

# WEB SISTEMAK

## 2020/2021 IKASTURTEA

HTTP - HyperText Transfer Protocol

Content Length vs Transfer Encoding  
Erantzunaren kodifikazioa

2021-02-12 M (2. zatia)



Web Sistemak by [Oskar Casquero](#) & [María Luz Álvarez](#) is licensed under a [Creative Commons Reconocimiento 4.0 Internacional License](#).

# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH

## ETA TRANSFER ENCODING

- Web zerbitzari batek itzulitako erantzun bat irakurri behar duen web bezero batek, mezuaren gorputza non amaitzen den jakin behar du.
  - HTTP protokoloak mezuaren gorputzaren luzeera adierazteko duen modu bat “Content-Length” goiburua da.
    - » OHARRA: HTTP erantzuna bidali aurretik, “Content-Length” goiburuak erantzunean sartuko den edukiaren tamaina ezagutzea eskatzen du.
- Erantzunean datu kopuru handi bat itzuli behar bada, ez dago datu horien tamaina kalkulatzetik eskaera guztiz prozesatzen denerarte; adibidez: demagun datu base bateko eskaera baten emaitzak web orri batean bistaratu nahi direla.
  - “Content-Length” goiburua erabiltzekotan, mezuaren gorputzean doazen datuen tamaina erantzuna bidali aurretik ezagutu beharra dago. Honek, alde batetik, datu baseko eskaeraren prozesamendua bukatzeari itxoitea suposatzen du. Bestetik, datuak gordetzeko buffer handi bat behar da.
  - Erantzuna bidali aurretik hau prozesatzeko beharrezkoa den denborak eta erantzunak saretik bidaiatzen ematen duen denborak sortutako atzerapenak erabiltzailearen esperientzian eragin negatiboa dauka.
- HTTP protokoloak erantzunaren edukia zatika bidaltzen joateko modua eskeintzen du: “Transfer-Encoding” goiburua “chunked” balioarekin.

# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH

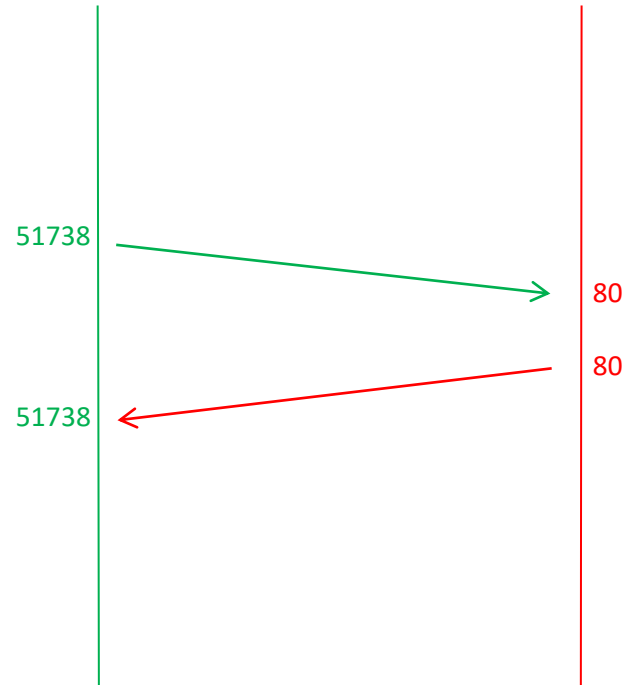
## ETA TRANSFER ENCODING

- “Content-Length” adibidea: zerbitzariak “Hello World!!” testua bidali behar duela suposatuko dugu.

