dynamic content responses?

- Web servers following the receipt of a resource request and ... web application's code.

Programmatically in the browser (request)

```
function ajax(url, vars, callbackFunction) {  
    var request = new XMLHttpRequest();  
    request.open("POST", url, true);  
    request.setRequestHeader(  
        "Content-Type",  
        "application/x-www-form-urlencoded");  
    request.onreadystatechange = function() {  
        if (request.readyState == 4 &&  
            request.status == 200) {  
            if (request.responseText) {  
                callbackFunction(request.responseText);  
            }  
        }  
    };  
    request.send(vars);  
}
```

Header injected by
the application code
(javascript)



Programmatically in the server (response)

```
@HttpContext.Current.Response.AddHeader("CustomHeader", "CustomValue")
```

```
<HTML>
```

```
<TITLE>Test</TITLE>
```


```
<BODY>
```

```
<p>This page has a custom HTTP header</p>
```

```
</BODY>
```

```
</HTML>
```

**Header injected by
the application code
(Razor)**



Dynamic content response

In the server (response)

This always work?

```
<HTML>  
<TITLE>Test</TITLE>  
<BODY>  
<p>This page has a custom HTTP header</p>  
@HttpContext.Current.Response.AddHeader("CustomHeader", "CustomValue")  
</BODY>  
</HTML>
```

Dynamic content response

Current servers implement response buffering:

When the output is buffered, the server will hold back the response to the browser until all of the server scripts have been processed, or until the script calls the Flush or End method.

Examples of buffer manipulation:

```
@HttpContext.Current.Response.ClearContent();  
@HttpContext.Current.Response.ClearHeaders();  
@HttpContext.Current.Response.Buffer = true;  
@HttpContext.Current.Response.End();
```

Programmatically changing headers...

A **forbidden header name** is the name of any HTTP header that cannot be modified programmatically.

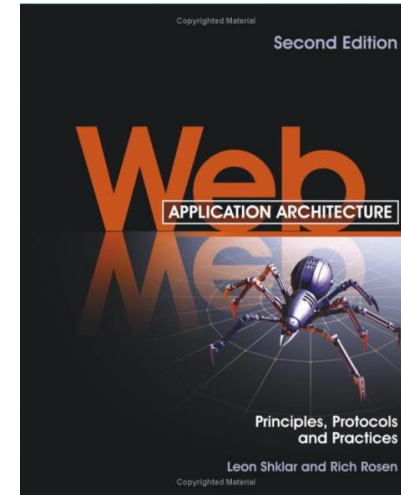
Modifying such headers is forbidden because the user agent retains full control over them.

Forbidden header names start with Proxy- or Sec-, or are one of the following names:

- Accept-Charset
- Accept-Encoding
- Connection
- Content-Length
- Origin
- Cookie
- Date
- Host
- Keep-Alive
- Referer
- ...

Associated Readings

Web Application Architecture, Second Edition
Cap. 3: “Birth of the Web: HTTP”, pages 44 to 60.



Hypertext Transfer Protocol -- HTTP/1.1
<http://www.w3.org/Protocols/rfc2616/rfc2616.html>



MIME types in HTTP protocol

Web Engineering

Multipurpose Internet Mail Extensions

HTTP/1.1 200 OK

Server: Microsoft-IIS/4.0

Date: Tue, 30 Sep 2008 13:45:38 GMT

Content-Type: text/html

Accept-Ranges: bytes

Last-Modified: Wed, 06 Jan 1999 18:56:06 GMT

Content-Length: 9934

MIME encoding

Content-Type: type/subtype [; parameters]

Content-Type: text/html

Content-Type: text/plain

Content-Type: text/css

Content-Type: image/gif

Content-Type: image/jpeg

Content-Type: audio/x-wav

Content-Type: audio/x-mpeg-2

Content-Type: video/quicktime

Content-Type: video/mpeg-2

MIME encoding

Content-Type: type/subtype [; parameters]

Content-Type: text/html ; charset = us-ascii

Content-Type: text/html ; charset = utf-8

(The default character set is [ISO-8859-1](#).)

MIME encoding

Content-Type: type/subtype [; parameters]

```
Content-Type: multipart/x-mixed-replace ;  
                boundary=XPTOText
```

```
. . .
```

```
--XPTOText
```

```
Content-Type: image/gif
```

```
. . .
```

```
--XPTOText
```

```
Content-Type: image/gif
```

```
. . .
```

```
--XPTOText--
```

Content-Type: multipart/form-data ; boundary=...

MIME encoding

Content-Type: type/subtype [; parameters]

Accept-Encoding: x-compress; x-zip

```
GET /encrypted-area HTTP/1.1  
Host: www.example.com  
Accept-Encoding: gzip, deflate
```

```
HTTP/1.1 200 OK  
Date: mon, 28 Sep 2015 22:38:34 GMT  
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)  
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT  
Accept-Ranges: bytes  
Content-Length: 438  
Connection: close  
Content-Type: text/html; charset=UTF-8  
Content-Encoding: gzip
```

MIME encoding

Base64

often used to transmit binary data by transmission means that deal only with text

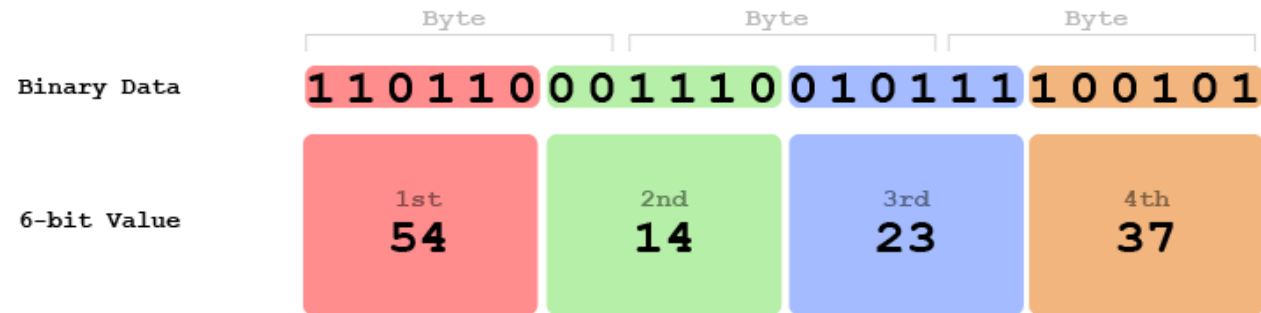
consisting of 64 characters ([A-Z],[a-z],[0-9], "/" e "+")

data:[<mediatype>][;base64],<data>

MIME encoding

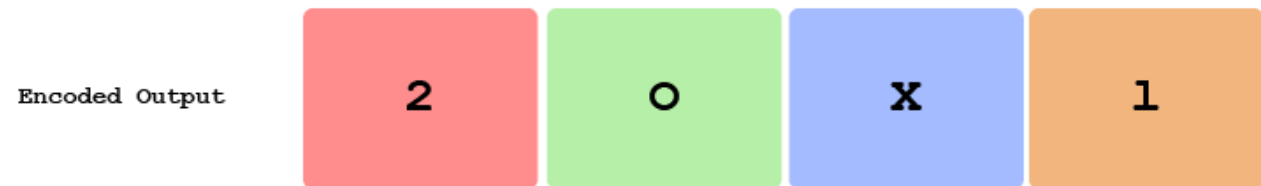
How it works?

Base64 Encoding



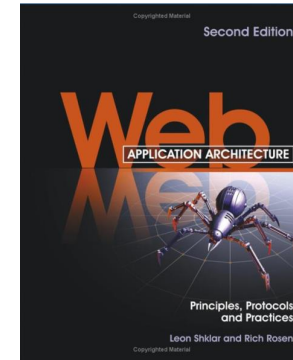
Character Lookup

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
52	53	54	55	56	57	58	59	60	61	62	63														
0	1	2	3	4	5	6	7	8	9	/	+														



Associated Readings

Web Application Architecture, Second Edition
Cap. 3: “Birth of the Web: HTTP”, pages 46 to 49.



**MIME (Multipurpose Internet Mail Extensions) Part One:
Mechanisms for Specifying and Describing the Format of
Internet Message Bodies**