### “Content-Length”-dun HTTP erantzuna

```
HTTP/1.1 200 OK
Date: Thu, 20 Nov 2015 20:25:52 GMT
Last-Modified: Tue, 17 Sep 2015 13:00:02 GMT
ETag: "1a968-3ec-4e693e61bb8b6"
Content-Length: 13
Content-Type: text/plain

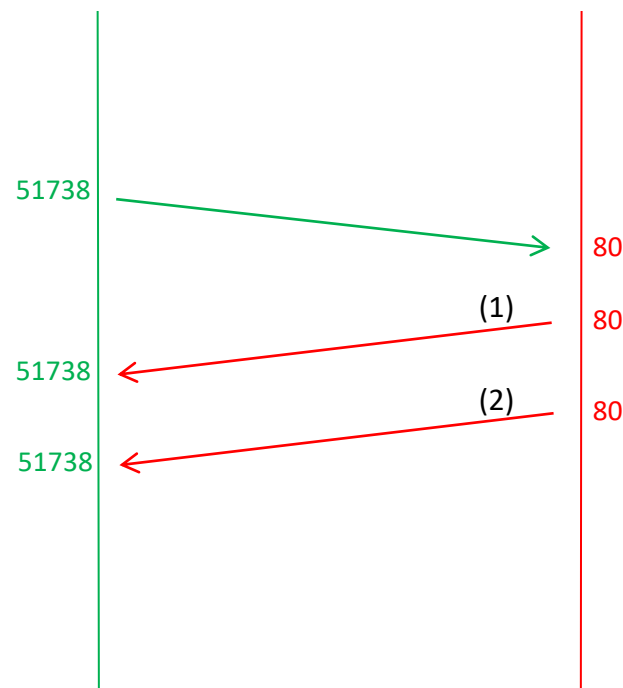
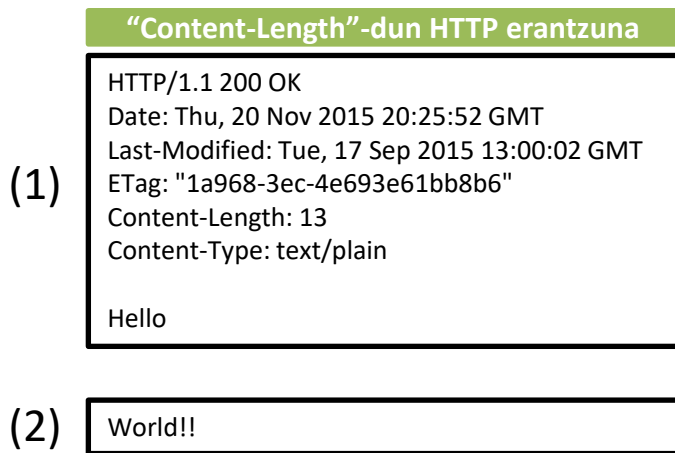
Hello World!!
```



# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH

## ETA TRANSFER ENCODING

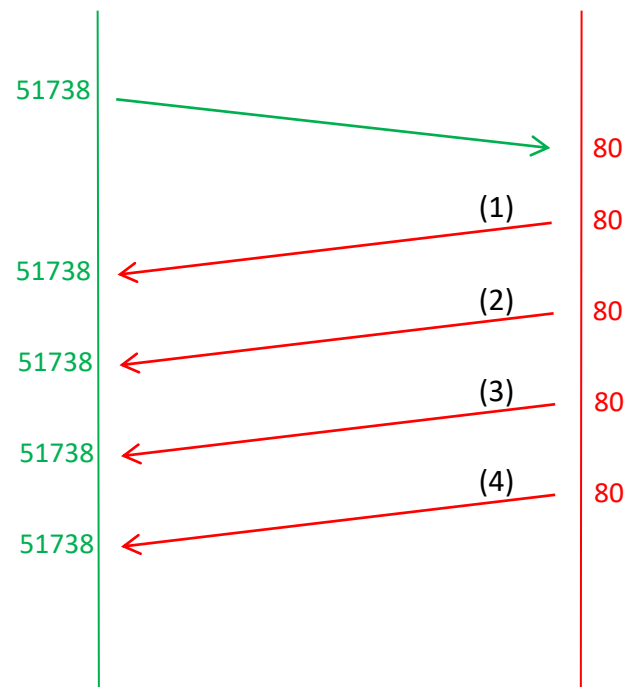
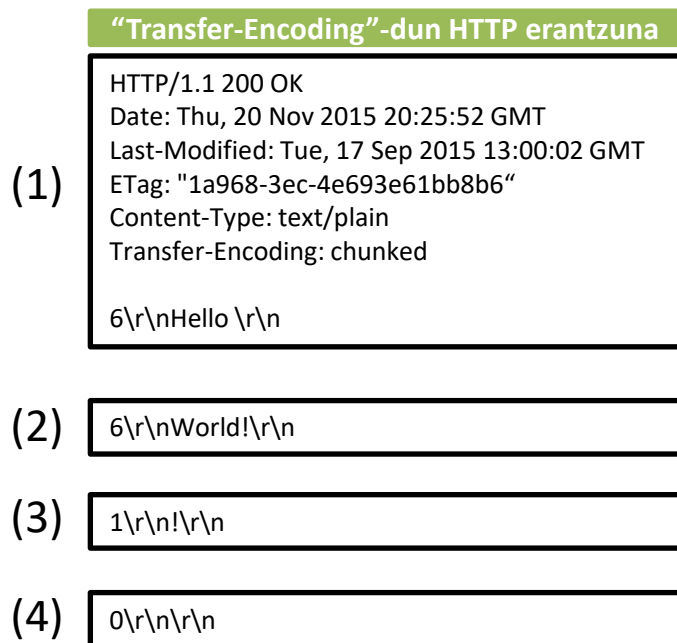
- OHARRA: erantzunaren edukia TCP mailako MTU-a baino handiagoa bada, HTTP mezua hainbat TCP segmentutan bidaliko da.
- Adibidea: TCP mailaren MTU-a 187 zortzikote direla suposatuko dugu.



# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH

## ETA TRANSFER ENCODING

- “Transfer-Encoding: chunked” adibidea: zerbitzariak “Hello World!!” testua bidali behar duela suposatuko dugu.



Zati bakoitzaren edukiaren luzeera notazio hamaseitarrean adierazten da.

2, 3 eta 4 zatiek ez daramate HTTP goibururik. Zuzenean TCP segmentuetan kapsulatuta doaz.

# HTTP-REN FUNTZIONAMENDUA: CONTENT LENGTH

## ETA TRANSFER ENCODING

---

- Bi moduen arteko ezberdintasunak :
  - "Content-Length" goiburuarekin HTTP mezuak ez dira erantzunaren eduki osoa prozesatzen denerarte bidaltzen; aldiz, "Transfer-Encoding" goiburuarekin HTTP mezuak eduki zatiak prozesatu ahala bidaltzen dira.
  - "Content-Length" goiburuarekin HTTP mezuen zatitzea TCP mailak egiten du; aldiz, "Transfer-Encoding" goiburuarekin HTTP mailak berak egiten du.

# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

---

- Nabigatzaile batean ondorengo irudia deskargatu:  
<http://www.httpwatch.com/httpgallery/chunked/chunkedimage.aspx>
- Wireshark erabiliz, HTTP erantzunaren harrapaketa aztertu:
  - Zein da TCP segmentuaren datu eremuaren eta HTTP mezua zatiaren arteko tamainen erlazioa??
    - $3 + 2 + 1024 + 2 = 1031 \rightarrow$  Aztertu zenbaki hauek nondik datozen

# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port==51312

No.	Time	Source	Destination	Protocol	Length	Info
194	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=29175 Win=66048 Len=0
198	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
200	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=30206 Win=65024 Len=0
204	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
206	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=31237 Win=66048 Len=0
207	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
213	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=32268 Win=65024 Len=0
214	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
215	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=33299 Win=66048 Len=0
224	9.000	191.236.16.125	192.168.1.130	TCP	946	[TCP segment of a reassembled PDU]
225	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=34191 Win=65280 Len=0

> Frame 207: 1085 bytes on wire (8680 bits), 1085 bytes captured (8680 bits) on interface 0

> Ethernet II, Src: Comtrend\_cb:de:cb (38:72:c0:cb:de:cb), Dst: IntelCor\_77:98:48 (18:3d:a2:77:98:48)

> Internet Protocol Version 4, Src: 191.236.16.125, Dst: 192.168.1.130

> **Transmission Control Protocol** Src Port: 80 (80), Dst Port: 51312 (51312), Seq: 31237, Ack: 155, **Len: 1031**

**TCP segmentuaren edukiak duen luzeera: 1031 zortzikote**

```
0000  18 3d a2 77 98 48 38 72 c0 cb de cb 08 00 45 00  .=.w.H8r .....E.
0010  04 2f 21 d0 40 00 70 06 52 65 bf ec 10 7d c0 a8  ./!.@.p. Re...}..
0020  01 82 00 50 c8 70 fc 7a d2 aa 99 fe 07 77 50 18  ...P.p.z .....wP.
0030  02 04 58 45 00 00 34 30 30 0d 0a 60 4b b0 2a dc  ..XE..40 0..`K.*.
0040  0a ec 09 6b 02 bb 02 5a c0 55 ac 09 6b 02 bb 02  ...k...Z .U.k...
0050  b5 91 29 6b 02 bb 02 5a 38 15 6b aa ba 95 61 55  ..)k...Z 8.k...aU
0060  61 42 3d 8e 22 44 1b 0c 67 01 20 41 e4 58 d4 d1  aB="D.. g. A.X..
0070  34 52 bc 67 aa 9a 67 4b 8e 62 71 12 1d 5e 07 3e  4R.g..gK .bq..^.>
0080  13 8e 66 07 a1 58 32 c6 96 f0 aa e1 8a 5b c2 ad  ..f..X2. ....[..
0090  e1 56 c6 29 46 e9 1a 64 fa 96 a1 15 9c 3d 64 3f  .V.)F..d .....=d?
```