<https://www.ietf.org/rfc/rfc1521.txt>



```
Network Working Group                                N. Borenstein
Request for Comments: 1521                           Bellcore
Obsoletes: 1341                                       N. Freed
Category: Standards Track                           Innosoft
                                                    September 1993

      MIME (Multipurpose Internet Mail Extensions) Part One:
      Mechanisms for Specifying and Describing
      the Format of Internet Message Bodies

Status of this Memo

This RFC specifies an Internet standards track protocol for the
Internet community, and requests discussion and suggestions for
improvements. Please refer to the current edition of the "Internet
Official Protocol Standards" for the standardization state and status
of this protocol. Distribution of this memo is unlimited.

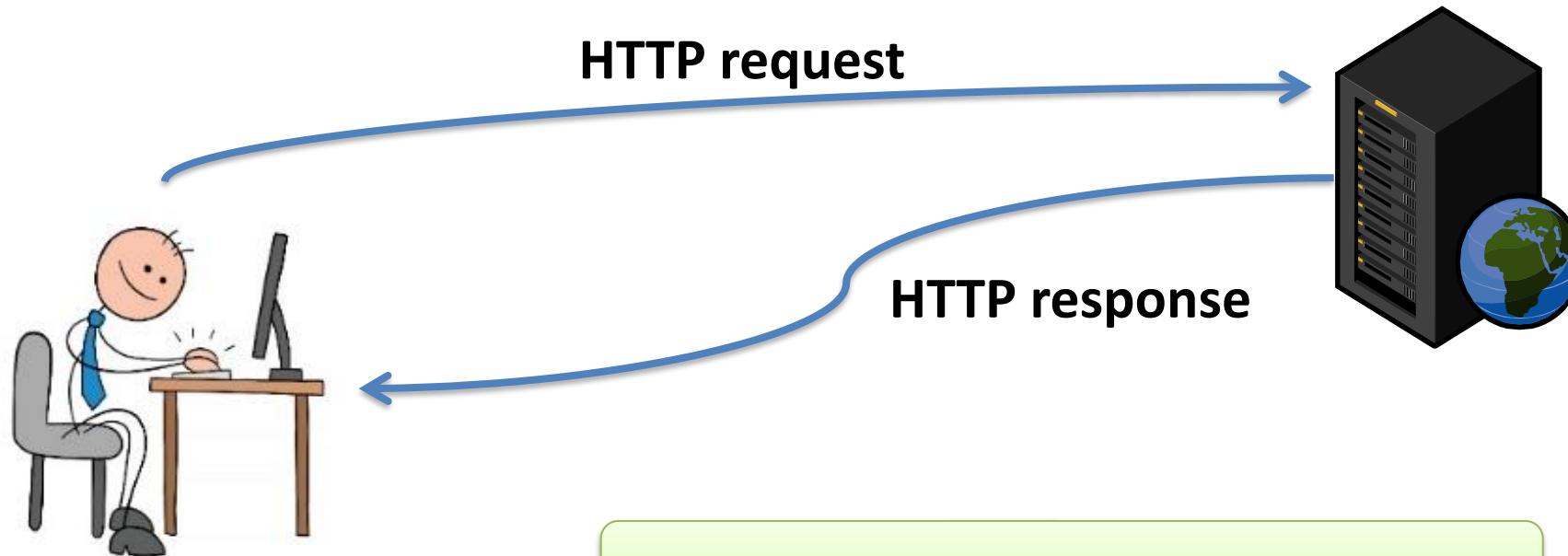
Abstract

STD 11, RFC 822 defines a message representation protocol which
specifies considerable detail about message headers, but which leaves
the message content, or message body, as flat ASCII text. This
document redefines the format of message bodies to allow multi-part
textual and non-textual message bodies to be represented and
exchanged without loss of information. This is based on earlier work
```

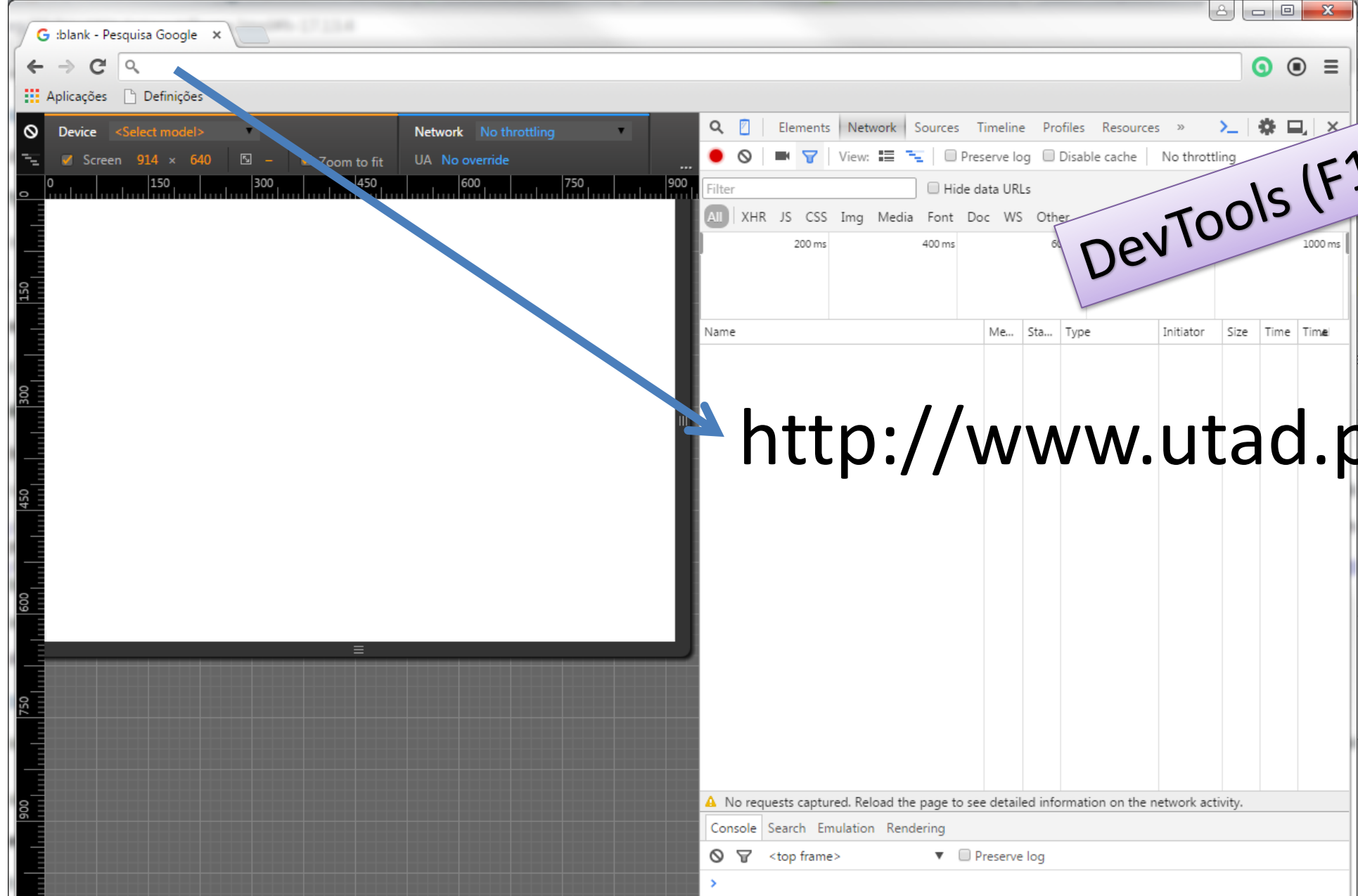
HTTP protocol response status codes

Web Engineering

```
GET /tc/home.html HTTP/1.1
Host: www.tribunalconstitucional.pt
...
```



```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 9934
...
```



DevTools (F12)

<http://www.utad.pt>

UTAD – Universidade de Trás-os-Altos

utad.pt

Aplicações Convert PDF to JPG... The Pirate Bay - Th... Epson printer resett... Pedido - ClimAhorro

Candidatos · Estudantes · Docentes · Funcionários · Alumni · Sociedade · EN

traste

Used to change from HTTP to HTTPS

Name	Method	Status	Type	Initiator	Size	T..	Waterfall
www.utad.pt	GET	307		Other	0 B	8...	
www.utad.pt	GET	200	doc...	www.utad.pt/	70...	1...	
easy-sidebar-menu-...	GET	200	styl...	(index)	(di...	5...	
animate.css?ver=751...	GET	200	styl...	(index)	(di...	5...	
pe-easy-slider.css?v...	GET	200	styl...	(index)	(di...	5...	
panels.min.css?ver=...	GET	200	styl...	(index)	(di...	4...	
jquery.js?ver=1.12.4	GET	200	script	(index)	(m...	0...	
jquery-migrate.min.j...	GET	200	script	(index)	(m...	0...	
font-awesome.css?v...	GET	200	styl...	(index)	(di...	4...	
jquery.cookie.js?ver...	GET	200	script	(index)	(m...	0...	
language-cookie.js?...	GET	200	script	(index)	(m...	0...	

92 requests | 217 KB transferred | 9.3 MB resources | Finish: 4.06 s | DOMContentLoaded: 3.54 s | Load: 3.82 s

Console

What's New

Highlights from the Chrome 77 update

Copy element styles
Right-click an element in the DOM Tree and select Copy > Copy Styles.