Packets: 280 · Displayed: 74 (26.4%) · Dropped: 0 (0.0%) Profile: Default



# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port==51312

No.	Time	Source	Destination	Protocol	Length	Info
206	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=31237 Win=66048 Len=0
207	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
213	8.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=32268 Win=65024 Len=0
214	8.000	191.236.16.125	192.168.1.130	TCP	1085	[TCP segment of a reassembled PDU]
215	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=33299 Win=66048 Len=0
224	9.000	191.236.16.125	192.168.1.130	TCP	946	[TCP segment of a reassembled PDU]
225	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=155 Ack=34191 Win=65280 Len=0
229	9.000	191.236.16.125	192.168.1.130	HTTP	59	HTTP/1.1 200 OK (JPEG JFIF image)
230	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [FIN, ACK] Seq=155 Ack=34196 Win=65280 Len=0
231	9.000	191.236.16.125	192.168.1.130	TCP	54	80 → 51312 [FIN, ACK] Seq=34196 Ack=156 Win=132096 Len=0
232	9.000	192.168.1.130	191.236.16.125	TCP	54	51312 → 80 [ACK] Seq=156 Ack=34197 Win=65280 Len=0

HTTP chunked response

- Data chunk (1024 octets)
  - Chunk size: 1024 octets
  - Data (1024 bytes)
  - Chunk boundary: 0d0a
- Data chunk (1024 octets)
- Data chunk (1024 octets)
- Data chunk (1024 octets)
- Data chunk (1024 octets)
- Data chunk (1024 octets)

Mezu zatiaren edukiaren luzeera notazio hamartarrean

Mezu zatiaren edukiaren luzeera notazio hamaseitarrean

HTTP mezu zatien edukiak duen luzeera: 1024 zortzikote

Offset	Hex	ASCII
0130	0d 0a 34 30 30 0d 0a ff d8 ff e1 00 18 45 78 69	.400. ....Exi
0140	66 00 00 49 49 2a 00 08 00 00 00 00 00 00 00	f..II*.. .....
0150	00 00 00 ff ec 00 11 44 75 63 6b 79 00 01 00 04	.....D ucky....
0160	00 00 00 3c 00 00 ff ed 00 2c 50 68 6f 74 6f 73	...<.... .,Photos
0170	68 6f 70 20 33 2e 30 00 38 42 49 4d 04 25 00 00	hop 3.0. 8BIM.%..
0180	00 00 00 10 00 00 00 00 00 00 00 00 00 00 00	.....
0190	00 00 00 00 ff ee 00 0e 41 64 6f 62 65 00 64 c0	..... Adobe.d.
01a0	00 00 00 01 ff db 00 84 00 06 04 04 04 05 04 06	.....
01b0	05 05 06 09 06 05 06 09 0b 08 06 06 08 0b 0c 0a	.....

Frame (59 bytes) Reassembled TCP (34195 bytes) De-chunked entity body (33653 bytes)

Chunk size (http.chunk\_size), 5 bytes

Packets: 280 · Displayed: 74 (26.4%) · Dropped: 0 (0.0%) Profile: Default

# ADIBIDEA:

## IRUDI BATEN ZATIKAKO DESKARGAREN AZTERKETA

### LABURPENA:

- Aurreko irudietan ikusten denez,
  - TCP segmentuaren edukiak duen luzeera: **1031** zortzikote
  - HTTP mezuaren eduki zatiak duten luzeera: **1024** zortzikote
- HTTP zati batek ondorengo eredua du:
  - ZATIAREN\_EDUKIAREN\_LUZEERA\r\nZATIAREN\_EDUKIA\r\n
- HTTP zatia TCP segmentuaren eduki eremuan sartzen da, beraz:
  - $\text{len}("1024") + \text{len}("\r\n") + 1024 + \text{len}("\r\n") = 4 + 2 + 1024 + 2 = \mathbf{1032}$
- $1031 \neq 1032 \rightarrow$  Zergaitik ez datoz bat?
  - Zatiaren luzeera notazio hamaseitarrean sartu behar da!!!
    - 1024 notazio hamartarrean = 400 notazio hamaseitarrean
    - $\text{len}("400") + \text{len}("\r\n") + 1024 + \text{len}("\r\n") = 3 + 2 + 1024 + 2 = \mathbf{1031}$