Visualize layout shifts
Press Control+Shift+P, run Show Rendering, and enable Layout Shift Regions to visualize content shifts.

http://www.utad.pt/

307

200

UTAD, UMA ECO-UNIVERSIDADE PARA O FUTURO

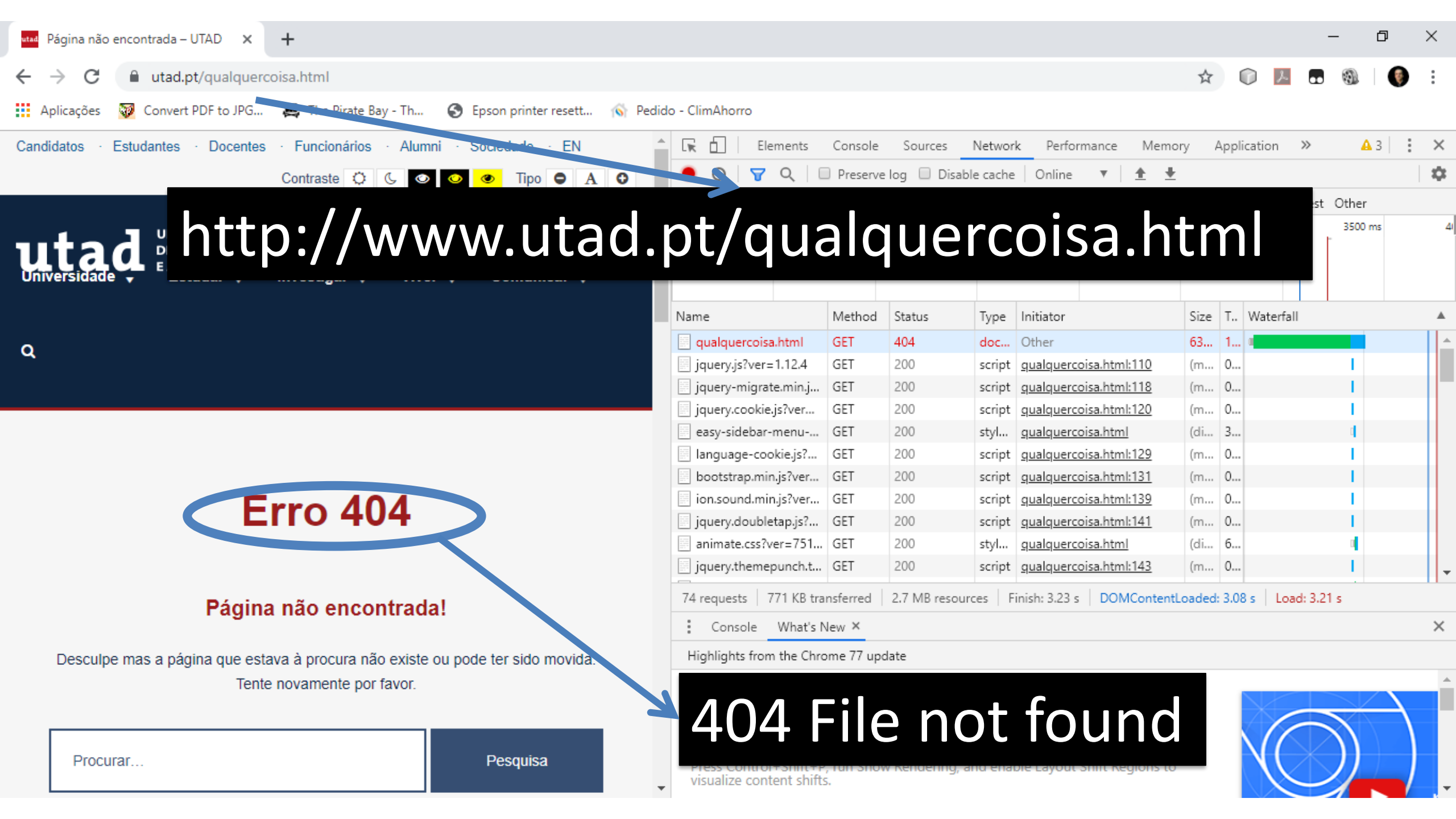
Cursos
Oferta educativa

Serviços Académicos
Informação ao estudante

Sociedade
Ligação à sociedade

Internacional
Estudante internacional

Apoios Sociais
Apoio ao estudante





http://www.utad.pt/pt/index.asp

Method	Result	Type	URL
GET	302	Redirect to: http://www.utad.pt/pt/index.ht...	http://www.utad.pt/
GET	301	Redirect to: http://www.utad.pt/pt/index.asp	http://www.utad.pt/pt/index.html
GET	200	text/html	http://www.utad.pt/pt/index.asp
<div><div>User-AgentMozilla/5.0 (Windows NT 6.0; WOW64; rv:7.0.1) Gecko/20100101 Firefox/7.0.1</div><div>Acceptimage/png,image/*;q=0.8,*/*;q=0.5</div><div>Accept-Languageen-gb,en;q=0.5</div><div>Accept-Encodinggzip, deflate</div><div>Accept-CharsetISO-8859-1,utf-8;q=0.7,*;q=0.7</div></div>			

Firefox

Universidade de Trás-os-Montes e Alto ...

www.utad.pt/pt/index.asp

Google

utad Universidade de Trás-os-Montes e Alto Douro

www.utad.pt

Home Instituição Escolas Ensino e Formação Investigação

Procurar a Alguém?

ACTIVIDADES E ACONTECIMENTOS

Ciclo Cultural da UTAD

Ano Internacional da Química

DESTAQUES

Cerimónia de Abertura

302

HTTP/1.1 302 Object Moved

Location: http://www.utad.pt/pt/index.html

Server: Microsoft-IIS/5.0

Content-Type: text/html

Content-Length: 155

00:03:04.904 0.090 348 268 GET 301 Redirect to: http://www.utad.pt/pt/index.asp http://www.utad.pt/pt/index.html

00:03:05.025 0.794 347 15056 GET 200 text/html http://www.utad.pt/pt/index.asp

00:03:05.162 0.654 406 9987 GET 200 text/css http://www.utad.pt/include/estilos.css

Method	Result	Type	URL
GET	302	Redirect to: http://www.utad.pt/pt/index.ht...	http://www.utad.pt/
GET	301	Redirect to: http://www.utad.pt/pt/index.asp	http://www.utad.pt/pt/index.html
GET	200	text/html	http://www.utad.pt/pt/index.asp

Firefox

Universidade de Trás-os-Montes e Alto ...

www.utad.pt/pt/index.asp

utad Universidade de Trás-os-Montes e Alto Douro

Home Instituição Escolas Ensino e Formação Investigação

Procurar a Alguém?

ACTIVIDADES E ACONTECIMENTOS

Ciclo Cultural da UTAD

Ano Internacional da Química (De Março a Novembro de 2011 - UTAD - Vila Real)

DESTAQUES

Cerimónia de Abertura do Ano Letivo da UTAD

1º Encontro Internacional de Pesquisadores em Saúde e Esporte (12 a 16 de Outubro de 2011 - Montes Claros - MG, Brasil)

XV Jornadas Internacionais de Medicina Veterinária da UTAD (15 e 16 de Outubro de 2011 - UTAD - Vila Real)

II Fórum em Investigação Farmacológica (19 de Outubro de 2011 - Aula Magna - UTAD - Vila Real)

4.ª Edição do Congresso Nacional de Ginástica e 2.ª Edição do Congresso

UTAD - "Award 2011"

Estados Gerais do Conselho Geral da UTAD

Qualificação da UTAD - 2010

Formação Procedimento Concursal

302

301

200

http://www.utad.pt/pt/index.html

http://www.utad.pt/pt/index.asp

Start Stop

Started	Time	Sent	Received	Method	Result	Type
00:03:03.356	1.528	335	276	GET	302	Redirect to: http://www.utad.pt/pt/index.html
00:03:04.904	0.090	348	268	GET	301	Redirect to: http://www.utad.pt/pt/index.asp
00:03:05.025	0.794	347	15056	GET	200	text/html
00:03:05.162	0.654	406	9987	GET	200	text/css

Method	Result	Type	URL
GET	302	Redirect to: http://www.utad.pt/pt/index.ht...	http://www.utad.pt/
GET	301	Redirect to: http://www.utad.pt/pt/index.asp	http://www.utad.pt/pt/index.html
GET	200	text/html	http://www.utad.pt/pt/index.asp

10.3.2 301 Moved Permanently

The requested resource has been assigned a new permanent URI and any future references to this resource SHOULD use one of the returned URIs. Clients with link editing capabilities ought to automatically re-link references to the Request-URI to one or more of the new references returned by the server, where possible. This response is cacheable unless indicated otherwise.

The new permanent URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).

If the 301 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT automatically redirect the request unless it can be confirmed by the user, since this might change the conditions under which the request was issued.

Note: When automatically redirecting a POST request after receiving a 301 status code, some existing HTTP/1.0 user agents will erroneously change it into a GET request.

10.3.3 302 Found


The requested resource resides temporarily under a different URI. Since the redirection might be altered on occasion, the client SHOULD continue to use the Request-URI for future requests. This response is only cacheable if indicated by a Cache-Control or Expires header field.