# HTTP-REN FUNTZIONAMENDUA: MEZUAREN GORPUTZA

---

- Mezuaren gorputza zortzikotez (byte-z) osoturik dago:
  - Testua (testu sinplea, HTML, CSS, XML, JSON, CSS, ...)
  - Eduki binarioa (PDF, bideoa, exekutagarriak, etc.)
- Eduki binarioak aplikazio edo plataforma jakin batzuentzako pentsaturik daude. Adibidez, PDF-ak PDF-irakurle batek soilik ulertu ditzake ondo. PDF bat testu editore arrunt batekin zabaltu edo irteera estandarrera bidaltzen bada, bere edukiaren adierazpen oker bat bistaratzen da, testu oinarria duena.
- Testua programa ugarirekin bistaratu daiteke, baina guztiek ez dute zortzikoteak dekodifikatzeko konfigurazio berdina erabiltzen.
  - Adibidez, jatorrian fitxategi bat [UTF-8](#) erabiliz kodifikatu bazan, "ñ" ikurra C3 B1 byte-ekin gordeko da.
  - Fitxategia [ISO-8859-1](#)-en konfiguratutako testu editore baten bitartez zabaltzen badugu, dekodifikazioaren eraginez ondorengo bi ikurrak bistaratuko dira: Ã±

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...

Kode taula	4. byte	3. byte	2. byte	1. byte
US-ASCII (7 bit)				0
ISO-8859-1 (1 zortzikote)				
UTF-8 (1-4 zortzikote)				0
			1 1 0	1 0
		1 1 1 0	1 0	1 0
	1 1 1 1 0	1 0	1 0	1 0

Taula honetan kontutan hartu beharrekoa:

- latin-1 -ek eta UTF-8 -k US-ASCII kode taula barnean daramate.
- latin-1 -en, US-ASCII -rekiko gehitutako ikurrek "1" dute MSB-n.

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...

Kode taula	Ikurra	Hex		binarioa															
US-ASCII (7 bit)	H	--	48															0	1 0 0 1 0 0 0
	ñ	--	--																
	á	--	--																
ISO-8869-1 (1 zortzikote)	H	--	48															0	1 0 0 1 0 0 0
	ñ	--	F1															1	1 1 1 0 0 0 1
	á	--	E1															1	1 1 0 0 0 0 1
UTF 8 (1 - 4 zortzikote)	H	--	48															0	1 0 0 1 0 0 0
	ñ	C3	B1	1	1	0	0	0	0	1	1	1	0	1	1	0	0	0	1
	á	C3	A1	1	1	0	0	0	0	1	1	1	0	1	0	0	0	0	1

- Demagun ondorengo testua kodifikatu nahi dela: Hola, Iñaki Vázquez!

US-ASCII: Ezin da kodifikatu, “ñ” y “á” ikurrak ez baitira US-ASCII hiztegien existitzen.