The temporary URI SHOULD be given by the Location field in the response. Unless the request method was HEAD, the entity of the response SHOULD contain a short hypertext note with a hyperlink to the new URI(s).

If the 302 status code is received in response to a request other than GET or HEAD, the user agent MUST NOT


Status code in generated messages examples by ASP.NET c# :

```
public ActionResult Index()  
{  
    return View();  
}
```

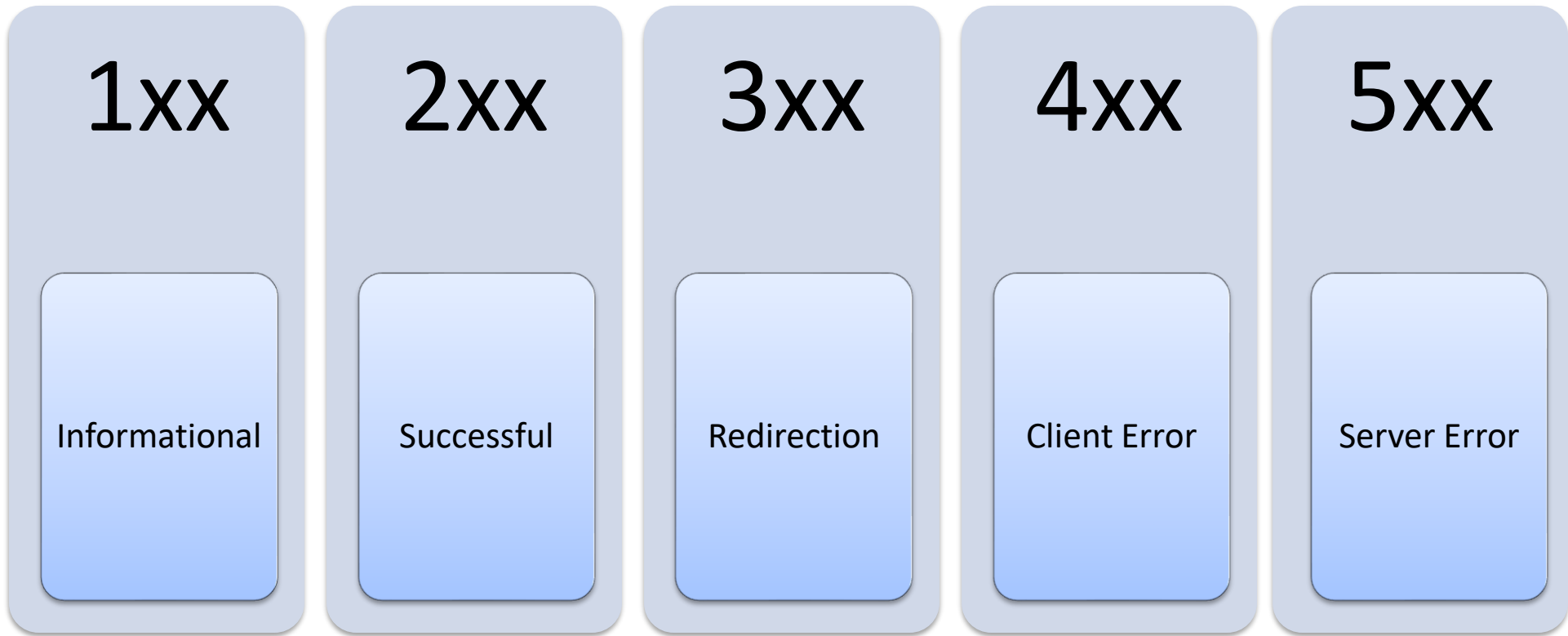


HTTP/1.1 200 OK
Server: Kestrel
Date: Wed, 11 Oct 2023 13:05:44 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 24570
...

```
[HttpPost]  
public ActionResult Index(string Name)  
{  
    return RedirectToRoute("MyRoute");  
}
```



HTTP/1.1 302 Found
Content-Length: 0
Date: Wed, 11 Oct 2023 22:05:48 GMT
Server: Kestrel
Location: /MyController/MyRoute
...



- 1xx (Informational): The request was received, continuing process
- 2xx (Successful): The request was successfully received, understood, and accepted
- 3xx (Redirection): Further action needs to be taken in order to complete the request
- 4xx (Client Error): The request contains bad syntax or cannot be fulfilled
- 5xx (Server Error): The server failed to fulfill an apparently valid request

Informational 1xx

- 100 Continue
- ...

Successful 2xx

- **200 OK**
- ...
- **206 Partial Content**
- ...