Latin-1: 48 6F 6C 61 2C 20 49 **F1** 61 6B 69 20 56 **E1** 7A 71 76 65 7A 21

UTF-8: 48 6F 6C 61 2C 20 49 **C3 B1** 61 6B 69 20 56 **C3 A1** 7A 71 76 65 7A 21

# KODIFIKAZIOARI BURUZKO ARGIBIDE BATZUK...

z

gorde  
ASCII  
latin-1  
UTF-8

5A

zabaldu  
ASCII  
latin-1  
UTF-8

z

gorde  
ASCII

✗

ñ

gorde  
latin-1

F1

zabaldu  
ASCII  
latin-1  
UTF-8

✗

ñ

gorde  
UTF-8

C3 B1

zabaldu  
ASCII  
latin-1  
UTF-8

✗

ñ

Ã ±

# HTTP-REN FUNTZIONAMENDUA: MEZUAREN GORPUTZA

---

- “Content-Type” goiburaren bitartez, zerbitzariak, itzultzen duen eduki mota adierazi dezake: text/html, application/pdf, ...
- Nabigatzaileak “Content-Type” goiburua irakurtzen duenean, bere balioa testu motakoa edo beste mota batekoa ote den egiaztatzen du.
  - Edukia beste mota batekoa bada, bera zabaltzeko plugin-ik konfiguratuta ote duen egiaztatzen du. Horrela ez bada, erabiltzaileari edukia gordetzeko edo dagokion aplikazioarekin zabaltzeko aukera ematen dio.
  - Edukia testu motakoa bada, nabigatzaileak berak interpretatu (bistaratu) egiten du. Horretarako zelan kodifikatuta dagoen jakin behar du.
    - HTTP-k edukiaren kodifikazioa “Content-Type” goiburuan adierazteko aukera du:

Content-Type: text/html; charset=ISO-8859-1

# ADIBIDEA:

## KODIFIKAZIO ARAZO BATEN AZTERKETA

- Wireshark erabilita, nabigatzailean ondorengo URI-a zabaldu eta bere kodifikazio arazoaren arrazoia aurkitu

<http://ws-responsecontentcoding.appspot.com/>





# ADIBIDEA:

## KODIFIKAZIO ARAZO BATEN AZTERKETA

\*Conexión de red inalámbrica

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
535	2...	10.107.26.226	66.102.1.141	HTTP	398	GET / HTTP/1.1
539	2...	66.102.1.141	10.107.26.226	HTTP	309	HTTP/1.1 200 OK (text/plain)

Frame 539: 309 bytes on wire (2472 bits), 309 bytes captured (2472 bits) on interface 0

Ethernet II, Src: CheckPoi\_3a:b1:d9 (00:1c:7f:3a:b1:d9), Dst: IntelCor\_f2:d2:5c (10:0b:a9:f2:d2:5c)

Internet Protocol Version 4, Src: 66.102.1.141, Dst: 10.107.26.226

Transmission Control Protocol, Src Port: 80 (80), Dst Port: 49511 (49511), Seq: 1, Ack: 346, Len: 255

Hypertext Transfer Protocol

HTTP/1.1 200 OK\r\n

Cache-Control: no-cache\r\n

Content-Type: text/plain; charset=latin-1\r\n

Content-Encoding: gzip\r\n

Vary: Accept-Encoding\r\n

Date: Thu, 04 Feb 2016 08:25:33 GMT\r\n

Server: Google Frontend\r\n

Content-Length: 39\r\n

\r\n

[HTTP response 1/1]

[Time since request: 0.147155000 seconds]

[Request in frame: 535]

Content-encoded entity body (gzip): 39 bytes -> 19 bytes

Line-based text data: text/plain

Hola I\303\261aki P\303\251rez!

0000 48 6f 6c 61 20 49 c3 b1 61 6b 69 20 50 c3 a9 72 65 7a 21

0010

Hola I.. aki P..r ez!

Edukia notazio hamaseitarrean

Frame (309 bytes) Uncompressed entity body (19 bytes)

Text item (text), 19 bytes

Packets: 646 · Displayed: 2 (0.3%) Profile: Default

- Nabigatzaileak edukia “text/plain” motakoa dela ikusten du, beraz, bera arduratzen edukia interpretatzeaz.
- Nola interpretatu behar du testua nabigatzaileak? Hau da, zein kodifikazio taula erabili behar du? “charset” parametroak adierazten duena; kasu honetan: latin-1.

48 → H  
6F → o  
6C → l  
61 → a  
20 →  
49 → I  
C3 → Ñ  
B1 → ±

# ADIBIDEA:

## KODIFIKAZIO ARAZO BATEN AZTERKETA

- Zergaitik gertatzen da hau?
  - Zerbitzaria programatu duenak edukia latin-1 -en kodifikatuta dagoela adierazten duen kode lerroa idatzi du:  
**`self.response.charset = 'latin-1'`**
  - Baina ez du kontuan izan *write* metodoa erabitzen denean zerbitzariak edukia UTF-8 -n kodifikatzen duela:

**`self.response.write("Hola Iñaki Pérez!")`**

**Content-Type: text/plain; charset: latin-1**

```
import webapp2
class MainHandler(webapp2.RequestHandler):
    def get(self):
        self.response.content_type = 'text/plain'
        self.response.charset = 'latin-1'
        self.response.write("Hola Iñaki Pérez!")

app = webapp2.WSGIApplication([('/', MainHandler)], debug=True)
```