Redirection 3xx

- **301 Moved Permanently**
- **302 Found**
- **303 See Other**
- **304 Not Modified**
- **...**
- **307 Temporary Redirect**
- **308 Permanent Redirect**

Client Error 4xx

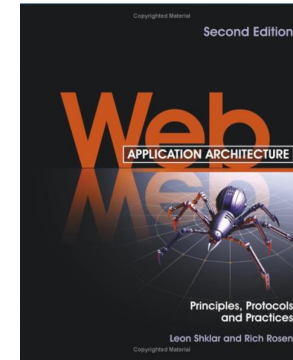
- **400 Bad Request**
- **401 Unauthorized**
- ...
- **403 Forbidden**
- **404 Not Found**
- ...
- **411 Length Required**
- ...

Server Error 5xx

- **500 Internal Server Error**
- ...
- **502 Bad Gateway**
- **503 Service Unavailable**
- ...
- **505 HTTP Version Not Supported**
- ...

Associated Readings

Web Application Architecture, Second Edition
Cap. 3: “Birth of the Web: HTTP”, pages 41 to 44.



Section 15 of the HTTP protocol specification:
“Status Codes”

<https://www.rfc-editor.org/rfc/rfc9110.html#name-status-codes>



part of Hypertext Transfer Protocol -- HTTP/1.1
RFC 2616 Fielding, et al.

10 Status Code Definitions

Each Status-Code is described below, including a description of which method(s) it can follow and any metainformation required in the response.

10.1 Informational 1xx

This class of status code indicates a provisional response, consisting only of the Status-Line and optional headers, and is terminated by an empty line. There are no required headers for this class of status code. Since HTTP/1.0 did not define any 1xx status codes, servers MUST NOT send a 1xx response to an HTTP/1.0 client except under experimental conditions.

A client MUST be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message. Unspecified 1xx status responses MAY be ignored by a user agent.

Proxies MUST forward 1xx responses, unless the connection between the proxy and its client has been closed, or unless the proxy itself requested the generation of the 1xx response. (For example, if a

proxy adds a "Expect: 100-continue" field when it forwards a request, then it need not forward the corresponding 100 (Continue) response(s).)

10.1.1 100 Continue

The client SHOULD continue with its request. This interim response is used to inform the client that the initial part of the request has been received and has not yet been rejected by the server. The client SHOULD continue by sending the remainder of the request or, if the request has already been completed, ignore this response. The server MUST send a final response after the request has been completed. See section 8.2.3 for detailed discussion of the use and handling of this status code.

10.1.2 101 Switching Protocols

The server understands and is willing to comply with the client's request, via the Upgrade message header field (section 14.42), for a change in the application protocol being used on this connection. The server will switch protocols to those defined by the response's Upgrade header field immediately after the empty line which terminates the 101 response.

The protocol SHOULD be switched only when it is advantageous to do so. For example, switching to a newer version of HTTP is advantageous over older versions, and switching to a real-time, synchronous protocol might be advantageous when delivering resources that use such features.

Resume

HTTP PROTOCOL HEADERS

WHAT ARE? WHAT DO THEY CONSIST OF?

WHAT TYPES (OR GROUPS) EXIST?

WHAT ARE FORBIDDEN HEADER NAMES?

WHO INSERTS HEADERS INTO MESSAGES?

KNOW SOME OF THE MOST COMMON HEADERS.

MYME TYPES IN HTTP PROTOCOL

WHAT ARE MIME-TYPES?

WHAT IS THE PURPOSE/IMPORTANCE OF ITS USE?

CAN ITS VALUE BE PROGRAMMATICALLY CHANGED?

HTTP RESPONSE STATUS

GENERALLY SPEAKING, WHAT DO STATE CODES MEAN?

WHAT ARE THE EXISTING GROUPS?

WHAT DOES EACH GROUP REPRESENT?

UNDERSTAND THE MEANING AND CONSEQUENCES OF USING THE MOST COMMON CODES...

Next section

RESOLVING ADDRESSES ON HTTP SERVERS

Readings until October 7th's class

- 3.4 Better Information Through Headers
 - 3.4.1 Support for content types
 - 3.4.2 Caching control
 - 3.4.3 Security

- 3.5 Evolution of the HTTP Protocol
 - 3.5.1 Virtual hosting

- 6 Web Servers
 - 6.1 Basic Operation
 - 6.1.1 HTTP request processing
 - 6.1.2 Delivery of static content
 - 6.1.3 Delivery of dynamic content

- 6.3 Advanced Functionality
 - 6.3.1 Virtual hosting

